How well do transformer models learn impossible languages?

Sinthalapadi Ram Janarthana Raja

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2025

Abstract

Recent studies has shown that transformer models learn possible languages better than impossible languages. However, the evidence presented is coarse grained, making it difficult to characterise the differences in the models' language capabilities. Using pairs of minimally differing sentences, I probe the linguistic knowledge of models trained on these languages across a variety of grammatical phenomena. My results show that while language learning is impeded in the impossible setting, these models still acquire a lot of linguistic knowledge. I present an analysis of the dependency parses of sentences from impossible languages to quantity their hierarchical complexity, and I also test the models for an information locality bias and a hierarchical bias. I find that they exhibit the former but not the latter, reporting these findings and the evaluation methodology used to inform future related work.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Sinthalapadi Ram Janarthana Raja*)

Acknowledgements

I would like to thank Professor Sharon Goldwater and Coleman Haley for their invaluable guidance and support throughout this project. I am also grateful to my classmates and friends for many thoughtful discussions, and to my family for their never-ending support. Finally, my thanks go to the city of Edinburgh for being a wonderful home for these memories.

Table of Contents

Chapter 1

Introduction

A crucial part of understanding a system is knowing when it fails and why. Linguists have long sought to understand how humans learn languages, but are limited in their experiments because they cannot directly manipulate the inner workings of the human mind. Transformer-based language models have shown impressive language capabilities and expose the ability to manipulate their internal architecture, offering a chance to extensively experiment with them and explore their learning mechanisms. An ensuing question is their viability as analogous language learners, that is whether they share learning/inductive biases with humans.

Recent work has shown evidence supporting this to varying degrees. Kallini et al. [15] studied how transformer-based models learned impossible languages, ones that contain grammatical patterns that are hypothesized to never occur in any possible human languages. They cited evidence like trends in perplexity to demonstrate a preference against learning impossible languages. Someya et al. [25] offered an information-theoretic perspective on such languages, using cross-entropy to show that these models exhibited an information locality bias similar to humans.

However, current research uses learning evaluations reliant on statistical metrics which don't directly measure whether a model has acquired abstract, generalisable linguistic knowledge. I hypothesize that if language models showed a preference for possible languages over impossible ones, it should be reflected in linguistically motivated evaluations. Moreover, two systems that succeed or fail in the same tasks may not be doing so for the same reasons, calling for more a detailed test of the learning biases of these models. A key bias not tested in these studies is sensitivity to hierarchical structures in language. This is when a word far from other words in a linear sequence can still influence them. For instance, in the sentence 'I shot the elephant in my pajamas', the phrase 'in my pajamas' describes the subject 'I' despite being closer to the object 'the elephant'. Modeling this relationship necessitates a dimension beyond linear structures, giving rise to hierarchical structures. The presence of this inductive bias is an important characteristic of humans [3] but remains debatable for these models, presenting a gap in the understanding of their viability as good models of human language learners.

In this thesis, I make use of linguistically motivated evaluations to characterise with granularity how well transformer-based language models learn impossible languages to evaluate their similarity to humans as language learners. I leveraged the Benchmark of Linguistic Minimal Pairs (BLiMP) [29] to test language models' linguistic knowledge on various grammatical phenomena using minimally differing sentences. I developed impossible BLiMP datasets that consisted of minimally differing sentences that were perturbed following the rules of the impossible languages. I curated 15 grammatical phenomena to test a variety of syntactic and semantic knowledge and generated the corresponding impossible BLiMP datasets. I then evaluated the models trained on English and impossible languages on these tests to get a holistic view of their linguistic capabilities. My results showed that language learning for these models was impeded in the impossible language setting, confirming the findings of prior work. However, they also highlighted tasks where impossible language-based models outperformed English-based ones, indicating that they still learned well from impossible languages and possibly in ways unintuitive to humans.

Further, I investigated the presence of two biases-information locality bias and hierarchical bias- hypothesizing that it would translate into the performance on related evaluation tasks. These biases are relevant because the impossible languages manipulate related language properties as demonstrated in this thesis. I showed that impossible languages perturb a language's hierarchical structure, and conducted a hierarchical analysis and defined dependency parse tree-based metrics to capture hierarchical complexity. I investigated how well these metrics captured the difficulty of language learning, hypothesizing that they could predict the performance ranking of impossible language-based models on related tasks. My results indicated that these models showcase an information locality bias and lack a hierarchical bias, yielding insights about the model's biases and the evaluation techniques used, to provide a valuable foundation for future studies.

Chapter 2

Background

The main questions I pursue in this thesis are whether transformer-based language models exhibit preferences when learning possible and impossible languages, characterising how well they learn these languages, and understanding how hierarchical and information locality biases influence this. To inform the reader about relevant background topics, this section presents key concepts and established knowledge in this domain that form the basis of my research methodology.

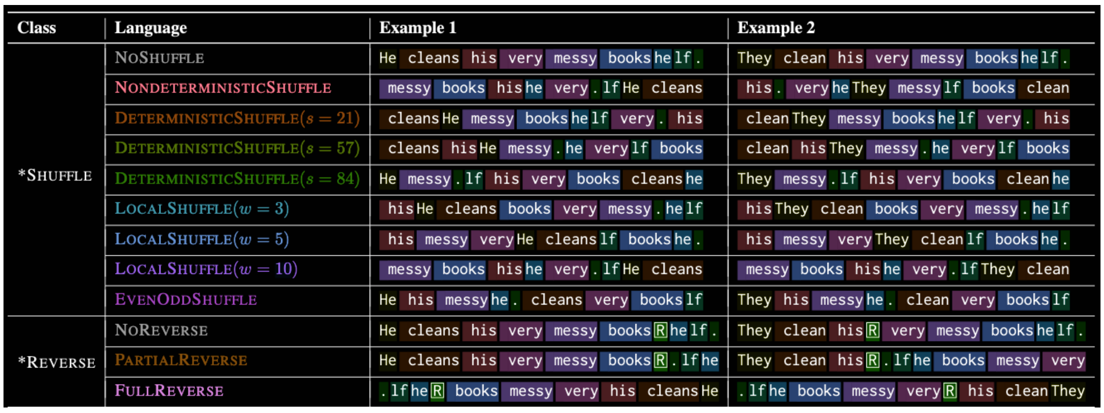
2.1 Impossible languages

The formal definition of an 'impossible language' is the subject of ongoing discussion within linguistics, primarily due to the difficulty in establishing linguistic properties that are truly universal across all natural languages [15]. Notwithstanding this ambiguity, there exist methods for generating impossible languages that involve systematically manipulating the word order of a possible language. These methods result in an impossible language since they also manipulate the grammatical rules tied to linear word order, which violates the hypothesized universal recursive nature of human languages [19]. Following the methodology of [15], I adopt two categories of impossible languages defined by them that are derived from English: **Shuffle** and **Reverse**. My study focuses only on these two impossible language families from their definitions and omits others that included syntactic- or semantic- based perturbations for reasons elaborated in Section 3.1.

The Shuffle family of languages involve manipulating the word order of a sentence by shuffling words within a small window deterministically (giving rise to local shuffles with varying window sizes '*w*'), by shuffling them across the whole sentence deterministically (giving rise to deterministic shuffles with varying shuffling seeds '*s*'), by shuffling them across the whole sentence non-deterministically (referred to as `nondeterministic shuffle`), or by reordering them such that all even-indexed words appear first followed by all odd-indexed words (referred to as `even-odd shuffle`). In this study, I focus on local shuffles with window sizes of 3, 5, and 10, referred to as `local shuffle3`, `local shuffle5` and `local shuffle10` respectively, and focus on one deterministic shuffle with the seed set to 21, referred to as `deterministic shuffle21`.

The Reverse family of languages involve reversing the entire list of words (giving rise to `full reverse`), or randomly selecting a position in sentence and only reversing the subsequent words (giving rise to `partial reverse`). This family also introduces a special token 'R' that is added to the sentence. It is used to mark the position of reversal in `partial reverse` sentences, or inserted at a random position in `full reverse` sentences to maintain a consistent vocabulary within the language family.

Figure 2.1 displays two sample English (referred to as 'NoShuffle' in the table) sentences alongside their canonical version in each impossible language. Crucially, a detail surfaced here is that the perturbations are applied at the token level, not the word level, explaining why words appear to be broken in the impossible sentences. This approach is preferred because transformer models process text by tokens and not words, which are produced by its tokeniser.



Class	Language	Example 1	Example 2
	NoShuffle	He cleans his very messy bookshelf.	They clean his very messy bookshelf.
	NondeterministicShuffle	messy books hishe very.lfHe cleans	his. veryheThey messylf books clean
	DeterministicShuffle($s = 21$)	cleansHe messy bookshelf very. his	cleanThey messy bookshelf very. his
	DeterministicShuffle($s = 57$)	cleans hisHe messy.he verylf books	clean hisThey messy.he verylf books
*Shuffle	DeterministicShuffle($s = 84$)	He messy.lf his very books cleanshe	They messy.lf his very books cleanhe
	LocalShuffle($w = 3$)	hisHe cleans books very messy.helf	hisThey clean books very messy.helf
	LocalShuffle($w = 5$)	his messy veryHe cleanslf bookshe.	his messy veryThey cleanlf bookshe.
	LocalShuffle($w = 10$)	messy books hishe very.lfHe cleans	messy books hishe very.lfThey clean
	EvenOddShuffle	He his messyhe. cleans very bookslf	They his messyhe. clean very bookslf
	NoReverse	He cleans his very messy booksRhelf.	They clean hisR very messy bookshelf.
*Reverse	PartialReverse	He cleans his very messy booksR.lfhe	They clean hisR.lfhe books messy very
	FullReverse	.lfheR books messy very his cleansHe	.lfhe books messy veryR his cleanThey

Figure 2.1: List of impossible languages with examples, from [15]. Differently colored blocks represent different tokens

2.2 Hierarchical structures in language

Human languages are known to exhibit hierarchical structures [17, 11] that are often modeled using tree-like structures, primarily through two formalisms: constituency parses and dependency parses. A constituency parse groups words into syntactic constituents which are then related by grammatical rules [14]. A dependency parse, in contrast, models the binary grammatical relationships between words directly. In my study, I focus on dependency parses because their emphasis on pairwise relationships closely mirrors one of the transformer model's key underlying mechanisms: attention [27].

2.2.1 Dependency parsing

The dependency parse of a sentence produces a directed graph where vertices correspond to words and edges (or arcs) capture the grammatical relationships between them [14]. A valid dependency tree typically satisfies several constraints, such as having a single, designated root vertex with no incoming arcs and a unique path from the root to every other vertex. The arcs are directed from a 'head' to its 'dependent' and are labeled according to their grammatical function, as assigned by the parser. Figure 2.2 provides an example of a dependency parse tree, with the arcs annotated by their type.

