Generating Diverse Negations from Affirmative Sentences

Darian Rodriguez Vasquez[†]
University College London
darian.vasquez.23@ucl.ac.uk

Afroditi Papadaki[‡]
Legal & General
afroditi.papadaki@landg.com

Abstract

Despite the impressive performance of large language models across various tasks, they often struggle with reasoning under negated statements. Negations are important in real-world applications as they encode negative polarity in verb phrases, clauses, or other expressions. Nevertheless, they are underrepresented in current benchmarks, which mainly include basic negation forms and overlook more complex ones, resulting in insufficient data for training a language model. In this work, we propose NegVerse, a method that tackles the lack of negation datasets by producing a diverse range of negation types from affirmative sentences, including verbal, non-verbal, and affixal forms commonly found in English text. We provide new rules for masking parts of sentences where negations are most likely to occur, based on syntactic structure and use a frozen baseline LLM and prompt tuning to generate negated sentences. We also propose a filtering mechanism to identify negation cues and remove degenerate examples, producing a diverse range of meaningful perturbations. Our results show that NegVerse outperforms existing methods and generates negations with higher lexical similarity to the original sentences, better syntactic preservation and negation diversity. The code is available in https://github.com/DarianRodriguez/NegVerse.

1 Introduction

Recent advancements in natural language processing (NLP) have enhanced various applications such as text generation [42], translation [9] and summarization [2], but handling negation remains a significant challenge [13]. Negations are crucial for reasoning and effective communication, as they express denial, contradiction, and absence. This is especially important in critical fields like biomedicine, where misinterpreting negated conditions can have serious consequences. For example, Large Language Models (LLMs) identifying acute bleeding [32] have misclassified cases with negated phrases, revealing bias and a limited understanding of negations [8, 20].

Despite their importance, existing literature has established that language models struggle with negated sentences in tasks such as cloze completion, NLI, QA, and classification [1, 13, 20]. For example, the work in [39] found an inverse scaling trend among models such as GPT-J, GPT-3, Flan-T5, GPT-Neo, and OPT (ranging from 125M to 6B parameters), where larger models tend to perform worse on negation tasks and often produce incorrect answers with high confidence. Similarly, [16] and [18] demonstrated that models like BERT, RoBERTa, GPT-2, BART, and T5 frequently generate identical outputs for opposite statements and misinterpret sentences, such as classifying "The man in the blue shirt is relaxing on the rocks" as entailing "A man is not wearing a blue shirt".

Negations are also underrepresented in most benchmark datasets, both in terms of frequency and complexity. In particular, the works in [12] and [13] show that general-purpose English corpora, such as reviews, conversations, Wikipedia, and books, contain between 22.6% and 29.9% sentences with

[†] Research conducted as an MSc student in the Department of EEE at University College London.
[‡] Research conducted as an academic in the Department of EEE at University College London.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

negations. In contrast, some natural language inference benchmarks have around 8.7%, while other datasets, such as COPA [29] and QQP [5], contain 0.8% and 8.1% respectively.

To improve negation understanding in NLP models, it is crucial to expand annotated datasets to cover various types of negation across different domains [23]. Transformer-based models, such as RoBERTa [21] and BERT [7], often struggle with negations due to their underrepresentation in training data [12]. Current benchmarks primarily focus on verbal negations, lacking syntactic and morphological negations [8, 12]. Although some existing methods address verbal negations [13] or use rule-based augmentation [10], they still cover only a limited range of negation types.

Contributions: To address this issue, we introduce NegVerse, a method that generates a diverse range of syntactic and morphological negations, including non-verbal, verbal, and affixal forms, to enrich the training datasets. NegVerse (a) keeps the produced negated data closely aligned with the original sentences by employing a masking strategy at both token and subtree levels; and (b) addresses the shortage of affixal negation datasets and other negation forms, by assembling 362 unique sentences using LLama-2 and other sources, such as COPA [29] and SNLI [3]. We introduce an efficient masking strategy to insert negations while maintaining sentence fluency. Additionally, a new filtering mechanism is used to exclude degenerate outputs, capturing key negation cues effectively. We use a GPT-2-based model to generate negated sentences and implement a filtering mechanism that screens the generated negations for closeness, duplicates, and validity. We provide extensive empirical evidence of our NegVerse's efficiency and improved performance using relevant criteria such as closeness, diversity, and text quality [22, 31, 43] on various datasets against state-of-the-art baselines.

