Reproducibility Report for AANN

Yuchen Xin, Yash Mathur, Hyeonjeong Byeon
University of Washington
{yuchxin, yashmat, hjbyeon}@uw.edu

Reproducibility Summary

Motivation

The paper's study is motivated by the phenomenon that humans can learn and utilize rare grammatical structures, even when they have encountered them infrequently or not at all.

Scope of Reproducibility

The paper we are reproducing is "Language Models Learn Rare Phenomena from Less Rare Phenomena: The Case of the Missing AANNs" [5]. [1] The main hypothesis is that LMs can learn rare grammatical phenomena by generalizing from less rare phenomena. The particular phenomena tested was the AANN.

Methodology

Using the BabyLM 100M corpus, we trained transformer-based LMs (a LLaMA variant) under each condition. We evaluate models by comparing SLOR scores computed for well-formed AANNs against four corrupted variants. Accuracy is defined as the proportion of test cases where the valid AANN achieves the highest SLOR among five candidates. We also compared different tokenization schemes by using the pretrained `albert-base-v2` tokenizer versus a custom BPE tokenizer.

Results

Overall, our study was successful. We reproduced the general trend of results reported in the original paper. All models performed well above the 20% chance baseline. The model trained after removing AANNs performed significantly better than chance. Models trained after further removal of related constructions were still able to correctly predict AANN's in the vast majority of cases, supporting the hypothesis that LMs rely on related patterns for generalization. Models with the `albert-base-v2` tokenizer showed higher accuracy than those with the custom BPE tokenizer. Surprisingly, POS-tag augmentation did not improve performance.

What was Easy

While a large portion of the paper was complex, having access to Hugging Face libraries made it easy to set up the dataset, so the training itself was not too difficult to reproduce. Additionally, with frequent communication with the author of the paper, we were able to resolve many errors, especially regarding the ablated datasets.

What was Difficult

The biggest difficulty was the lack of computing power. The original paper had far more parameters and number of training epochs than the resources we could afford, so we could had to train models with reduced hyperparameters and number of epochs. We also had to use the Llama 2 architecture instead of the OPT architecture the authors used.

Communication with Original Authors

We communicated with the authors via email to get assistance with running scripts to generate the dataset.

[1] https://aclanthology.org/2024.emnlp-main.53/

1 Introduction

The paper we are reproducing is "Language Models Learn Rare Phenomena from Less Rare Phenomena: The Case of the Missing AANNs" [5] by Dr. Kanishka Misra and Dr. Kyle Mahowald.

The paper's study is motivated by the phenomenon that humans can learn and utilize rare grammatical structures, even when they have encountered them infrequently or not at all. The authors focus on a specific rare grammatical structure, Article + Adjective + Numeral + Noun (AANN) (e.g., "a beautiful five days"), and investigate whether language models trained on a human-size corpus can learn to generate AANNs (which ablated from the training data) correctly through generalization from other, more frequent phenomena in the corpus. Findings demonstrate that language models can learn these rare structures far better than chance without direct exposure. Additionally, the paper identifies key related grammatical phenomena that the language model learned to generalize from.

The paper's hypothesis is "generalization abilities of LMs on such rare phenomena come from abstractions and structures learned from more frequent related phenomena". We will replicate an AANN-ablation experiment and a related-phenomena-ablation experiment from the paper as evidence that LMs can indeed learn via generalization of related phenomena. For our additional experiment, we will train the same LM after adding POS tags to the original data, to test whether LMs can learn to generalize from related morphological structure.

1.1 Related Work

Prior research in both cognitive and computational linguistics has demonstrated that humans can learn and use rare grammatical constructions even with minimal exposure. This observation challenges theories that rely exclusively on innate, hard-wired linguistic knowledge and suggests instead that generalization from more frequent constructions may play a crucial role in language acquisition. Recent studies have shown that LLMs are not only capable of learning complex grammatical patterns from human-scale datasets [4], but they can also generalize to infrequent phenomena by leveraging underlying abstract structures rather than simply memorizing surface patterns [6].

For example, work on subject-verb agreement, negative polarity items, and passive constructions has illustrated how systematically manipulated corpora can be used to isolate and test the causal effects of specific linguistic features on model performance [3]. These studies highlight a continuum between memorization of fixed phrases and the abstraction of more flexible, rule-governed patterns. In a similar vein, our study investigates the learning of rare constructions—specifically, the AANN pattern—as an instance of the "long tail" of linguistic phenomena.