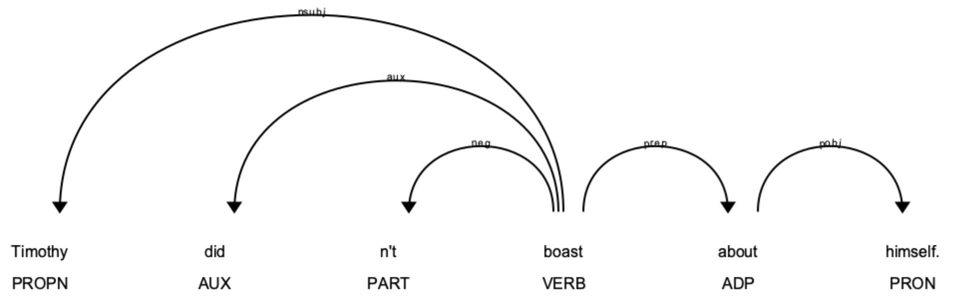


Figure 2.2: Dependency parse tree of an English sentence

A salient features of dependency parse trees is projectivity, a qualitative measure of hierarchical complexity. An arc from a head to its dependent is projective if there is a path from the head to every word that lies between them in the linear sentence; a tree is projective if all its arcs are projective [14]. In languages with stricter word order like English, non-projective sentences are infrequent and are considered a signal of complexity [16]. It has been theorized that this is because producing crossing dependencies incurs an increased processing cost for humans [31], possibly arising from memory limitations [8]. Therefore, non-projective sentences in a language make it more difficult for humans to process and examining how non-projectivity affects the learning process for transformer-based models can inform if they share this bias.

Hierarchical structure is considered a universal feature of human language [3], with some theories suggesting that language co-evolved with humans to include it [4, 18]. This marks hierarchy as a key inductive bias in human language learning. Whether transformer models exhibit a similar hierarchical bias is a subject of ongoing debate, with varying and sometimes contradictory conclusions. For example, [23] showed evidence that these models did not, whereas [21] argued that they can develop this capacity with sufficient training (a phenomena they termed 'structural grokking'). Investigating the source of this bias, [20] suggested that the pretraining of models can impart a hierarchical preference, while [1] argued that it was a consequence of the training objective. My study aims to provide a new perspective grounded in linguistic evaluations. I differentiate my work from previous studies by using impossible languages that systematically manipulate hierarchical structure, allowing me to study the effects of such manipulations on models acquiring knowledge about structural grammatical phenomena.

2.3 BLiMP

The primary evaluation framework I use in this study to assess language learning is the Benchmark of Linguistic Minimal Pairs (BLiMP) [29]. BLiMP offers a targeted assessment of a language model's grammatical knowledge by presenting it with pairs of sentences that differ by a single word, one grammatically correct and the other incorrect [29]. A model's preference for a sentence is measured by which of the two it assigns a higher probability to. The model scores one point if it prefers the grammatical sentence and the sum total score, expressed as a proportion of the total dataset (typically 1000 pairs), is reported as the model's accuracy on that test. The accuracy signals the model's sensitivity to that grammatical distinction and offers a fine-grained view of its syntactic, morphological, and semantic competence when measured across different datasets. An advantage of this evaluation is that the accuracy of a model on a dataset can be directly compared to those of other models, and even to human accuracies.

Each BLiMP dataset contains pairs of sentences that differ by a single word, and the choice of this word determines the linguistic phenomenon being tested. There are 12 categories of grammatical phenomena defined by [29], each containing datasets to test specific variations of that phenomenon or to vary properties like sentence length, for a total of 67 datasets. These categories include morphological phenomena like ‘Subject Verb Agreement’ and ‘Determiner Noun Agreement’, semantic phenomena like ‘Irregular Form’ and ‘Quantifiers’, and complex structural phenomena like ‘Island Effects’ and ‘Filler-Gap’. Table 2.1 shows examples of minimal pairs and the grammatical phenomena they test from some of the BLiMP datasets used in this study (an extended version containing examples from all BLiMP datasets used in this study is Table A.1 in the Appendix).

Grammatical Phenomena	Dataset Name	Grammatical Example	Ungrammatical Example
Anaphor Agreement	Anaphor Gender Agreement	Paula cares for **herself**.	Paula cares for **himself**.
Anaphor Agreement	Anaphor Number Agreement	Todd is admiring **himself**.	Todd is admiring **themselves**.
Argument Structure	Animate Subject Passive	The soda was drunk by some **child.**	The soda was drunk by some **diagnosis**.
Determiner Noun Agreement	Determiner Noun Agreement with Adjective	Gerald could examine **that** long essay.	Gerald could examine **those** long essay.
Irregular Forms	Irregular Past Participle Adjective	The **hidden** coffee wasn’t hot.	The **hid** coffee wasn’t hot.

Table 2.1: Examples of minimal pairs for selected grammatical phenomena from BLiMP

BLiMP datasets are constructed automatically from linguistically-crafted templates populated with randomly chosen lexical items (e.g., nouns, verbs, adjectives). To ensure that automatically generated sentences represented a valid contrast in grammaticality, [29] had human annotators verify samples from each dataset. In my study, I leverage their publicly available code and datasets to build my evaluations.

2.3.1 Using BLiMP to test for inductive biases

Many grammatical phenomena tested in the BLiMP datasets/tasks, such as ‘Subject Verb Agreement,’ ‘Island Effects,’ and ‘Filler Gap,’ are represented hierarchically by linguists to reflect the hypothesized structures native to humans [29, 3]. However, I argue that strong performance on a BLiMP task does not guarantee the model’s usage of hierarchical representation to judge grammaticality. For instance, in a task like ‘Anaphor Gender Agreement,’ when the two related words relevant to solving the task are close together (e.g., ‘Paula’ and ‘herself’ in the example in Table 2.1), a model could rely on local co-occurrence statistics to assess grammaticality successfully. This strategy depends on surface-level patterns, not hierarchical structures, yet can lead to a high accuracy when evaluated. A stronger test of a model’s ability to generalise such knowledge involves controlling for such ’shortcuts’, for instance by increasing the distance between related words. In BLiMP, each grammatical phenomena contains multiple datasets that vary different properties to control for confounders like sentence length or word choices. However, not all grammatical phenomena control for the distance between related words (e.g 'Filler Gap' has 'WH Question Subject Gap' and 'WH Question Subject Gap Long Distance' tasks which do, while 'Anaphor Agreement' contains 'Anaphor Gender Agreement' and 'Anaphor Number Agreement' tasks which control for lexical properties instead), highlighting a key characteristic of BLiMP datasets: construction choices affect the kind of knowledge being tested.

This detail motivates an analysis to classify BLiMP tasks based on the knowledge that could be used to solve them. Such a classification is crucial because it allows me to probe how a model solves a task, and understanding this helps illustrate its inductive biases [7]. In this study, since I am interested in an information locality bias and a hierarchical bias, I probe whether tasks can be solved using **local information** or must rely on **hierarchical knowledge**. I present an analysis of the BLiMP tasks used in this study along this axis in Section 3.2.2.

To identify the type of knowledge a model uses to solve a task, I propose two evidentiary signals. The first is a model's absolute performance on that task. Strong performance on a task designed to test a type of knowledge indicates a propensity to learning it from the training data. However, performance alone can be misleading if the training data itself is biased towards a certain type of knowledge. I suggest that a stronger signal is observing performance gradients—that is, how a model's performance on a task changes when the training data is controllably manipulated to hamper that knowledge. This involves defining a property of the training data that can be a good proxy for that type of knowledge, and subsequently testing if there is a significant relationship between that property and a model's performance on a task. I introduce the properties used to test for an information locality bias and a hierarchical bias in Section 3.3.

2.4 Related work

The research conducted by Kallini et al. [15] provides a foundational exploration into the use of impossible languages for probing the learning capacities of language models. They defined impossible languages, as described in the previous section, and manipulated an English corpus to produce canonical impossible corpora. The English corpus they used was the BabyLM corpus, a 100-million-word dataset developed to approximate the linguistic data available to a child learner [28]. For the purpose of their study, the exact choice of training corpus was less important and more likely influenced by resource constraints, hence preferring a smaller dataset. They trained GPT-2 based transformer models [24] on the impossible corpora to produce various impossible language-based models. The GPT-2 transformer architecture is a flavour of the base transformer architecture introduced by [27], and was likely also chosen for being a good fit for their resource constraints. Crucially, they ensured that all models were trained under identical constraints, including dataset size, training duration, and computational hardware.

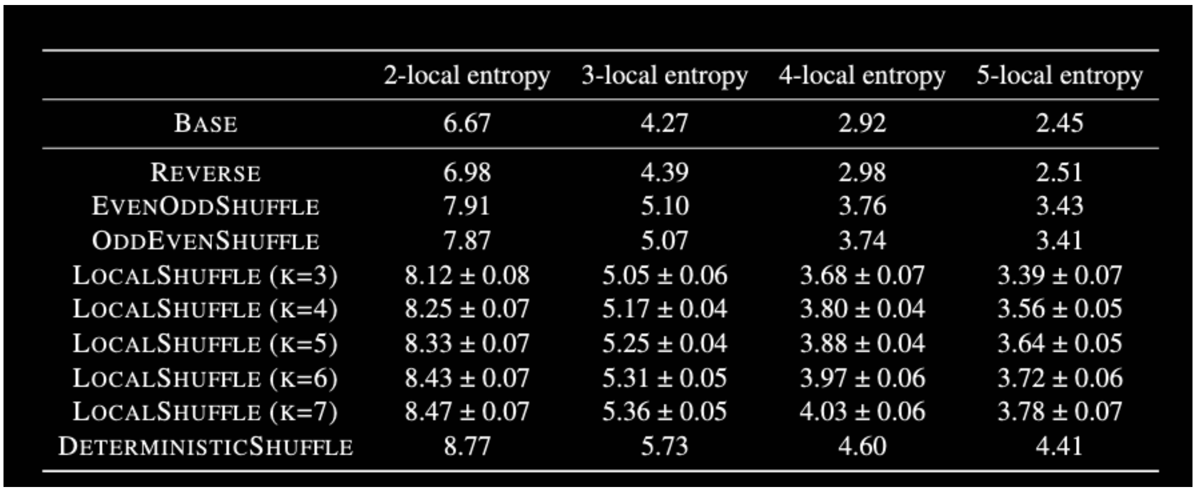
They primarily assessed model learnability by measuring training-time perplexity on a held out test set and showed that impossible language-based models had less steep curves for perplexity, suggesting they fared worse on learning such languages compared to a baseline (an English-based language model). Perplexity is a metric that measures the inverse probability assigned to a test set by a language model [14], and while it is a popular metric to measure how good a language model is, its usage here raises concerns as it is too coarse grained to offer direct evidence regarding the learnability of a language. My study extends their work by employing a more robust set of evaluation benchmarks based on BLiMP, to achieve a comprehensive understanding of how well a model learns a language. I also leverage their work by using their pretrained language models in my evaluations, which were made publicly accessible on HuggingFace [13].

One important comparison to make is the vocabulary used when generating BLiMP datasets and the vocabulary of BabyLM [9], the training dataset of the pretrained models from [15]. This is key to ensuring that my evaluation is fair and that the results can be interpreted confidently. The BabyLM training corpora are limited in size (one track consisting of 10M words and another consisting of 100M words of text) and come mainly from child-directed speech, subtitles, and simple books [9]. The BLiMP benchmark in contrast uses a varied lexicon of over 3,000 distinct English words including proper nouns to populate its minimally-differing sentence pairs [29]. While common grammatical words and simple nouns/verbs tend to overlap, specialized or rare words in BLiMP are often absent from BabyLM's child-directed corpora. As a result, no BabyLM variant fully contains the BLiMP vocabulary. However, I maintain that this does not pose a problem when evaluating a model's linguistic knowledge since by definition such rules are robust to instances of specific words. Moreover, no measures were taken to ensure this kind of overlap when training models that were evaluated by the authors of BLiMP in their study [29], likely based on the same principle.