2 Related Work

LLMs have excelled in various tasks [2, 9, 17, 42], but they consistently struggle with understanding negated sentences [15], which limits their reasoning abilities [39] and sometimes worsens with the model size. Current solutions, such as syntactic data augmentation using Semgrex patterns [14] and the TINA method [10], aim to enhance LLMs' robustness to negations in textual entailment tasks by augmenting training datasets with grammatically correct negated instances. However, they face errors in complex sentences. Other approaches like [34] generate negated data using tense patterns and keywords, while [8] uses WordNet to create true/false sentences. Nevertheless, these methods are not adaptable across diverse datasets. Polyjuice [43] generates sentence perturbations but produces nonsensical outputs and handles a limited range of negation types.

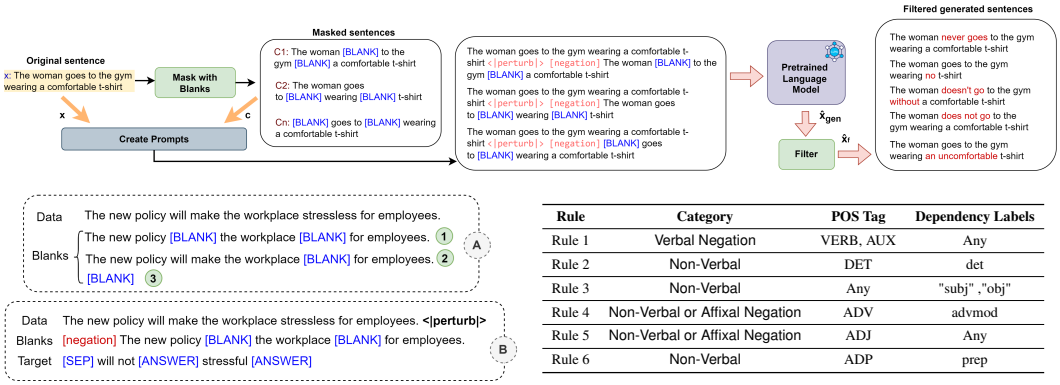
The work in [11] transforms negated sentences into affirmative ones using sentence pairs and back-translation yet it falls short compared to human understanding. Similar to the aforementioned approaches, our goal is to produce new negated sentences to augment the existing datasets. However, in contrast to these methods, our proposed approach generates a wider range of negations – including verbal, non-verbal, and affixal forms – from affirmative sentences. It employs an efficient masking strategy to maintain fluency and structural preservation, resulting in outputs that align lexically better with the original sentences and overcome the limitations of earlier methods.

3 Problem Formulation

We consider a dataset $\mathcal{D} = \{(x_i, \mathbf{c}_i, \hat{\mathcal{X}}_i)\}_{i=1}^{m}$, where x_i denotes an affirmative sentence, $\mathbf{c}_i = \{c_i^{(j)}\}_{j=1}^{n}$ is the corresponding context vector, and $\hat{\mathcal{X}}_i = \{\hat{x}^{(1)}, \hat{x}^{(2)}, \ldots, \hat{x}^{(n)}\}$ the set of all the valid ground-truth negated sentences. The affirmative sentences lack any negation and do not include information guiding the construction of its negation. Each context $\mathbf{c}_i$ includes n–structured prompts with placeholders denoted as [BLANK], indicating where the negation should be applied within a sentence x_i. The set $\hat{\mathcal{X}}_i$ contains the respective valid negated sentences corresponding to context $\mathbf{c}_i$. Our goal is to learn a language generator model $g \in \mathcal{G}$, parametrized by a vector $\boldsymbol{\theta} \in \Theta$, that, given an affirmative sentence x and context $\boldsymbol{c}$, produces a set of negated versions $\hat{\mathcal{X}}_{\text{gen}}$ that closely approximates the ground truth negated set $\hat{\mathcal{X}}$. This is equivalent to solving

$$\min_{\boldsymbol{\theta} \in \Theta} \mathbb{E}_{\mathcal{D}} \left[\frac{1}{m} \sum_{\hat{x} \in \hat{\mathcal{X}}} \ell(\hat{x}_{\text{gen}}, \hat{x}) \right], \tag{1}$$

where $\ell : \Delta^{m-1} \times \Delta^{m-1} \to \mathbb{R}_+$ is the loss function with Δ representing the probability simplex and $\hat{x}_{\text{gen}}$ being the output of the generator model given a pair of an affirmative sentence and a context vector, formally defined as $\hat{x}_{\text{gen}} = g(\boldsymbol{\theta}; x, c)$, with $x \in \mathcal{X}$.