Our approach extends the current body of work by explicitly testing whether LLMs can predict unseen rare structures solely through exposure to related, more common constructions. By using targeted ablations and comparing models trained on systematically altered datasets, we not only replicate previous findings but also probe the extent to which linguistic generalization in LLMs mirrors human language processing.

2 Scope of Reproducibility

This paper examines how the generalization abilities of language models (LMs) on rare phenomena, such as AANNs, stem from abstractions and structures learned from more frequent, related phenomena. To investigate this, we will train and compare different language models trained on different ablations of the same underlying dataset: (1) the full corpus of text, (2) the same corpus with AANNs removed, and (3) the same corpus with both AANNs and the related indefinite article + measure word + plural noun (Indef NNS) combination removed. Examples of the Indef NNS combination include "a few days" or "a couple bottles", which is similar to an AANN without an adjective.

Then, we will compare the accuracy of these 3 models. Model (2) should perform worse than the unablated model (1), but since we hypothesize that generalization comes from related phenomena, model (3) should have even worse accuracy than model (2).

We will also conduct an additional experiment by training language models on the full corpus of text, and on the no AANN ablation, but with all words having parts-of-speech (POS) tagging. Since the AANN is a syntactic construction defined almost purely by the parts of speech of each word, we expect the models trained with POS-tagging to have much higher accuracy than any other model.

To test our models and determine accuracy, we use the method in described in the paper. For each prefix + well-formed AANN combination in the test data, 4 corresponding corrupted versions of the AANN part are generated: (1) adjective-numeral swapped, (2) no article, (3) no adjective, and (4) no numeral. For example, a well-formed "a whopping ninety LMs" would be corrupted into (1) "a ninety whopping LMs", (2) "whopping ninety LMs", (3) "a ninety LMs", and (4) "a whopping LMs", and each version would follow the same prefix as the original well-formed AANN.

Then, given the prefix, we compare the probabilities of the LM generating the well-formed version versus its corrupted versions. In particular, we normalize each probability using the Syntactic Log-odds Ratio: given an LM m and a unigram estimator u (trained on the same data as m), the probability of generating C given a prefix is normalized to the SLOR score

$$\text{SLOR}(C, \text{prefix}) = \frac{1}{|C|} \log \frac{p_m(C \mid \text{prefix})}{p_u(C)} = \frac{1}{|C|} \log \frac{\prod_{i=1}^{n} P_m(c_i \mid \text{prefix}, c_{1:i-1})}{\prod_{i=1}^{n} P_u(c_i)}$$

where we expanded the definition using the chain rule. An LM m is correct for a prefix + AANN test instance iff the SLOR score of the well-formed version is greater than the SLOR scores of all 4 corrupted versions. Accuracy is then measured as the ratio of correct to incorrect instances out of the entire test set.

By chance, the LM will assign the highest SLOR score for the well-formed version over all 4 corrupted versions with a probability of $\frac{1}{5} = 20\%$. This will be our baseline to compare the accuracies our models achieve in our experiments.

2.1 Addressed Claims from the Original Paper

The original paper makes two key claims regarding the language model's capabilities which we will attempt to replicate in this paper:

1. It asserts that the model demonstrates statistically significant accuracy in predicting rare phenomena that were not present in the training data.
2. It argues that the model's ability to generalize to such rare phenomena, including AANNs, stems from abstractions and structures it has learned from more frequently occurring related phenomena.

3 Methodology

As stated in the Scope of Reproducibility section, we will train three models on (1) the full corpus of text, (2) the same corpus with AANNs removed, and (3) the same corpus with both AANNs and the measure noun phrase + NNS phenomena removed. The regex expressions to prune through the BabyLM dataset are all provided in the paper, and the training code is all open sourced. Then, using the SLOR scores as discussed earlier, we will calculate accuracies and compare them with what was achieved in the paper. The results should be a reproduction of what was described in the paper, i.e. accuracy dropping with AANNs removed (but still many magnitudes above the accuracy of random predictions, 20%) but dropping significantly more with both AANNs and the measure noun phrase + NNS phenomena removed.

Additionally, we will explore the model's ability to generalize using additional morphological information. If the model predicts AANNs more accurately when trained with the original text appended with morphological tags, it would suggest that explicit morphological information can help LMs generalize to better predict unseen linguistic phenomena. To test this, we specifically modify the training dataset by appending the part-of-speech (POS) after each word (separated by a "|" character, e.g. "run" becomes "run|verb"), and trained new models on the modified dataset.