Building on the work of [15], Xu et al. [30] explored a softer learning preference of transformer models by using implausible languages instead of impossible languages. These languages were created by introducing non-harmonic word order to possible languages to create ones that lied closer to the boundary of possibility [30]. Similar to [15], they measured perplexity on a held out test set and found that models exhibited a preference for learning possible languages over typologically implausible languages. They also made use of BLiMP-style evaluations to check if implausible languages impeded the learnability of specific linguistic phenomena. They created implausible versions of all of BLiMP's 67 datasets by applying the same non-harmonic word order swapping algorithm to the minimal pair sentences. However, they down-sampled each dataset (only picking 5 out of 1000 examples per dataset) and aggregated them across the categories of grammatical phenomena, to produce 12 test suites that models were tested on. While these macro accuracies indicated a slight worsening of performance in the implausible language setting, a limitation of this methodology is that it does not utilise BLiMP-style datasets to the fullest degree to tease out specific linguistic knowledge. My thesis expands the size of the test suites (using 1000 examples) and reports micro scores to offer nuanced insights into a model's linguistic knowledge.

Taking an information-theoretic approach to analysing impossible languages, Someya et al. [25] designed a framework for capturing the local information of a language to predict how easy or hard it would be for transformer models to learn them. They quantified local information by defining m-local entropy, an information-theoretic metric that measured the local predictability of a language within a window of size *m*. They presented a spectrum of impossible languages based on their m-local entropy as reported in Figure 2.3.

They focused on local information because humans have shown a sensitivity to information locality in languages [10], and designed their experiments to test if transformer language models exhibited a similar preference. They modeled aspects of their methodology on the study done by [15], adopting their definitions of impossible languages but omitting `partial reverse` and `nondeterministic shuffle`, and used a different base corpus to create their impossible corpora. They also included three new impossible languages: local shuffles with window sizes *'w'* of 2 and 4, and `odd-even shuffle` which followed the same principle as `even-odd shuffle` but swapped the order of tokens. Moreover, they looked at learnability from a different lens, adopting a statistical definition called next-symbol cross entropy. Having measured the m-local entropy of each language's dataset (this served as a proxy for the local information of a



	2-local entropy	3-local entropy	4-local entropy	5-local entropy
Base	6.67	4.27	2.92	2.45
Reverse	6.98	4.39	2.98	2.51
EvenOddShuffle	7.91	5.10	3.76	3.43
OddEvenShuffle	7.87	5.07	3.74	3.41
LocalShuffle (k=3)	8.12 ± 0.08	5.05 ± 0.06	3.68 ± 0.07	3.39 ± 0.07
LocalShuffle (k=4)	8.25 ± 0.07	5.17 ± 0.04	3.80 ± 0.04	3.56 ± 0.05
LocalShuffle (k=5)	8.33 ± 0.07	5.25 ± 0.04	3.88 ± 0.04	3.64 ± 0.05
LocalShuffle (k=6)	8.43 ± 0.07	5.31 ± 0.05	3.97 ± 0.06	3.72 ± 0.06
LocalShuffle (k=7)	8.47 ± 0.07	5.36 ± 0.05	4.03 ± 0.06	3.78 ± 0.07
DeterministicShuffle	8.77	5.73	4.60	4.41

Figure 2.3: m-local entropy values for the Base (English) corpus and the different impossible language corpora, by Someya et al. [25]

language), they trained transformer language models on these corpora and measured how well they learned each language [25].

Their results showed a positive correlation between the difficulty of learning a language and its m-local entropy (strongest when $m = 4$) and suggested that these models exhibited an information locality bias similar to humans [25]. However, a limitation of their definition of learnability is that it can overlook semantic and structural knowledge as these are difficult to isolate using such statistical measures. Moreover, their characterisation of impossible languages focused primarily on local statistics and did not consider the effects of hierarchical structure on learning, even though their impossible languages manipulate this aspect of language. In this thesis, I address these gaps by creating metrics that characterise a language structurally, and analysing whether it can predict the learnability of a language.

Chapter 3

Methodology

This chapter details the methodology I developed to investigate the learning preferences of transformer models. I begin by stating the core hypotheses of my study in Section 3.1. In Section 3.2 I describe the process of creating ‘impossible’ BLiMP-style tasks used to measure the linguistic competence of models trained on impossible languages. Section 3.3 details the systematic characterisation of impossible languages used to analyse the influence of the specific inductive biases. Finally, Section 3.4 outlines the experimental setup used for model evaluation.

3.1 Hypotheses

The goal of my study is to find linguistically motivated evidence for the learning preferences of transformer-based language models. Beyond just observing these preferences, I also aim to investigate the role inductive biases play in influencing them. This leads me to the following hypotheses:

1. **H1 (Overall Performance):** GPT-2 models trained on impossible languages will achieve lower overall scores than an equivalent model trained on English when evaluated on BLiMP-style linguistic tasks.

2. **H2 (Information Locality Bias):** The performance of all GPT-2 models on tasks solvable using local information will be ranked by the information locality of their training language. I predict this ranking will be according to the language’s m-local entropy.

3. **H3 (Hierarchical Bias):** The performance of all GPT-2 models on tasks requiring hierarchical knowledge to be solved will be ranked by the structural complexity of

their training language. I predict this ranking will be according to the language's dependency parse-based metrics, proposed in this study.

Hypothesis **H1** is motivated by the prior work which found that transformer models exhibited a preference against impossible languages. It posits that this preference will be reflected as a performance deficit on a broad suite of grammatical evaluation tasks.

Hypotheses **H2** and **H3** delve deeper, investigating whether transformers struggle with impossible languages for the same reasons humans do. Human language processing is known to be sensitive to both information locality and structural complexity [25] [31], and research suggests that all the languages used in my study exist on a quantifiable spectrum of naturalness along these lines. When [25] ordering languages by their m-local entropy in ascending order (lower values are better), English was first while their most extreme shuffled language was last. Similarly, in Section 3.3 I ordered these languages using dependency-parse based metrics in order of the proportion of projective sentences (higher values are better), English was first while a shuffled language was last. Therefore, a model with human-like biases should show a corresponding decline in performance as its training language becomes less natural. Conversely, a model that can learn grammatical rules from these highly perturbed languages better than they can learn on English may be leveraging non-human-like strategies. These hypotheses also test the quality of the metrics used as proxies for each bias, probing whether the degree to which model performance is affected can be predicted by them.

A core assumption of my methodology is that the BLiMP tasks themselves do not become consistently easier or harder in the impossible languages. The impossible perturbations used to create these languages are not sensitive to syntax or semantics [15]; they are applied to tokens irrespective of their grammatical role. This implies that for most sentences in a dataset, such perturbations do not make it consistently easier or harder to locate specific parts of it. Since solving the task involves identifying the relevant parts in a sentence and understanding their relationship, this is not fundamentally changed after perturbation. Thus when evaluated on an impossible language-based minimal pair, the same kind of knowledge is being tested as compared to a possible language-based minimal pair, allowing for a fair comparison of a model's ability to generalise that knowledge from different training environments. Moreover, the consistency in training time, compute, and corpus size established by [15] ensure that such cross-model performance comparisons remain meaningful.

3.2 Evaluating Language Learning

In this section, I describe my methodology for constructing 'impossible' BLiMP datasets for various grammatical phenomena. I also present an analysis of these datasets based on the type of knowledge that could be used to 'solve' them.

3.2.1 Impossible BLiMP datasets

To generate 'impossible' versions of the BLiMP datasets, I wrote software that iterated over a standard BLiMP dataset and applied a chosen linguistic perturbation to each sentence in a minimal pair, transforming it into an 'impossible' one. Table 3.1 shows an example of this process applied to a minimal pair from the 'Anaphor Gender Agreement' task for a subset of the impossible languages. Since these datasets had to be produced for each impossible language, I used the same English-based BLiMP dataset to generate canonical impossible BLiMP datasets, with my software enabling parallel application of each impossible perturbation. This ensured that all related impossible BLiMP datasets contained sentences that used the same words as the English-based BLiMP dataset but in different arrangements, allowing me to control for the dataset's inherent lexical and syntactic properties while isolating the effect of each impossible language.

Language	Grammatical Example	Ungrammatical Example
English	Paula cares for **herself**.	Paula cares for **himself**.
Full Reverse	. **herself** R for caresaPaul.	. **himself** R for caresaPaul
Deterministic Shuffle21	Paula **herself** for. cares	Paula **himself** for. cares
Local Shuffle3	caresPaula. for **herself**	caresPaula. for **himself**

Table 3.1: Examples of possible and impossible minimal pairs for the 'Anaphor Gender Agreement' BLiMP task.

Furthermore, I implemented a filtering step to preserve the integrity of the minimal pairs after impossible perturbations. In a minimal pair, the ordering of tokens in both sentences of a pair have to highly consistent except for the intended difference, since sentence probabilities are aggregated based on each token's probability given its preceding context. Inconsistent token ordering could dilute the causal effect intended by the minimal pair design. Therefore, after applying a perturbation to the sentences in a minimal pair, I used the GPT-2 tokeniser to identify and discard any pairs where the two sentences resulted in an unequal number of tokens. This step was essential for ensuring that perturbations like shuffling were applied consistently, as this could only be done with inputs of equal length. Table 3.2 provides an example of this discrepancy. Since the grammatical and ungrammatical English sentences are tokenised into a different number of tokens, their shuffled counterparts are not valid minimal pairs. This is evident from the different positions of the critical tokens between sentences in a pair, highlighted in bold.

Language	Grammatical Example	Ungrammatical Example
English	All teenagers that bother Allison **litter**.	All teenagers that bother Allison **litters**.
Deterministic Shuffle21	All Allison teenagers **litter** bother. that	**lit**All Allison teenagers**ters** bother. that
Local Shuffle3	thatAll teenagers **litter** bother Allison.	thatAll teenagers **lit** bother Allison**ters.**

Table 3.2: Examples of English (valid) and invalid impossible minimal pairs for the ‘Distractor Agreement Relative Clause’ BLiMP task.

Language	**Example Minimal Pair**
English	*Grammatical*: There were **some** coats bothering this man. *Ungrammatical*: There were **all** coats bothering this man.
Full Reverse	*Grammatical*: manR this bothering coats **some** wereThere. *Ungrammatical*: manR this bothering coats **all** wereThere.
Partial Reverse	*Grammatical*: There were **some** coats bothering thisR. man *Ungrammatical*: There were **all** coats bothering thisR. man
Local Shuffle3	*Grammatical*: **some**There were this coats bothering man. *Ungrammatical*: **all**There were this coats bothering man.
Local Shuffle5	*Grammatical*: **some** bothering coatsThere were. this man *Ungrammatical*: **all** bothering coatsThere were. this man
Local Shuffle10	*Grammatical*: **some** bothering coats man thisThere were. *Ungrammatical*: **all** bothering coats man thisThere were.
Even-Odd Shuffle	*Grammatical*: There **some** bothering man were coats this. *Ungrammatical*: There **all** coats bothering man were this.
Deterministic Shuffle21	*Grammatical*: thisThere bothering were man coats. **some** *Ungrammatical*: thisThere bothering were man coats. **all**
Nondeterministic Shuffle	*Grammatical*: coats bothering. **some** this wereThere man *Ungrammatical*: coats bothering. **all** this wereThere man

Table 3.3: Minimal pairs from possible and impossible versions of the ‘Existential There Quantifiers’ BLiMP dataset

Table 3.3 illustrates a sample of minimal pairs from all possible and impossible versions of 'Existential There Quantifiers 1' dataset.

3.2.2 Choice of Grammatical Phenomena

To ensure a broad linguistic evaluation, I selected 15 datasets from the BLiMP benchmark [29]. This selection spans 10 of the 12 major grammatical categories identified in the original work, providing broad coverage for this study. A complete list of the chosen datasets is provided in Table 3.4, with a categorical breakdown illustrated in Table A.2 in the Appendix. Due to time constraints, I excluded datasets from the 'Control/Raising' and 'NPI Licensing' categories. However, the methodology used in this study is extensible and incorporating these categories could be considered in future work.