Rule	Category	POS Tag	Dependency Labels
Rule 1	Verbal Negation	VERB, AUX	Any
Rule 2	Non-Verbal	DET	det
Rule 3	Non-Verbal	Any	"subj" ,"obj"
Rule 4	Non-Verbal or Affixal Negation	ADV	advmod
Rule 5	Non-Verbal or Affixal Negation	ADJ	Any
Rule 6	Non-Verbal	ADP	prep

Figure 1: *(Top):* Overview of NegVerse steps. The input sentence x is masked with blanks on specific positions based on structural rules. The masked sentences $\boldsymbol{c}$ and the original sentence x are used to create prompts that are fed to a pretrained language model, which then generates n candidate negations, $\hat{\mathcal{X}}_{gen}$. A filtering mechanism selects the most relevant negations from these candidates, producing the final set, $\hat{\mathcal{X}}_f$. *(Bottom Left):* (A) The input data is masked at different spans using the token [BLANK]. Masks cover different parts of the sentence, or the entire sentence (see sentence 3). (B) Training samples concatenate the input text with the masked sentence and the target words needed to fill the blanks. Each span is separated by the token [ANSWER], and [SEP] separates the context from the answers. During inference, the model accepts the sentence as input, masks the sentence, and predicts the words to fill the blanks, effectively negating the input text. *(Bottom Right)*: Summary of our token selection rules for masking. Tokens are chosen based on Part-of-speech (POS) tags and dependency labels. NegVerse masks either the selected token or its entire subtree.

One challenge with the objective in Eq. 1 is that the generated set $\hat{\mathcal{X}}_{\text{gen}}$ may contain incoherent or irrelevant sentences, leading to nonsensical outputs that reduce model effectiveness. Additionally, the generator requires a context vector to determine the appropriate negation placement, which might not be available for every input. To address these issues, we next propose masking spans in sentences at positions where negation is appropriate, which are used to generate structured prompts that approximate the missing context $\mathbf{c}$, thereby enabling the model to produce accurate and contextually appropriate negations, even without the original context. We also provide a filter that selects only the contextually accurate and meaningful negations $\hat{\mathcal{X}}_{\text{f}}$ from $\hat{\mathcal{X}}_{\text{gen}}$, i.e., $\hat{\mathcal{X}}_{\text{f}} = f(\hat{\mathcal{X}}_{\text{gen}})$. Our framework is illustrated in Figure 1.

4 NegVerse Data Augmentation Method

4.1 NegVerse Prompt Format

Prompt Design. Our model aims to generate negated sentences that meet three key criteria: *closeness*, *quality*, and *diversity*. Closeness ensures the negated sentence minimally differs from the original in structure and meaning. Quality emphasizes grammatical correctness, syntactic accuracy, and coherence. Diversity involves generating a variety of negations across different sentence spans, including verbal, non-verbal, and affixal forms, to enrich the dataset and test multiple negation forms. To maintain closeness, we place [BLANK] tokens at likely negation points in the original sentence x and generate various perturbations for each blank. Our prompt format, adapted from Polyjuice, includes a negation control code, a blank sentence, and outputs separated by the [ANSWER] token, allowing the model to generate up to n possible negations per blank. During training, both individual tokens and entire sentences are masked to teach the model sentence structure and negation patterns. We provide details about the metrics for assessment in Section 5.

Masking/Blanks Placement Strategy. We propose a masking strategy that enhances negation generation by strategically placing blanks in sentences, addressing limitations in traditional methods like Polyjuice, which often miss key elements such as main verbs, auxiliary verbs, contractions like "`wasn't`, and tense variations. Our approach masks key components, including verbs, adjectives, and specific nouns, to support both verbal and non-verbal negations with flexible granularity, allowing for individual token or subtree masking. We developed the token selection rules, summarized in the Bottom Right Table of Figure 1 based on sentence structure analysis and token functions, covering various aspects of sentence construction like determiners, subjects, objects, adverbs, adjectives, and

prepositions, thus enabling the generation of diverse forms of negation. More information about the masking strategy and examples of prompt formats are provided in Appendix B.