3.1 Datasets

For our unablated dataset, we use the public BabyLM 100M corpus from `https://babylm.github.io/`. For our experiments, we performed several different ablations of our dataset, and trained separate models for each, by referencing Kanishka's HuggingFace [1]. In particular, we henceforth refer to our ablations (and their corresponding experiments) as

- "Normal": unablated, BabyLM 100M corpus.
- "No AANN": the original dataset with 2,448 detected AANN sentences removed. Kanishka and Mahowald [5] used POS tagging and a regex to identiy AANNs, and then refined the list manually. To ensure consistency among all experiments, 2,448 other sentences in the corpus were duplicated.
- "No AANN + No Indef NNS": the original dataset with 2,448 detected AANN sentences *and* 55,373 detected indefinite article + measure word + plural noun sentences removed. Again, as many other sentences were duplicated as the number of removed sentences.

These datasets were used to reproduce experiments found in the paper. For our additional experiment with POS-tag annotations, we used the `spaCy` library to extract the morphological features, such as tense, number, and case, from the original corpus.

Furthermore, after getting results, we realized that further experimentation to test the claims would help make a more robust analysis. In particular, we trained on one more dataset, the (unablated) BabyLM 10M token corpus.

3.2 Tokenizer Descriptions

We tokenized all our datasets using the Albert tokenizer, specifically the pretrained `albert-base-v2` tokenizer [2]. Crucially, `albert-base-v2` has a vocabulary size of 30,000 with and a max position embedding of 512, which are both different from what was used in the paper.

Hence to compare, we also trained our own BPE tokenizer using the HuggingFace `transformers` library [8]. We trained the tokenizer on the BabyLM 100M corpus.

3.3 Model Descriptions

The paper used OPT LM [9] as the autoregressive transformer architecture for all language models, using 12 layers and attention heads, a vocabulary size of 16,384, and training for a maximum of 20 epochs. However, due to library incompatibilities, we instead used the LLaMA 2 [7] open source transformer architecture.

Beyond generative language models, we also needed to train unigram models to compute SLOR scores for each dataset. To do so, we computed the probability of each token as the ratio of its frequency to the total number of tokens in the dataset. This probability distribution was stored in a text file and used for SLOR score analysis (See Section 4).

3.4 Hyperparameters

The hyperparameters for our model were initially set based on the configuration used by Kanishka *et al.* for their experiments with the BabyLM 100M token dataset.

However, due to our compute constraints, we had to reduced several hyperparameters. Please refer to 1 for a comparison of our final configuration.

Table 1: Comparison of LM Traning details

(Hyper)parameter	**Our Configuration**	**Paper**
Tokenizer	albert-base-v2	-
Vocab size	30,000	16,384
Max Position Embeddings	512	256
Number of Attention Heads	12	12
Hidden Size (Embedding Dimension)	512	768
Number of Layers	6	12
Intermediate FFN Dimension	2048	3072
Number of Epochs	10	20
Learning Rate	3e-4	1e-4, 3e-4, 1e-3, 3e-3
Seed	42	-
Train Batch Size	32	32
Eval Batch Size	32	-
Learning Rate Scheduler	Linear (32,000 warmup steps)	Linear (32,000 warmup steps)
Gradient Accumulation Steps	1	-
Total Parameters	56M	97M

3.5 Implementation

Although the paper links to a public codebase: `https://github.com/kanishkamisra/aannalysis/tree/main`, some intermediate scripts were unavailable, and we were unable to find documentation about the codebase. Also, the required GPU was not accessible to us. Therefore we chose to write our own code for the reproduction which can be found here: `https://github.com/RockingMat/AANN-NLP-reproduction.git`. We coded the entire project using shell scripts and python. The libraries used are: datasets, evaluate, nltk, numpy, pandas, scikit-learn, spaCy, tokenizers, torch, tqdm, transformers, minicons, accelerate, inflect, kenlm.

3.6 Computational Requirements

3.6.1 Estimation

The original paper stated that training each language model for 20 epochs it took 21 hours using a single NVIDIA A40 GPU. However, we only have access to a single NVIDIA L4 GPU (via Google Cloud), and we have a limited time frame, so we will reduce the number of training epochs from 20 to 1. Assuming slightly slower GPU performance, we expect the training time to be roughly 10% as that reported by the paper, so roughly 2 hours.