No.	Grammatical Phenomena	Dataset Name
1	Anaphor Agreement	Anaphor Gender Agreement
2	Anaphor Agreement	Anaphor Number Agreement
3	Argument Structure	Animate Subject Passive
4	Determiner Noun Agreement	Determiner Noun Agreement with Adjective 2
5	Subject Verb Agreement	Distractor Agreement Relative Clause
6	Ellipsis	Ellipsis N-Bar 1
7	Quantifiers	Existential There Quantifiers 1
8	Irregular Forms	Irregular Past Participle Adjective
9	Island Effects	Adjunct Island
10	Island Effects	Left Branch Island Simple Question
11	Binding	Principle A C-Command
12	Filler Gap	WH Question Object Gap
13	Filler Gap	WH Question Object Gap Long Distance
14	Filler Gap	WH Question Subject Gap
15	Filler Gap	WH Question Subject Gap Long Distance

Table 3.4: BLiMP datasets selected for this study.

3.2.3 Classification of BLiMP datasets

The BLiMP datasets selected for this study primarily evaluate knowledge of grammatical phenomena involving hierarchical syntax like ‘Subject Verb Agreement’, ‘Determiner Noun Agreement’ and ‘Anaphor Agreement’. Certain datasets also assess semantic understanding of specific linguistic features, such as ‘Animacy Subject Passive’ which examines whether the model recognises the animacy of nouns. However, as I argued in Section 2.3.1, the construction design of a dataset can also influence the knowledge required to ‘solve’ its corresponding task. One salient feature of a dataset is its favourability to local cues. Strong local cues refer to simple, contextually proximate patterns in the sentences such as word co-occurrences, which can allow a model to predict grammaticality without deep syntactic understanding. On the other hand, weak local cues are ambiguous or unreliable local patterns, which increase the chances that broader, abstract structural knowledge was used to correctly predict grammaticality. I define a property called **Cue Reliability** to assess whether a task can be solved with strong local cues or if it demands an understanding of hierarchical structure because local cues are weak or misleading. Crucially, knowing if a task can be solved with strong local cues or hierarchical structures enables me to probe for the presence of an information locality bias and a hierarchical bias. In order to quantify the cue reliability of a task, I used a model that is fundamentally local in its design: an n-gram model. By definition, n-gram models process language based only on patterns within a local, fixed-size window of $n-1$ words [14], making strong performance on a task an excellent proxy for the availability of strong local cues.

Leveraging this insight, I used the 5-gram model (n-gram model with $n = 5$) performance scores from the BLiMP study [29] to classify each dataset. I selected an accuracy threshold of 80% to consider a task as ‘solved’ based on the results of various models in the BLiMP paper to identify a reasonable threshold of good performance, though the authors themselves did not propose such a cutoff.

The results of this classification are summarized in Table 3.5, with the performance for ’WH Question Object Gap Long Distance’ missing because it was not reported in the BLiMP study though the dataset was present in their code [29]. For most phenomena the classification is clear: tasks where the 5-gram model scores approximately 80% or higher are categorized as having ‘Strong’ cues, while those with lower scores are categorized as ‘Weak.’ Two notable exceptions to this heuristic are ‘Anaphor Gender Agreement’ (44%) and ‘Anaphor Number Agreement’ (52%), both of which I classify as having 'Strong' cues despite their low scores on account of their short average sentence length ($\simeq 4$ words) and simple structure. These discrepancies do not indicate weak linguistic cues but rather highlight a limitation of the 5-gram model itself. In both these tasks, its low accuracies can be attributed to a lack of world knowledge—specifically, the association between proper nouns and gender (e.g., that 'Mary' is female) or number (e.g., that 'Mary' is singular). Acquiring generalisable world knowledge from a limited training set is a challenge for any n-gram model due to its statistical nature, and this reliance on world knowledge for strong performance is a limitation of these datasets acknowledged by the BLiMP authors [29].

Dataset Name	**Cue Reliability**	**n-gram Accuracy (%)**
Adjunct Island	Weak	48
Anaphor Gender Agreement	Strong	44
Anaphor Number Agreement	Strong	52
Animate Subject Passive	Weak	70
Determiner Noun Agreement with Adjective 2	Weak	50
Distractor Agreement Relative Clause	Weak	22
Ellipsis N-Bar 1	Weak	23
Existential There Quantifiers 1	Strong	91
Irregular Past Participle Adjective	Strong	79
Left Branch Island Simple Question	Weak	57
Principle A C-Command	Weak	58
WH Question Object Gap	Weak	53
WH Question Object Gap Long Distance	Weak	–
WH Question Subject Gap	Strong	82
WH Question Subject Gap Long Distance	Strong	86

Table 3.5: Classification of BLiMP Datasets by Cue Reliability and n-gram Accuracy

This classification splits the tasks into those with strong cue reliability which I term **local tasks**, and those with weak reliability which I term **structural tasks**. Tasks like 'WH Question Subject Gap' and 'WH Question Subject Gap Long Distance' which test a complex hierarchical dependency may be expected to require structural knowledge, yet they are local tasks. This is because the subject and the gap are often adjacent despite the long average sentence lengths (9.25 and 13.97 words respectively), contributing to strong cue reliability. The 'Determiner Noun Agreement with Adjective 2' task in contrast tests agreement but has weak cue reliability, possibly due to the presence of adjectives between related words which increases the distance between them. Lastly, the task 'Existential There Quantifiers 1' where related words co-occur closely also exhibits strong cue reliability (examples of all these tasks are displayed in Table A.1 in the Appendix).

3.3 Characterising Impossible Languages

Kallini et al. [15] introduced the impossible languages used in this study with a qualitative spectrum of 'impossibility' (shown in Figure 3.1). However, a more rigorous characterisation is needed to understand the role of specific inductive biases in influencing a language's learnability. To capture the role of a hierarchical bias, I present an analysis of impossible languages based on the dependency parse trees of their sentences. In the following sections, I describe the strategy I used to produce these dependency parse trees, the metrics I measured, and the resulting characterisation.

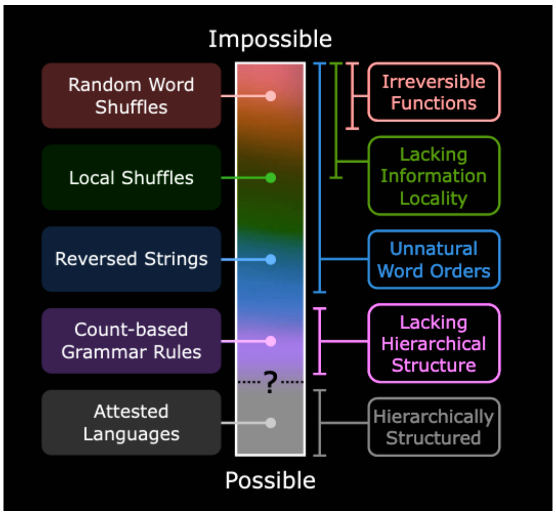


Figure 3.1: Impossibility continuum of languages based on complexity by Kallini et al. [15]

3.3.1 Constructing impossible dependency parse trees

My strategy to construct dependency parse trees for impossible sentences was to first generate a dependency parse tree for a grammatical English sentence, and then apply a perturbation directly to the parse tree. This was possible because the parse tree of a sentence is commonly represented as a list of tokens, each annotated with pointers that link related tokens, which can be perturbed in the same order as the impossible sentence. A key assumption I made here is that the semantic relationships between tokens are preserved post-perturbation, enabling the pointers to stay valid. This assumption also underpins the validity of ’impossible’ BLiMP-style evaluations used in this study.

Figure 3.2 shows an example of a parse tree for the sentence “Timothy didn’t boast about himself”, represented as a list of tokens with pointers represented as arcs. This approach enabled me to leverage existing dependency parsers and I used the one from spaCy, a widely used open-source NLP library [12]. However, this also presented a new technical challenge: spaCy’s tokeniser and the GPT-2 tokeniser, which was used to produce the tokens for the impossible language perturbations, sometimes tokenise words differently. As shown in Figures 3.3 and 3.4, they disagree on how to split words like ‘Timothy’ and ‘didn’t’ when tokenising the example sentence, resulting in a mismatch in token counts and boundaries.

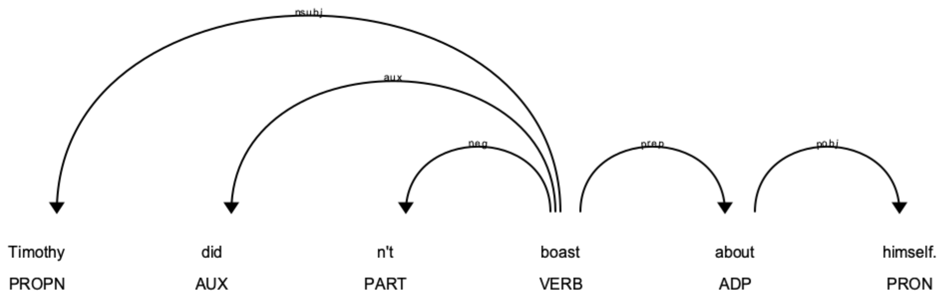


Figure 3.2: Possible dependency parse tree

Timothy, did, n't, boast, about, himself, .,

Figure 3.3: SpaCy tokenisation

Tim, othy, didn, 't, boast, about, himself, .,

Figure 3.4: GPT-2 tokenisation

To resolve this tokeniser mismatch without building a new dependency parser from scratch, I developed an alignment algorithm which relied on mapping spaCy tokens to GPT-2 tokens using character-level alignment. This was also important to enable transferring rich linguistic attributes from spaCy's parse trees (such as part-of-speech tags and dependency types) to the impossible parse trees. When token boundaries aligned, the two tokens in each parse tree were identical. If the GPT-2 tokeniser split a spaCy token into multiple tokens, then the left-most token was be assigned as the head token on account of English being processed left to right. However, this risked the other tokens being isolated in the impossible parse tree, creating the need for a new type of dependency that I called 'LINKED', that linked these tokens to the head token. For example, in Figure 3.4 the spaCy token 'Timothy', which is the 'NSUBJ' dependent of the token 'boast', is split into 'Tim' and 'othy' by the GPT-2 tokeniser. In the impossible dependency parse tree, the token 'othy' would be the 'LINKED' dependent of 'Tim'.