4.2 NegVerse Prompt-Tuning Process

Dataset	Samples #
AFFIXAL NEGATION (SST-2)	59
AFFIXAL NEGATION WITH LLAMA 2	130
NON-VERBAL NEGATION (NAN-NLI)	173
Total	362

Table 1: Summary of dataset samples used for NegVerse across different negation types.

We learn the generator using prompt tuning to update only the virtual token embeddings, while keeping a GPT-2 baseline model frozen. To achieve this, we leverage the non-verbal negations from NAN-NLI [40], and the affixal negations from SST-2 [35], as shown in Table 1. Given the underrepresentation of affixal negations in the existing datasets, we develop a new dataset of diverse affixal negations from minimal input [27] using one-shot learning and LLaMA2. During the prompt tuning process, each sentence is masked either by negated parts or entirely, which helps the model learn to handle different spans and levels of context, thereby enhancing its ability to produce accurate negations. The masked sentences are tokenized, padded, and then split into training and validation sets. Additional details on the datasets and hyperparameters are provided in Appendix C.1 and Appendix C.2, respectively.

4.3 Proposed Filtering Mechanism for Degenerate Sentences

Even though our proposed approach is designed to generate fluent and diverse negations, some of the generated outputs may still contain errors or nonsensical phrases. To address this, we propose a filtering process, outlined in Algorithm 1, that normalizes the original and generated sentences by converting it to lowercase and removing trailing punctuation or whitespace (lines 3-5), removes duplicates and uses Levenshtein distance to retain sentences that closely resemble the original (lines 7-10). We use NegBERT to detect the negation cues [19], and we output ϵ negations, that were uniformly sampled from the set extracted by NegBERT, to increase the diversity in the sets (line 12).

Algorithm 1 NegVerse Filtering Mechanism

1: **Input:** $\{\hat{\mathcal{X}}_{\text{gen},i}\}_{i=1}^{m}$: Generated negated set, $\{x_i\}_{i=1}^{m}$: affirmative sentences, ϵ: negations sample number, $B = 0.5$: Levenshtein distance threshold
2: **for** $i \in [m]$ **do**
3: $\quad x'_i \leftarrow \texttt{Trim}(\texttt{Lowercase}(x_i))$
4: $\quad \mathcal{X}'_{\text{gen},i} \leftarrow \texttt{Trim}(\texttt{Lowercase}(\hat{\mathcal{X}}_{\text{gen},i}))$
5: $\quad \mathcal{X}_{\tau,i} = \emptyset$
6: $\quad$ **for** $x'_{\text{gen}} \in \mathcal{X}'_{\text{gen},i}$ **do**
7: $\quad\quad$ **if** $x'_{\text{gen}} \neq$ "" **then**
8: $\quad\quad\quad d \leftarrow \texttt{LevenshteinDistance}(x'_{\text{gen}}, x'_i)$
9: $\quad\quad\quad \mathcal{X}_{\tau,i} = \begin{cases} \mathcal{X}_{\tau,i} \cup \{x'_{\text{gen}}\}, & x'_{\text{gen}} \notin \mathcal{X}_{\tau,i} \wedge d < B \\ \mathcal{X}_{\tau,i}, & \text{o.w.} \end{cases}$
10: $\quad\quad$ **end if**
11: $\quad$ **end for**
12: $\quad \hat{\mathcal{X}}_{\text{f},i} = \{\hat{x}_{\text{f},i}^{(1)}, \ldots, \hat{x}_{\text{f},i}^{(\epsilon)}\} \sim \text{Uni}\big(\texttt{NegBERT}(\mathcal{X}_{\tau,i})\big)$
13: **end for**
14: **Output:** Filtered negation sets $\{\hat{\mathcal{X}}_{\text{f},i}\}_{i=1}^{m}$

5 Empirical Results

Datasets, Baselines and Metrics. We evaluate our approach on five datasets: the Stanford Natural Language Inference (SNLI) dataset [3], the Semantic Textual Similarity Benchmark (STS) [24], COPA dataset [29], and the SemEval Aspect-Based Sentiment Analysis datasets for both restaurant and laptop domains [28]. We compare NegVerse against Polyjuice [43] and evaluate the generated text using (i) Levenshtein Distance (NLD) [25, 30, 38] that measures the minimal edits required to transform one sentence into another; and (ii) Syntactic Tree Edit Distance (Syntactic), which focuses on surface-level changes [45], to assess closeness. For diversity, we use the Self-BLEU Score [46], and for grammaticality and fluency we use a fine-tuned BERT model, following [44]. The quality of the generated sentences is further evaluated using Perplexity (PPL) [25, 38]. We provide more details and results, including generation examples and degenerate cases of NegVerse, in Appendix C.4.