3.6.2 Actual Requirements

Using the same hyperparameters as the paper, we realized that even training with 1 epoch would take 15 hours, much longer than we anticipated (and beyond the scope of our timeframe). We attribute this to the fact that we used the LLaMA architecture instead of OPT, which the paper used.

To reduce the training time, we decreased many hyperparameters to reduce the total parameters our model has. This includes reducing the hidden size from 12 to 6, the intermediate transformer FFN dimension from 3072 to 2048, etc. Details are all shown in 1. With these changes, the total number of parameters in our model was 56M (compared to 97M for the models used in the original paper), and each training run took around 8 hours. In total, we trained 7 different models, and factoring in the tokenization and analysis times, the total GPU usage totals to approximately 70 hours.

4 Results

4.1 SLOR Accuracy

The model's accuracy is evaluated using the SLOR score by having the model predict correctly formed AANNs and comparing these scores to those of counterfactual constructions. Accuracy is determined by how often AANNs receive a higher SLOR score than their counterfactual counterparts.

The model trained on the unablated dataset achieves significantly higher accuracy in predicting AANNs than the model trained on the ablated dataset without AANNs, which, in turn, outperforms the model trained on the dataset excluding both AANNs and NNSs. A random-choice model would have an accuracy of 20%, as each construction has a one-in-five chance of achieving the highest SLOR score among four counterfactuals and the actual AANN. All language models (LMs) performed well above chance.

On average, all models assigned a higher SLOR score to correct AANNs than to their corrupted versions, with the ranking of model performance aligning with the Model Accuracy Comparison. Our SLOR scores were higher than those reported in the referenced paper, but this should not be overinterpreted, as differences in vocabulary size, tokenization, or other factors may account for the variation. The key takeaway is the relative SLOR scores within the same model.

Removing AANNs has a significant impact on accuracy, suggesting that the language model learns grammatical structure from examples. However, even in the absence of AANNs in the training data, the model performs well above chance in predicting them, indicating that it generalizes grammatical knowledge to unseen phenomena—supporting the claim in the paper. Furthermore, the additional performance drop when NNSs (a related phenomenon to AANNs) are removed suggests that language models learn AANN prediction by leveraging examples of related structures. This further reinforces the paper's hypothesis.

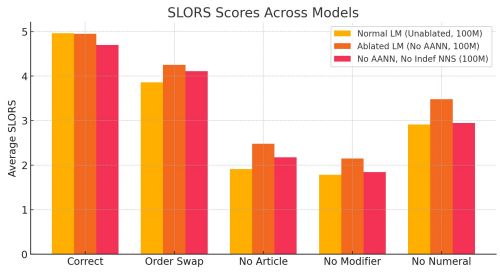
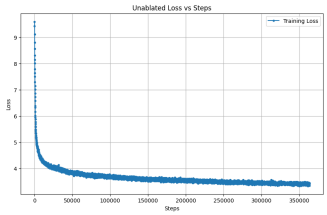
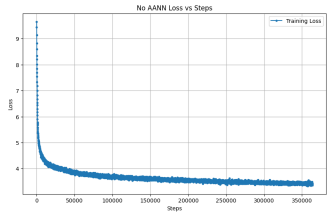


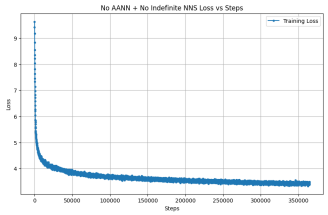
Figure 1: Comparison of SLOR scores across models each trained on full corpus vs. AANN removed vs. AANN and related phenomenon removed



(a) Model trained on full corpus



(b) Model trained on AANN removed corpus



(c) Model trained on AANN & Indef NNS removed corpus

Figure 2: Training loss curve of the three models

4.2 Loss Curve

4.2.1 Training Loss

The training loss exhibited a similar trend across all three models, following a smooth elbow-shaped decline and stabilizing towards the end. This indicates that the training was stable across runs. Please refer to 2 for comparison. Given that the ablation was relatively small compared to the full corpus. we did not anticipate significant differences in training loss. However, when comparing our results with those reported by the authors, we observed a discrepancy. Our models converged to a training loss of approximately 3.5, whereas the original implementation reported a lower final loss of around 2.6 [1].