Another scenario of token boundary mismatch is when multiple spaCy tokens map to a single GPT-2 token. In this case, the attributes of the right-most spaCy token were assigned to the GPT-2 token. For example, the 'AUX' and 'NEG' dependencies for 'did' and 'n't' in Figure 3.2 resolve to a single 'NEG' dependency for the token 'didn't' in the final parse shown in Figure 3.6. This choice affects the edge labels but does not impact the structure of the parse tree. The full pseudo-code of this algorithm is presented in the Appendix 1. After aligning and enriching the GPT-2 tokens with the tokens of the possible dependency parse tree, I apply a perturbation to generate the final dependency parse for the corresponding impossible sentence. Figure 3.5 shows the result of applying a `shuffle local3` perturbation to the example sentence, while Figure 3.6 shows it impossible dependency parse tree.

didn, Tim, othy, about, 't, boast, himself, .,

Figure 3.5: Shuffled GPT-2 tokenisation - shuffled in a local window of size 3

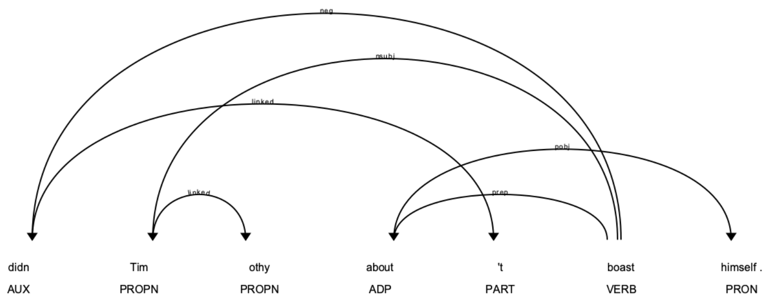


Figure 3.6: Impossible dependency parse tree

3.3.2 Dependency parse-based analysis

The dependency parse-based analysis of possible and impossible languages required corresponding corpora and as I did not have access to the perturbed datasets from [15] and faced time and resource constraints, I created my own at a smaller scale. I sampled 100 grammatical sentences from each of the 15 possible and impossible BLiMP datasets used in this study to create possible and impossible corpora. I then applied my parsing and perturbation algorithm to the sentences in these corpora, producing 100 dependency parse trees for each language variant. Using them, I measured two key metrics: the average total dependency distance (normalised by token count) and the proportion of projective trees. The first metric measured the distance between head and dependent tokens in a parse tree, normalised by the token count within a sentence and averaged across all sentences, and captured how close on average related information was in a sentence of that language. The second metric measured the proportion of projective sentences in a corpus, capturing the probability that a sentence in that language would be projective.

The results of this analysis are summarized in Tables 3.6 and 3.7. These rankings reveals trends in structural complexity, offering a more nuanced view than the qualitative spectrum proposed by [15]. While the various shuffle languages are graded in a similar order, my analysis provides a more precise measure of their relative complexity. The `full reverse` language, as expected by nature of its design as a mirrored transformation, is structurally identical to English. Notably, projectivity decreases drastically with more extreme perturbations, with `even-odd shuffle` having the lowest degree of projectivity, which is plausible given it deterministically splits the sentence.

3.4 Experimental Setup

In this section I outline the experimental setup used in this study. I conducted 135 distinct evaluations, encompassing 15 grammatical phenomenon datasets across nine languages (eight impossible languages and English). For each combination of language model and dataset, I computed the probability of each sentence from each minimal pair under that model, assessed whether the model assigned correct scores, and calculated the overall dataset accuracy. All inferences were performed using SLURM jobs on the University of Edinburgh's Teaching Cluster, leveraging Titan X GPUs to optimise computational efficiency [26][32].

Language	Avg. Norm. Dep. Distance
English	2.04
Full Reverse	2.04
Local Shuffle3	2.38
Partial Reverse	2.61
Local Shuffle5	2.67
Local Shuffle10	3.31
Deterministic Shuffle21	3.48
Even-Odd Shuffle	3.59
Nondeterministic Shuffle	3.65

Table 3.6: Average Normalized Dependency Distance for Different Languages

Language	Proportion Projective (%)
English	92
Full Reverse	92
Partial Reverse	37
Local Shuffle3	25
Deterministic Shuffle21	11
Local Shuffle5	9
Local Shuffle10	5
Nondeterministic Shuffle	5
Even-Odd Shuffle	1

Table 3.7: Proportion Projective for Different Languages

Chapter 4

Results

In the following sections, I present the results of my experiments, highlighting key observations that inform the hypotheses presented in Section 3.1. For brevity, I refer to the performance of the model trained on the English corpus as the baseline performance. A model trained on an impossible language corpus is referred to as the 'impossible language' model (e.g the model trained on `full reverse` is called the `full reverse` model), and all such models are collectively referred to as 'impossible models'. In all evaluations, each model trained on a language was tested on the corresponding version of the BLiMP dataset in that language.

4.1 Model Performance

Figure 4.1 presents the accuracy of all models across the full suite of grammatical tasks, and the performances reveal two key findings. First, as predicted by Hypothesis **H1**, the English model achieves the highest overall accuracy, demonstrating a superior aggregate grasp of the language's linguistic features when compared to any of the impossible models. However, a striking pattern emerges when examining performances on individual tasks. In 12 out of the 15 tasks, one or more of the impossible models outperformed the baseline, and this occurred both in tasks where the baseline was strong (above 80% accuracy) and where it was mediocre (around 60%). This performance suggests that models were able to extract powerful, grammatical phenomena-specific heuristics even from structurally degraded input, sometimes more effectively than from English. Their ability to grasp such patterns even in impossible languages indicates that they may not be suitable analogues for human language learners. In the following sub-sections, I investigate whether information locality and hierarchical biases can explain

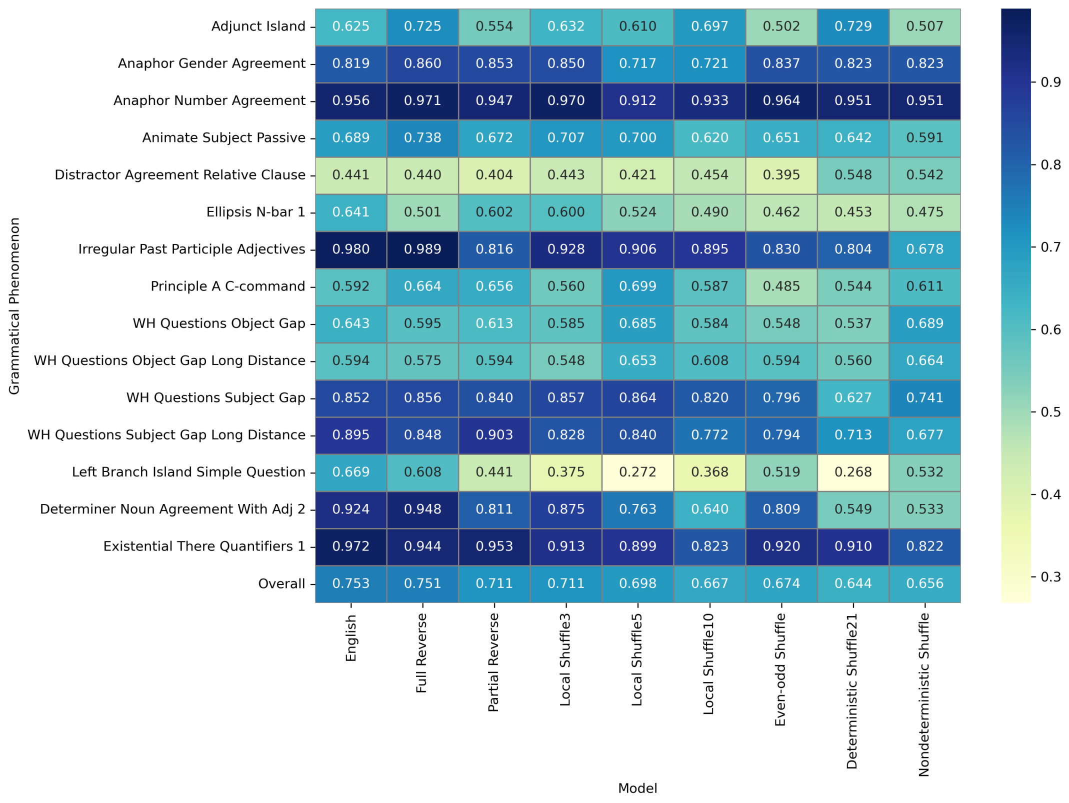


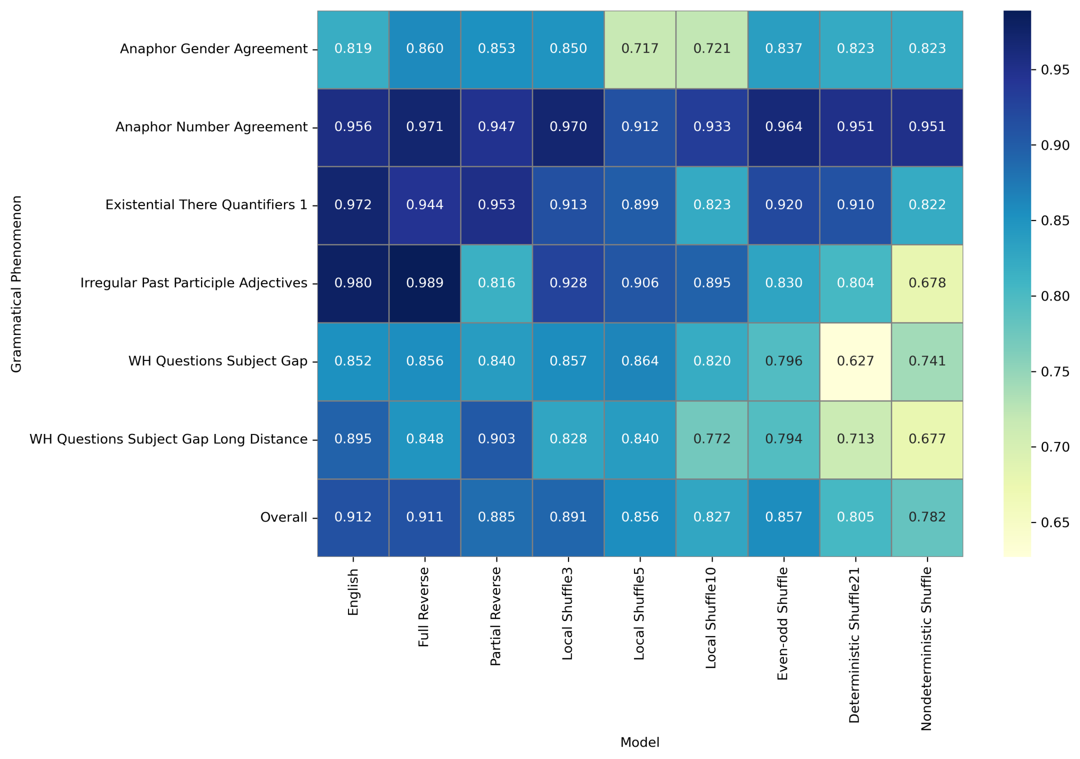
Figure 4.1: Performance of all models on all BLiMP datasets

this behavior by examining the models' performance on **local tasks** and **structural tasks** (as defined in 3.2.2).

4.1.1 Performance on Local Tasks

The central finding for this task group is twofold: all models perform remarkably well, and their performance ranking aligns perfectly with the m-local entropy based ranking of their training language.

As shown in Figure 4.2, nearly all models effectively 'solve' the six local tasks with performances consistently high. On 'Anaphor Gender Agreement' and 'Existential There Quantifiers' tasks most impossible models surpassed 80% accuracy, while results were even stronger for 'Anaphor Number Agreement' where these models achieved more than 90% accuracy. This was also managed by the `nondeterministic shuffle` model with an overall score of nearly 80%, even after being trained on a language with severely degraded hierarchical structure. This shows that on tasks that test hierarchical grammatical phenomena but have strong local cues, models trained on impossible languages can achieve high accuracies.