Results and Discussion. We evaluate the performance of our proposed method, NegVerse, and the baseline Polyjuice across all datasets using the closeness, diversity, and quality criteria. The results are presented in Table 2. We observe that NegVerse outperforms Polyjuice in closeness and text quality for both token and subtree masking criteria. Our method achieves a lower Levenshtein distance, indicating better lexical similarity to the original sentences, and a lower syntactic score,

Masking Type	Dataset	Generator	Closeness		Diversity	Quality		
			NLD ↓	Syntactic ↓	Self-BLEU ↓	Fluency ↑	Grammar ↑	PPL ↓
Token Level	SNLI	NegVerse (ours)	**0.200**	**1.275**	0.631	**0.783**	**0.814**	**185.535**
		Polyjuice	0.269	2.363	**0.465**	0.781	0.813	249.741
	STS	NegVerse (ours)	**0.216**	**1.190**	0.594	0.807	0.829	**295.861**
		Polyjuice	0.306	2.360	**0.422**	**0.809**	**0.831**	346.224
	COPA	NegVerse (ours)	**0.317**	**0.824**	0.415	**0.840**	0.850	404.206
		Polyjuice	0.434	2.451	**0.242**	0.838	**0.856**	**249.493**
	Restaurant	NegVerse (ours)	**0.189**	**1.443**	0.655	0.742	0.766	141.715
		Polyjuice	0.233	2.008	**0.564**	**0.743**	**0.768**	**134.103**
	Laptop	NegVerse (ours)	**0.199**	**1.490**	0.629	**0.757**	0.773	163.520
		Polyjuice	0.253	2.192	**0.530**	0.756	0.773	**147.523**
Subtree	SNLI	NegVerse (ours)	**0.200**	**1.275**	0.631	**0.783**	**0.814**	**185.535**
		Polyjuice	0.269	2.363	**0.465**	0.781	0.813	249.741
	STS	NegVerse (ours)	**0.216**	**1.185**	0.606	**0.809**	**0.832**	**296.538**
		Polyjuice	0.403	3.486	**0.328**	0.823	0.845	352.290
	COPA	NegVerse (ours)	**0.205**	**1.300**	0.640	**0.770**	**0.810**	**190.654**
		Polyjuice	0.275	2.400	**0.460**	0.760	0.805	250.890
	Restaurant	NegVerse (ours)	**0.206**	**1.509**	0.634	**0.748**	**0.772**	**143.577**
		Polyjuice	0.361	3.385	**0.404**	0.763	0.786	259.636
	Laptop	NegVerse (ours)	**0.216**	**1.547**	0.608	0.762	0.778	180.621
		Polyjuice	0.382	3.487	**0.371**	**0.780**	**0.794**	**152.233**

Table 2: Experimental results of NegVerse and Polyjuice for token level and subtree masking types using closeness, diversity and quality criteria. The bold numbers indicate the best performance.

reflecting better preservation of syntactic structure. In contrast, Polyjuice often introduces unrelated concepts and alters sentence types, affecting coherence, despite offering greater diversity with a lower Self-BLEU score. Moreover, our results show that both models have similar fluency and grammaticality with token masking, but Polyjuice slightly outperforms NegVerse in these aspects with subtree masking. This suggests Polyjuice performs better under challenging conditions but it does not necessarily produce more relevant text to the original content. Table 3 shows an example of negation generation from the two approaches. We expand our discussion and provide more details and results, including generation examples and degenerate cases of NegVerse, in Appendix C.

	NegVerse	Polyjuice
Original: They were cooking dinner and serving it to their guests. **Masked**: They [BLANK] cooking dinner and serving it to their guests.	1. They weren't cooking dinner and serving it to their guests. 2. They were not cooking dinner and serving it to their guests. 3. They didn't care for cooking dinner and serving it to their guests.	1. They cook cooking dinner and serving it to their guests. 2. They cook in the kitchen and not the dining room because the dining room is farthest from cooking dinner and serving it to their guests.