4.2.2 Evaluation Loss

The evaluation loss followed a similar trend across all models, with minor variations due to ablation. The model trained with the full corpus achieved a loss of 3.626, while the model with AANN ablated reached 3.647, and the model without AANN and indefinite NNS resulted in 3.645. While the differences are small, models trained on the ablated data showed slightly higher loss. One potential factor contributing to this result is the need to resample data to replace the ablated portions, which introduced some level of data repetition. This, in turn, may have led to partial overfitting.

4.3 Model Comparisons: Accuracy

Our models achieved substantially higher accuracy in selecting AANNs over counterfactual phenomena compared to the original paper. Our unablated model obtained an accuracy of 93% on predicting AANNs, whereas the unablated model in the original paper achieved only 70% for the same task. Similarly, our No AANN model obtained an accuracy of 82% while the original paper got 47%. We attribute this discrepancy to the following factors: number of training epochs, parameter count, tokenization scheme, and model architecture.

We trained our model for a single epoch with 56 million parameters on the BabyLM 100M dataset, while the original paper trained their model for 20 epochs with 97 million parameters. We hypothesize that their higher number of epochs and parameters led to overfitting, as the corpus was relatively small, thereby hindering generalizability. In contrast, our model's smaller parameter count and fewer training epochs likely lead to less overfitting, allowing for better generalization to new data — which includes predicting unseen grammatical phenomena such as the AANN.

Additionally, we employed the LLaMA architecture, whereas the original paper used the OPT architecture. This architectural difference likely contributed to the performance gap.

Furthermore, we utilized the `albert-base-v2` tokenizer [2], based on SentencePiece, which had been pretrained on a large corpus and has a larger vocab size of 30,000, whereas the original paper trained a Byte-Pair Encoding (BPE) tokenizer solely on the training data with a smaller vocab size of 16384. Our tokenization scheme likely improved performance by capturing more general grammatical structures.

Below, we show the results of conducting additional experimentation to investigate these hypotheses for the apparent accuracy discrepancy. We were able to test the effect of changing the number of training epochs, and changing the tokenization scheme, but were not able to test using the OPT architecture nor using the same total number of parameters as we did not have access to the required GPU.

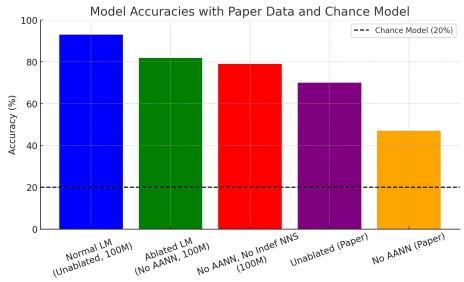


Figure 3: Model accuracy of the four models, compared to the original paper

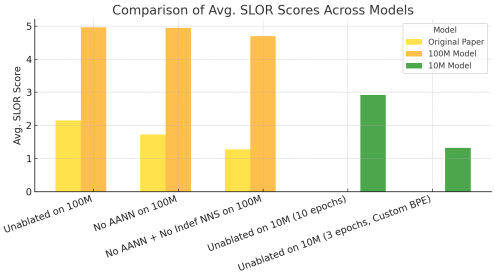


Figure 4: Average SLOR Scores

4.3.1 Hypothesis 1: Number of Training Epochs

One key factor that may influence the training results could be number of training epochs. Due to limited computing power, we trained on the BabyLM 10M corpus for 10 epochs, achieving an accuracy of 0.7982. In comparison, we also trained on the BabyLM 100M corpus for only 1 epoch, which resulted in a significantly higher accuracy of 0.9290. We hypothesize that less epochs leads to less overfitting in our model. Given that the authors trained for 20 epochs, training for fewer epochs could be the reason for the higher accuracy.

4.3.2 Hypothesis 2: Tokenization Scheme

Our second hypothesis regarding training results concerns the tokenization scheme. We use the Albert tokenizer, which has a vocabulary size of 30,000, employs SentencePiece tokenization, and is pretrained on a vast corpus. In contrast, their model uses a BPE tokenizer with a vocabulary of 16,384, trained on the BabyLM 100M dataset. We believe tokenization significantly impacts performance because when we trained a custom model using a tokenizer with identical hyperparameters to theirs (16,384 vocabulary size, 256 positional embedding) but on the 10M dataset for three epochs, the model achieved only 22% accuracy in predicting AANNs—worse than chance. Despite this, the model's evaluation loss was 4.0, relatively close to the 3.6 loss observed in models with strong accuracy. This suggests that tokenization plays a critical role in the language models understanding of grammatical structure, particularly in predicting AANNs. This could be the reason that our models had significantly greater accuracy than that of the original paper.