Grammatical Phenomenon	English	Full Reverse	Partial Reverse	Local Shuffle3	Local Shuffle5	Local Shuffle10	Even-odd Shuffle	Deterministic Shuffle21	Nondeterministic Shuffle
Anaphor Gender Agreement	0.819	0.860	0.853	0.850	0.717	0.721	0.837	0.823	0.823
Anaphor Number Agreement	0.956	0.971	0.947	0.970	0.912	0.933	0.964	0.951	0.951
Existential There Quantifiers 1	0.972	0.944	0.953	0.913	0.899	0.823	0.920	0.910	0.822
Irregular Past Participle Adjectives	0.980	0.989	0.816	0.928	0.906	0.895	0.830	0.804	0.678
WH Questions Subject Gap	0.852	0.856	0.840	0.857	0.864	0.820	0.796	0.627	0.741
WH Questions Subject Gap Long Distance	0.895	0.848	0.903	0.828	0.840	0.772	0.794	0.713	0.677
Overall	0.912	0.911	0.885	0.891	0.856	0.827	0.857	0.805	0.782

Figure 4.2: Performance of all models on local tasks

In Figure 4.3, I compare the overall accuracy-based performance ranking against the m-local entropy ranking of the training languages [25]. The strongest m-local entropy (when $m = 4$) ranking was chosen for this comparison, while matching languages by name and design (e.g., my `full reverse` language corresponds to their `reverse` language). The comparison reveals that the two rankings are identical. A model's relative ability to solve these local tasks perfectly correlates with the degree of local information disruption in its training data. Although [25] did not test all the same impossible languages, the trend is unambiguous where our studies overlap. Moreover, this performance ordering is still highly consistent at the level of individual tasks. Within the Reverse family of languages, the `full reverse` model always outperforms the `reverse-partial` model, while in the Shuffle family of languages, performance predictably degrades as the shuffling window widens (the `local shuffle3` model outperforms the `local shuffle5` model, which outperforms the `local shuffle10` model). Lastly, the `deterministic shuffle21` model outperforms the `nondeterministic shuffle` model in all but one task, which could be due to idiosyncrasies of that specific task and those languages. This consistent performance ordering and the overall high accuracy on these tasks provide strong evidence for an information locality bias, supporting Hypothesis **H2**.

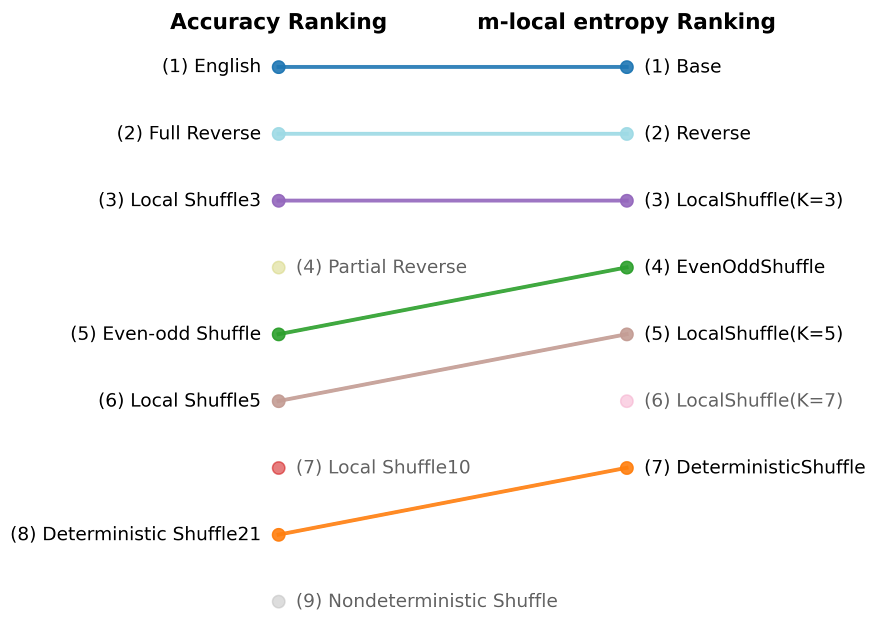


Figure 4.3: Consistency between rankings based on performance on local tasks and m-local entropy

4.1.2 Performance on Structural Tasks

In a stark contrast to their 'success' on local tasks, all models performed poorly on tasks requiring hierarchical knowledge, while displaying no consistent ranking of performance across them.

As shown in Figure 4.4, the average accuracy of all models on these nine structural tasks dropped by nearly 30% when compared to their performance on local tasks. On many individual tasks, the performance of the impossible models regressed toward chance (50%). Notably, the two models with lowest m-local entropy (`full reverse` model and the baseline) outperformed all the other impossible models. They performed on par overall but had varying performance orderings at the individual level (e.g the `full reverse` model outperformed the baseline in the 'Adjunct Island' task, but was outperformed in the 'Left Branch Island Simple Question' task).

Unlike the consistent ranking observed for local tasks, the ranking based on overall performances in structural tasks is not reflected at the individual task level, making any comparisons to the rankings of metrics intractable. For instance, the performance ordering reverses unpredictably both within families (e.g. the `local shuffle10` model both over-performs and under-performs the `local shuffle3` model on different tasks) and across them (e.g. the `even-odd shuffle` model both over-performs and under-

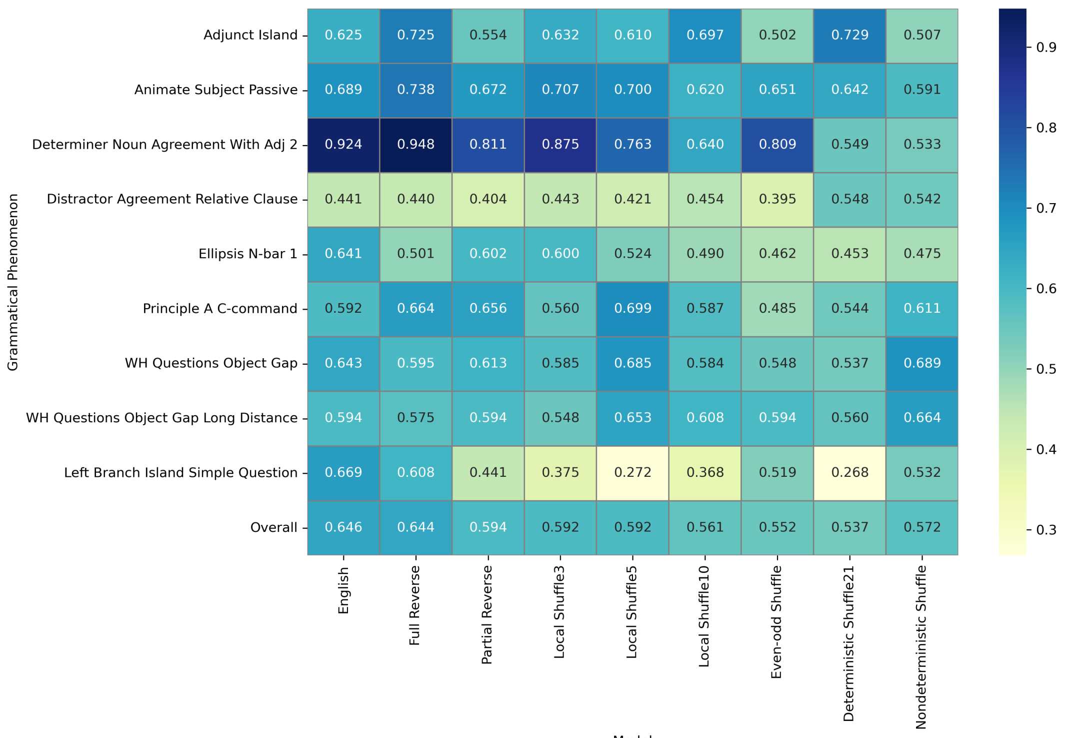


Figure 4.4: Performance of all models on structural phenomena

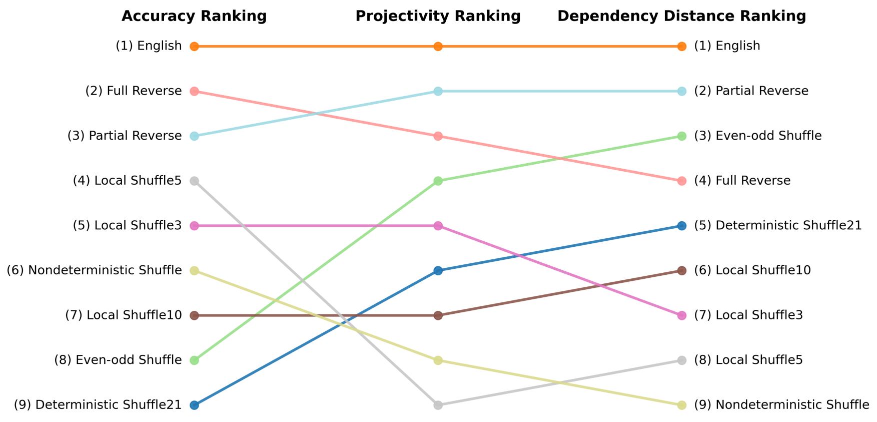


Figure 4.5: Consistency between rankings based on performance on structural tasks and dependency parse-based metrics

performs the `full reverse` model on different tasks). Nevertheless, in Figure 4.5, I compare the models' overall performance ranking against two hierarchical metrics of a language: the proportion of projectivity and average normalised dependency distance (from Section 3.3). This comparison underscores the lack of correlation between any of the two rankings. This lack of a consistent performance gradient, coupled with the poor overall accuracy, is evidence against the existence of a hierarchical bias in these models, addressing Hypothesis **H3**.

4.1.3 Comparison to BLiMP results

To contextualise the performance of the models, I compare them with the performances reported in the BLiMP study [29]. In Figure 4.6, I compare the performance of the GPT-2 model used in this study to theirs on overlapping BLiMP datasets, examining trends to validate my methodology. A strong correlation would suggest that the fundamental linguistic competencies required for these tasks are being tested through similar mechanisms in both cases, providing confidence in the validity of my experimental approach. This comparison however has a key limitation: the models in this study were trained on a considerably smaller dataset than the GPT-2 model reported in their work [29]. Nevertheless, the performances show a strong positive linear correlation (Pearson's r = 0.833, $p < 0.001$ [5]), though the BLiMP model achieves consistently higher absolute scores likely due to its larger training dataset size.