4.4 Model Comparisons: Average SLOR Scores

We also compare the average SLOR scores generated by our models for the well-formed AANN constructions compared to what was reported in the original paper. Figure 4 presents a comparison of the average SLOR scores between our experiments and the original paper across different experimental conditions.

We see that the general trend matches up: the average SLOR scores decrease from unablated, to no AANN, to no AANN + no indef. NNS, as expected. The decreases in SLOR scores found by our experiments are also comparable to that of the original paper. However, there is a noticeable absolute difference in SLOR scores, which again can be attributed to the hypotheses discussed previously in 4.3.

Examining the average SLOR scores for the two models trained to investigate hypotheses 1 and 2, we see much more comparable average SLOR scores to that of the original paper. When training on more epochs (albeit on a smaller

dataset), average scores drop; and when using a custom BPE tokenizer trained on the same data as that used to train the transformer model (albeit on a smaller dataset), average SLOR scores start to match up with the original paper's data.

This again highlights the significance of the tokenizer in affecting the model's outputs — specifically, the probabilities it assigns to certain sequences of text. Using a tokenizer trained on more data, gives the model higher confidence in predicting the correct sequence; using a tokenizer trained on less data correlates with decreased model generalizability. Furthermore, the fact that the tokenizer was trained on the same data as what is used to train the transformer model can increase overfitting, also possibly contributing to the decrease in model confidence of generating the correct sequences.

4.5 Additional Experiment: Adding POS Tags to Data

Beyond the additional experiments we performed to try to justify the discrepancies in our results compared to that of the paper, we also ran a completely new experiment on training a model on POS-tag augmented data. We hypothesized that this added representation would help the language model (LM) better predict grammatical structure. To implement this, we modified the training data by appending each word with its corresponding POS tag in the format {original word}|{POS tag}. We then trained an LM using the Albert tokenizer with the same hyperparameters as in our original experiment. We also recalculated unigram probabilities on the augmented dataset, and modified the AANN corruption data to incorporate POS tags.

The model trained with POS-tagged data performed significantly worse than those trained on the unmodified dataset, with an accuracy of only 41.2% (see 5). This suggests that appending POS tags to all words in the dataset suggests that appending POS tags may actually hinder, rather than enhance, the LM's ability to learn grammatical structure.

Interestingly, the model trained on the ablated dataset without AANNs, appended with POS tags, achieves a significantly better accuracy of 71.1%.

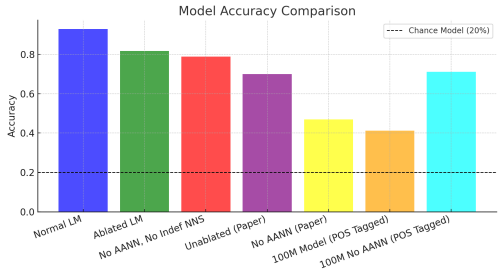


Figure 5: Model accuracy of our six models, compared to the original paper

5 Discussion

Our results broadly support the claims made in the paper. Specifically, we find that language models trained on datasets without AANNs perform far better than chance, suggesting that models generalize grammatical knowledge to unseen phenomenon. Furthermore, the additional drop in accuracy when Indef NNS were removed implies that language models leverage related linguistic phenomena to improve their predictions.

While our SLOR scores were generally higher than those reported in the paper, this difference should not be misinterpreted as a direct performance improvement. Several factors, as we tested independently, contribute to this discrepancy, including differences in the number of training epochs, tokenization schemes, and architectural details.

Indeed, one of our experimental oversights was using the pretrained `albert-base-v2` tokenizer instead of using custom trained BPE tokenizers on our model training data, which should better reproduce the paper's results as suggested in section 4.3. However, this does not impact our conclusion that the paper's claims were supported because the general trends observed were the same. The ranking of SLOR scores across different conditions follows the expected pattern, and all models performed far above the theoretical chance level of 20%. This consistency with the paper's findings strengthens the argument that removing AANNs from the training data has a significant impact on model accuracy, and that related linguistic structures further contribute to learning.