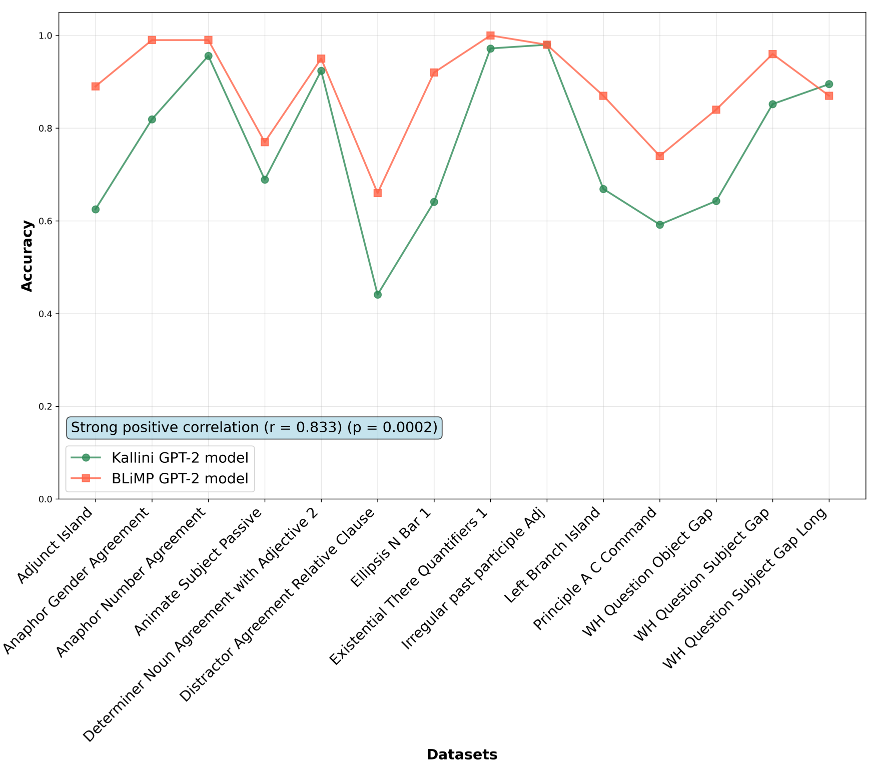


Figure 4.6: Performance comparison between Kallini's GPT-2 model and BLiMP's GPT-2 model on common BLiMP datasets

In Figure 4.7, I compare the performances of all models with the scores of humans (as reported in the BLiMP study), grouped together by grammatical phenomena. This

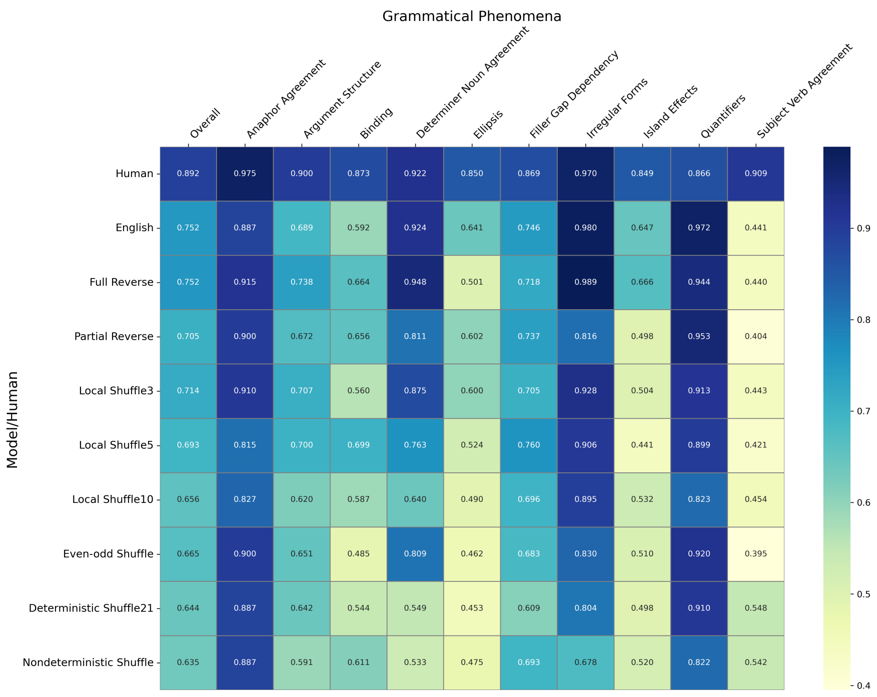


Figure 4.7: Comparison of model performance with human scores across grammatical phenomena on BLiMP tasks

surfaces broad comparable trends, though the human scores are aggregated over 67 datasets and those of these models are over 15. Nevertheless, some grammatical categories in my study contain 2 or more datasets (as shown in Figure A.2), making them more representative. Among these categories, the impossible models perform close to humans on 'Anaphor Agreement' and the `full reverse` model outperforms humans on 'Irregular Forms', underscoring their ability to learn effectively even when trained on impossible languages. In their study [29] concluded that transformer models generally performed better on morphological phenomena like 'Anaphor Agreement' and this is reflected in my results. My analysis of the tasks suggests one possible reason for this: such tasks tend to contain stronger local cues which the models are biased towards. [29] also suggested that language models represented long-distance dependencies in ways different from humans. My results support this finding by exhibiting model reliance on local cues to solve hierarchical tasks, indicating that such representations might not be rooted in a hierarchical bias.

Chapter 5

Discussion

My results provide linguistically grounded evidence that GPT-2 based transformer models learn impossible languages worse than possible languages, supporting the findings of prior work. They exhibit the strongest overall grasp on a variety of grammatical phenomena when trained on English. My results also provide evidence that these models exhibit an information locality bias but do not exhibit a hierarchical bias. Their performances across the languages indicate that they relied on local information better than hierarchical knowledge to solve BLiMP tasks.

The models performed well on local tasks but struggled on structural tasks, showing strong performances on BLiMP tasks with strong local cues. Moreover, performance on local tasks degraded reliably as the m-local entropy of the training language increased. This shows that the model did better on tasks which had strong local cues and when trained on languages with strong local cues, hinting at a deeper architectural bias. The usage of local cues to solve tasks in particular hints that while training, the models are learning to recognise specific kinds of patterns in the training data.

The performance on local and structural tasks could thus be reinterpreted as performances on BLiMP tasks that were designed in a manner that aligned with patterns from their training data and tasks that didn't. The prevalence of different types of patterns in the training data can thus directly influence the performance on different tasks. Moreover, the authors of the BLiMP study also found that the performance of models was heavily influenced by the size and quality of their training dataset [29]. This is further evidenced by the fact that for the same training dataset, the models in my study were able to grasp certain patterns better than others. For example, among all the structural tasks, baseline performance on 'Determiner Noun Agreement with Adjective 2' was uncharacteristically strong (above 90%). A likely reason is that the 'Determiner

Noun Agreement' phenomena is one of the most commonly occurring phenomena in English, and such patterns are bound to be rife in an English corpus [29]. In this context, an information locality bias is helpful to extract useful patterns since many occur locally in sentences.

Fundamentally however, my results also indicate transformer models don't exhibit a hierarchical bias which differentiates them from humans. Based on my results, I take the position that their main inductive bias is to find the path of least resistance to minimise prediction error during training, and when learning languages that path is paved with statistical based pattern matching. This still enables models to learn hierarchical rules as long as factors like the training time or dataset size favour it, an argument also proposed by [23]. Nevertheless, while they may not be complete models of human language learning, they have been shown to share some characteristics, cautioning future studies about making broad conclusions of human language learning when basing their findings on results from transformer models.

5.1 Limitations

A key limitation of this study is the limited assessment of statistical robustness of the results; replicating the analyses across multiple models and evaluation sets would strengthen the evidence. While [15] trained 5 models with varying seeds, only one set was released on HuggingFace [13] and this prevented me from using them in this study. Another limitation was only using a subset of the BLiMP datasets due to time and resource constraints. More broadly, the reliance on only one type of evaluation can also be limiting in nature. An alternative to consider in future work could the 'Question Formation' task used to test for hierarchical biases in a study by Murty et al. [21]. My study was also conducted only on English and its perturbed versions which limits the confidence of extending these results beyond them. Validating the results in other languages is a natural extension, preferably on those that have less strict word order and hence could be more robust to shuffling perturbations (e.g agglutinative languages like Turkish).

With respect to the extension of these results to large language models (LLMs), it is important to call out the limits of extrapolation with the primary factory being scale [22]. The models used in this study were trained on datasets that are orders of magnitude smaller than the state-of-the-art LLMs, while their architectures have been succeeded with more sophisticated versions [22]. This makes extrapolating these results to LLMs awkward, but empirical verification of similar behaviour remains scope for future work.

Another limitation of this study is that the metrics reported and compared were measured on different corpora due to resource access constraints. For instance, all the models were trained on one set of corpora, while m-local entropy values were measured on a different set of corpora, and my dependency parse-based metrics were calculated on a third set. Though they all used the same base language (English) and perturbation design, lexical variance can be controlled to make stronger conclusions.

5.2 Future Work

An interesting result in this study is the strong performance of the `nondeterministic shuffle` on many tasks. This model was trained on a language which essentially makes each sentence of the training corpus a bag-of-words by shuffling them [14][15], and yet it was capable of extracting many salient features of language. One reason could be that the nondeterministic shuffling of language acts as a regulariser, forcing this model to represent such grammatical phenomena in a more abstract manner. Understanding the internal techniques used by such models, and checking if they are similar to a model trained on English, would be useful to illustrate the robustness of their learning mechanisms models. This could motivate a mechanistic interpretation study using interventions methods like the ones illustrated in [2], to examine the specific computational strategies used by these models when solving tasks where it performs better than the baseline. One benefit of this kind of study is better regulariser signals useful for training, an idea also suggested by [25].

My analysis also showcased the degree of degradation of both local information and hierarchical structures in the impossible languages, but it did not probe the question of whether this relationship was causative or correlative. An approach to tease these two factors apart would be to control one while varying the other. One example of doing this during evaluation would be to replace key words with nonce words in tasks that the model is performing well on (e.g local tasks). Nonce words are non-dictionary words usually coined for a single use while filling a particular lexical role (e.g the noun-like word 'Jabberwocky') [6]. As these words are very unlikely to occur in the training dataset of these models, it disrupts the local information but retains the hierarchical structure, allowing for a stronger test of a model's knowledge. Controlling these factors may also be possible at training time by using carefully designed languages, though this design is left as future work.

Chapter 6

Conclusion

In this study, I aimed to examine linguistic evidence for the learning preference of transformer-based language models toward possible, human-like languages over impossible languages. I also aimed to investigate if they were exhibiting such a preference for the same reasons humans were, by probing for two inductive biases humans have shown to exhibit: information locality bias and hierarchical bias. I constructed BLiMP-style evaluations that tested the linguistic knowledge of models and my results showed that while these models exhibited a preference toward possible language, they were not doing so for the same reasons as humans. The models displayed an information locality bias but not a hierarchical bias, highlighting that they are not analogous models of human language learning. My results supported the findings of existing work while surfacing new salient evidence. My methodology highlighted how BLiMP-style evaluations can be used to provide relevant evidence when probing the language learning mechanisms of the models.

Bibliography

[1] Kabir Ahuja, Vidhisha Balachandran, Madhur Panwar, Tianxing He, Noah A. Smith, Navin Goyal, and Yulia Tsvetkov. Learning Syntax Without Planting Trees: Understanding Hierarchical Generalization in Transformers. *Transactions of the Association for Computational Linguistics*, 13:121–141, February 2025.

[2] Aryaman Arora, Dan Jurafsky, and Christopher Potts. CausalGym: Benchmarking causal interpretability methods on linguistic tasks, February 2024. arXiv:2402.12560 [cs].

[3] N. Chomsky. *Aspects of the Theory of Syntax*. MIT Press, 1969.

[4] Noam Chomsky. Three Factors in Language Design. *Linguistic Inquiry*, 36(1):1–22, January 2005.

[5] Jacob Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, Hillsdale, NJ, 2nd edition, 1988.

[6] A. Ross Eckler. Words, non-words, nonce words. *Word Ways*, 4(2), 1971.

[7] Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 478(2266):20210068, October 2022.

[8] Carlos Gómez-Rodríguez, Morten H. Christiansen, and Ramon Ferrer-i Cancho. Memory limitations are hidden in grammar. *Glottometrics*, 52:39–64, 2022. arXiv:1908.06629 [cs].

[9] Akari Haga, Akiyo Fukatsu, Miyu Oba, Arianna Bisazza, and Yohei Oseki. BabyLM Challenge: Exploring the Effect of Variation Sets on Language Model Training Efficiency, March 2025. arXiv:2411.09587 [cs].

[10] Michael Hahn, Richard Futrell, Roger Levy, and Edward Gibson. A resource-rational model of human processing of recursive linguistic structure. *Proceedings of the National Academy of Sciences*, 119(43):e2122602119, 2022.

[11] Ken Hale and Samuel Jay Keyser. On argument structure and the lexical expression of syntactic relations. 1993.

[12] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength Natural Language Processing in Python. 2020.

[13] HuggingFace. Huggingface. https://huggingface.co, 2025.

[14] Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, Upper Saddle River, N.J., 2009.

[15] Julie Kallini, Isabel Papadimitriou, Richard Futrell, Kyle Mahowald, and Christopher Potts. Mission: Impossible Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14691–14714, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[16] Sandra Kübler, Ryan McDonald, and Joakim Nivre. Dependency parsing. In *Dependency parsing*, pages 11–20. Springer, 2009.

[17] Shigeru Miyagawa. *Why Agree? Why Move?: Unifying Agreement-Based and Discourse-Configurational Languages*. The MIT Press, October 2009.

[18] Shigeru Miyagawa, Robert C. Berwick, and Kazuo Okanoya. The Emergence of Hierarchical Structure in Human Language. *Frontiers in Psychology*, 4, 2013.

[19] A. Moro. *Impossible Languages*. MIT Press, 2023.

[20] Aaron Mueller, Robert Frank, Tal Linzen, Luheng Wang, and Sebastian Schuster. Coloring the Blank Slate: Pre-training Imparts a Hierarchical Inductive Bias to Sequence-to-sequence Models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1352–1368, Dublin, Ireland, 2022. Association for Computational Linguistics.

[21] Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D. Manning. Grokking of Hierarchical Structure in Vanilla Transformers, May 2023. arXiv:2305.18741 [cs].

[22] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models. *ACM Trans. Intell. Syst. Technol.*, 16(5), August 2025.

[23] Jackson Petty and Robert Frank. Transformers Generalize Linearly, September 2021. arXiv:2109.12036 [cs].

[24] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI*, 2019. Accessed: 2024-11-15.

[25] Taiga Someya, Anej Svete, Brian DuSell, Timothy J. O'Donnell, Mario Giulianelli, and Ryan Cotterell. Information Locality as an Inductive Bias for Neural Language Models, June 2025. arXiv:2506.05136 [cs].

[26] University of Edinburgh, School of Informatics. Teaching cluster: Gpu computing resources, 2025. Accessed: 2025-08-19.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, August 2023. arXiv:1706.03762 [cs].

[28] Alex Warstadt, Leshem Choshen, Aaron Mueller, Adina Williams, Ethan Wilcox, and Chengxu Zhuang. Call for Papers – The BabyLM Challenge: Sample-efficient pretraining on a developmentally plausible corpus, January 2023. arXiv:2301.11796 [cs].

[29] 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, February 2023. arXiv:1912.00582 [cs].

[30] Tianyang Xu, Tatsuki Kuribayashi, Yohei Oseki, Ryan Cotterell, and Alex Warstadt. Can Language Models Learn Typologically Implausible Languages?, February 2025. arXiv:2502.12317 [cs].

[31] Himanshu Yadav, Samar Husain, and Richard Futrell. Assessing Corpus Evidence for Formal and Psycholinguistic Constraints on Nonprojectivity. *Computational Linguistics*, 48(2):375–401, June 2022.

[32] Andy B. Yoo, Morris A. Jette, and Mark Grondona. SLURM: Simple Linux Utility for Resource Management. In Gerhard Goos, Juris Hartmanis, Jan Van Leeuwen, Dror Feitelson, Larry Rudolph, and Uwe Schwiegelshohn, editors, *Job Scheduling Strategies for Parallel Processing*, volume 2862, pages 44–60. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003. Series Title: Lecture Notes in Computer Science.

Appendix A

Tables & Figures

Grammatical Phenomena	Dataset Name	Grammatical Example	Ungrammatical Example
Anaphor Agreement	Anaphor Gender Agreement	Paula cares for **herself**.	Paula cares for **himself**.
Anaphor Agreement	Anaphor Number Agreement	Todd is admiring **himself**.	Todd is admiring **themselves**.
Argument Structure	Animate Subject Passive	The soda was drunk by some **child.**	The soda was drunk by some **diagnosis**.
Determiner Noun Agreement	Determiner Noun Agreement with Adjective	Gerald could examine **that** long essay.	Gerald could examine **those** long essay.
Ellipsis	Ellipsis N-Bar 1	Curtis bikes to one **important** glacier and Kevin bikes to two.	Curtis bikes to one glacier and Kevin bikes to two **important**.
Irregular Forms	Irregular Past Participle Adjective	The **hidden** coffee wasn't hot.	The **hid** coffee wasn't hot.
Quantifiers	Existential There Quantifiers 1	There was **a** picture confusing customers.	There was **each** picture confusing customers.
Binding	Principle A C-command	The children that were complaining about Gregory hated **themselves**.	The children that were complaining about Gregory hated **himself**.
Island Effects	Adjunct island	Who was Paula fighting **before** talking to Sonia?	Who was Paula fighting **Sonia** before talking to?
Filler Gap	WH-Questions Subject Gap	Sabrina is researching a story **that** confused Mark	Sabrina is researching **who** a story confused Mark.
Filler Gap	WH-Questions Subject Gap Long Distance	Renee investigates these offspring that some drivers boast about **that** would heal some dog.	Renee investigates **what** these offspring that some drivers boast about would heal some dog.
Filler Gap	WH-Questions Object Gap	A driver has thought about **those** deer that many pictures look like.	A driver has thought about **what** many pictures look like those deer.
Filler Gap	WH-Questions Object Gap Long Distance	Ella saw all oxen **that** some waitresses that are playing heal.	Ella saw **what** some waitresses that are playing heal all oxen.
Subject Verb Agreement	Distractor Agreement Relative Clause	A customer that loved waiters **urges** Curtis to murmur.	A customer that loved waiters **urge** Curtis to murmur.
Island Effects	Left Branch Island Simple Question	What **movies** is Renee concealing?	What is Renee concealing **movies**?

Table A.1: Examples of minimal pairs for all selected BLiMP datasets used in this study

Grammatical Phenomena	Count
Anaphor Agreement	2
Argument Structure	1
Determiner Noun Agreement	1
Ellipsis	1
Irregular Forms	1
Quantifiers	1
Island Effects	2
Filler Gap	4
Subject Verb Agreement	1
Binding	1
Control Raising	0
NPI Licensing	0

Table A.2: Count of impossible BLiMP datasets grouped by phenomena.

Appendix B

Algorithm

Algorithm 1 Align spaCy Tokens with a Generic Tokenizer - Part 1

```
 1: procedure AlignTokens($sentence, original\_doc, tokenizer$)
 2:     $tokenizer\_ids \leftarrow$ TOKENIZER.ENCODE($sentence$)
 3:     $tokenizer\_tokens \leftarrow$ TOKENIZER.DECODE_EACH($tokenizer\_ids$)
 4:     $spacy\_tokens \leftarrow original\_doc$
    # 1. Create a map from character index to spaCy token index
 5:     $char\_to\_spacy\_map \leftarrow \{\}$
 6:     for $i \leftarrow 0$ to length($spacy\_tokens$) - 1 do
 7:         $token \leftarrow spacy\_tokens[i]$
 8:         for $char\_idx \leftarrow token.idx$ to $token.idx +$ length($token.text$) $- 1$ do
 9:             $char\_to\_spacy\_map[char\_idx] \leftarrow i$
    # 2. Align tokenizer tokens to spaCy tokens based on character spans
10:     $tokenizer\_to\_spacy\_mapping \leftarrow []$
11:     $current\_char\_pos \leftarrow 0$
12:     for each $tok\_token$ in $tokenizer\_tokens$ do
13:         $token\_start \leftarrow$ FIND_SUBSTRING_POS($sentence, tok\_token, current\_char\_pos$)
14:         if $token\_start = -1$ then
15:             Append $None$ to $tokenizer\_to\_spacy\_mapping$
16:             continue
17:         $token\_end \leftarrow token\_start +$ length($tok\_token$)
18:         $spacy\_tokens\_covered \leftarrow$ new set()
19:         for $char\_idx \leftarrow token\_start$ to $token\_end - 1$ do
20:             if $char\_idx$ in $char\_to\_spacy\_map$ then
21:                 Add $char\_to\_spacy\_map[char\_idx]$ to $spacy\_tokens\_covered$
22:         if $spacy\_tokens\_covered$ is not empty then
23:             $primary\_spacy\_token \leftarrow$ min($spacy\_tokens\_covered$)
24:             Append ($primary\_spacy\_token, spacy\_tokens\_covered$) to
    $tokenizer\_to\_spacy\_mapping$
25:         else
26:             Append $None$ to $tokenizer\_to\_spacy\_mapping$
27:         $current\_char\_pos \leftarrow token\_end$
```

Algorithm 2 Align spaCy Tokens with a Generic Tokenizer - Part 2

```
    # 3. Create mapping from old spaCy index to new primary tokenizer index
 1: spacy_to_new_token_map ← {}
 2: new_token_groups ← {}    ▷ Groups of new tokens from one original spaCy token
 3: for new_idx ← 0 to length(tokenizer_to_spacy_mapping) - 1 do
 4:     alignment ← tokenizer_to_spacy_mapping[new_idx]
 5:     if alignment is not None then
 6:         primary_spacy_idx, spacy_tokens_covered ← alignment
 7:         if primary_spacy_idx not in spacy_to_new_token_map then
 8:             spacy_to_new_token_map[primary_spacy_idx] ← new_idx
 9:         for each spacy_idx in spacy_tokens_covered do
10:             Append new_idx to new_token_groups[spacy_idx]
    # 4. Initialize new token list with basic info and copied spaCy attributes
11: aligned_tokens ← []
12: for new_idx ← 0 to length(tokenizer_tokens) - 1 do
13:     Create token_dict with index, text, and default head (self) and dependency
    (ROOT)
14:     if tokenizer_to_spacy_mapping[new_idx] is not None then
15:         primary_spacy_idx, _ ← tokenizer_to_spacy_mapping[new_idx]
16:         original_token ← spacy_tokens[primary_spacy_idx]
17:         Copy POS, tag, lemma, etc. from original_token to token_dict
18:     Append token_dict to aligned_tokens
```

Algorithm 3 Align spaCy Tokens with a Generic Tokenizer - Part 3

```
    # 5. Update dependency heads for the new tokens
    # First pass: Assign primary dependency heads
 1: for $old\_idx \leftarrow 0$ to length($spacy\_tokens$) - 1 do
 2:     if $old\_idx$ in $spacy\_to\_new\_token\_map$ then
 3:         $new\_idx \leftarrow spacy\_to\_new\_token\_map[old\_idx]$
 4:         $old\_token \leftarrow spacy\_tokens[old\_idx]$
 5:         $old\_head\_idx \leftarrow old\_token.head.i$
 6:         if $old\_head\_idx$ in $spacy\_to\_new\_token\_map$ then
 7:             $new\_head\_idx \leftarrow spacy\_to\_new\_token\_map[old\_head\_idx]$
 8:             $aligned\_tokens[new\_idx].head\_index \leftarrow new\_head\_idx$
 9:             $aligned\_tokens[new\_idx].dep \leftarrow old\_token.dep$
    # Second pass: Link sub-tokens within a split spaCy token
10: for each $spacy\_idx, new\_indices$ in $new\_token\_groups$ do
11:     if length($new\_indices$) ¿ 1 then
12:         Sort $new\_indices$
13:         $leftmost\_new\_idx \leftarrow new\_indices[0]$
14:         for $i \leftarrow 1$ to length($new\_indices$) - 1 do
15:             $current\_new\_idx \leftarrow new\_indices[i]$
16:             $aligned\_tokens[current\_new\_idx].head\_index \leftarrow leftmost\_new\_idx$
17:             $aligned\_tokens[current\_new\_idx].dep \leftarrow$ 'linked'
18: return $aligned\_tokens$
```