In fact, using the Albert tokenizer for most of our experiments actually reveals another aspect of language model generalizability that was not investigated in the paper: **how generalizable the tokenizer is significantly affects how generalizable the model is**. In future experiments, we hope to explore this aspect more by comparing the impact of using other tokenizers on AANN prediction accuracy.

5.1 Our Approach: Strengths and Weaknesses

One of the key strengths of our approach was leveraging strong LM architectures (LLaMA) and an effective tokenizer (Albert Tokenizer). This enabled our models to fully harness the power of the transformer architecture and test our hypothesis with recent engineering innovations. Additionally, having guidance from the original paper's author provided a significant advantage, allowing us to conduct rigorous ablations and effectively cleanse the data of AANN-related phenomena using a detailed regex.

However, our primary limitation was computational constraints, which hindered our ability to explore a wider range of hyperparameter settings, particularly those requiring extensive training epochs. As a result, we faced challenges in reconciling our findings with the original paper's results.

5.1.1 What was Easy

While a large portion of the paper was complex, having access to Hugging Face libraries made it easy to set up the dataset, so the training itself was not too difficult to reproduce. Additionally, with frequent communication with the author of the paper, we were able to resolve many errors, especially regarding the ablated datasets.

5.1.2 What was Difficult

The biggest difficulty was the lack of computing power. The original paper had far more parameters and number of training epochs than the resources we could afford, so we could had to train models with reduced hyperparameters and number of epochs. The lack of available GPUs also meant that we could not replicate the OPT LM architecture the paper used, so we instead opted to use the Llama 2 architecture.

5.2 Experimental Implications

One of the key takeaways from our experiments is the generalizability of language models to unseen linguistic patterns, even when exposed to a limited number of examples. This intuitively aligns with the dot product attention mechanism in transformer architectures, which captures similarity between token representations. A promising direction for future research is to analyze model parameters, particularly the Q, K, V multi-head attention projections, to investigate whether parameters associated with AANNs and related phenomena, such as indefinite article + plural noun (NNS), exhibit meaningful relationships in the vector space.

An additional interesting observation is that just 2448 AANN sentences out of a total of 110 million lines can lead to a statistically significant impact on accuracy for a specific linguistic phenomenon. This highlights the remarkable adaptability of transformers in *understanding* rare patterns from only a few examples, and suggests that the model has learned from underlying abstract structures, confirming related studies [6]. These findings further support the idea that fine-tuning with a small amount of data can substantially improve model performance, reinforcing evidence from recent studies demonstrating the effectiveness of targeted fine-tuning in NLP models.

Furthermore, the experiment challenges the common assumption that augmenting input with explicit linguistic annotations, such as POS tags, inherently improves a language model's ability to learn grammatical structures. By incorporating POS tags directly into the token stream, the structure of the input data is altered, which can confuse the tokenizer and actually obfuscate grammatical structures.

5.3 Recommendations for Reproducibility

When reproducing the experiments in the paper, refer to both the Github repository and the HuggingFace page of the authors. Run the code to gather and ablate datasets, and crosscheck with the gold standards on HuggingFace.

The original authors' provided codebase can be improved by better organizing the data files into directories that correspond to sections of the paper. Similarly, the code and scripts to train the tokenizer and compute SLOR accuracies can be better organized.

5.4 Communication with Authors

We already had access to the original code for the experiment. However, we discovered that the scripts to generate the training data were missing in the repository, so we contacted the author. The first author of the paper, Dr. Kanishka Misra, responded to our email and updated the repository upon our request, and provided additional detailed information for how to perform ablations on the dataset.

References

[1] Hugging Face. Kanishka misra on hugging face, 2025.

[2] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[3] Kyle Mahowald. A discerning several thousand judgments: Gpt-3 rates the article+ adjective+ numeral+ noun construction. *arXiv preprint arXiv:2301.12564*, 2023.

[4] Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. It's all in the name: mitigating gender bias with name-based counterfactual data substitution. *arXiv preprint arXiv:1909.00871*, 2019.

[5] Kanishka Misra and Kyle Mahowald. Language models learn rare phenomena from less rare phenomena: The case of the missing aanns. *arXiv preprint arXiv:2403.19827*, 2024.

[6] Christopher Potts. Characterizing english preposing in pp constructions. *Journal of Linguistics*, pages 1–39, 2023.

[7] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[8] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45, 2020.

[9] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.