Table 3: A negation generation example for NegVerse and Polyjuice. [BLANK] marks the masked parts of the original sentence, and the highlighted text shows the generated fill-ins. NegVerse produces outputs that closely mirror the original sentence, while Polyjuice offers more variety in outputs, which contributes to diversity, but can compromise the relevance and fidelity of the text.

6 Conclusions

In this work, we focus on improving the robustness of LLMs robustness on negated statements by proposing NegVerse, a method capable of generating various types of negations, including verbal, non-verbal, and affixal. We provide new masking rules and propose a filtering mechanism to identify negation cues and remove degenerate examples, producing diverse and in parallel meaningful negated sentences. We experiment with five real-world datasets and NegVerse outperforms existing methods and generates negations with higher lexical similarity to the original sentences, better syntactic preservation, and greater negation diversity. Our empirical results also highlight that the proposed approach can generate negated sentences without specific guidance on blank placement.

Limitations and Future Work. While NegVerse excels in preserving syntactic structure and offers a greater variety of negation forms, it still produces some degenerate outputs, particularly when blanks are placed at the end of sentences, leading to grammatically correct but contextually meaningless results. Furthermore, although NegVerse generates a range of affixal negations, certain expected forms are missing. Finally, automated and accurate annotation is essential for the generated negations, as negations can either preserve or invert labels depending on the task.

References

[1] Nicholas Asher and Swarnadeep Bhar. Strong hallucinations from negation and how to fix them, 2024.

[2] Lochan Basyal and Mihir Sanghvi. Text summarization using large language models: A comparative study of mpt-7b-instruct, falcon-7b-instruct, and openai chat-gpt models, 2023.

[3] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In Lluís Màrquez, Chris Callison-Burch, and Jian Su, editors, Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[5] Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In Steven Bethard, Marine Carpuat, Marianna Apidianaki, Saif M. Mohammad, Daniel Cer, and David Jurgens, editors, Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pages 1–14, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[6] Anwoy Chatterjee, Eshaan Tanwar, Subhabrata Dutta, and Tanmoy Chakraborty. Language models can exploit cross-task in-context learning for data-scarce novel tasks, 2024.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[8] Iker García-Ferrero, Begoña Altuna, Javier Álvez, Itziar Gonzalez-Dios, and German Rigau. This is not a dataset: A large negation benchmark to challenge large language models, 2023.

[9] Zhiwei He, Tian Liang, Wenxiang Jiao, Zhuosheng Zhang, Yujiu Yang, Rui Wang, Zhaopeng Tu, Shuming Shi, and Xing Wang. Exploring Human-Like Translation Strategy with Large Language Models. Transactions of the Association for Computational Linguistics, 12:229–246, 03 2024.

[10] Chadi Helwe, Simon Coumes, Chloé Clavel, and Fabian Suchanek. TINA: Textual inference with negation augmentation. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, Findings of the Association for Computational Linguistics: EMNLP 2022, pages 4086–4099, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[11] Md Mosharaf Hossain and Eduardo Blanco. Leveraging affirmative interpretations from negation improves natural language understanding, 2022.

[12] Md Mosharaf Hossain, Dhivya Chinnappa, and Eduardo Blanco. An analysis of negation in natural language understanding corpora, 2022.

[13] Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. An analysis of natural language inference benchmarks through the lens of negation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 9106–9118, Online, November 2020. Association for Computational Linguistics.

[14] Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R Devon Hjelm, Alessandro Sordoni, and Aaron Courville. Understanding by understanding not: Modeling negation in language models, 2021.

[15] Joel Jang, Seonghyeon Ye, and Minjoon Seo. Can large language models truly understand prompts? a case study with negated prompts, 2022.

[16] Myeongjun Jang, Deuk Sin Kwon, and Thomas Lukasiewicz. BECEL: Benchmark for consistency evaluation of language models. In Nicoletta Calzolari, Chu-Ren Huang, Hansaem Kim, James Pustejovsky, Leo Wanner, Key-Sun Choi, Pum-Mo Ryu, Hsin-Hsi Chen, Lucia Donatelli, Heng Ji, Sadao Kurohashi, Patrizia Paggio, Nianwen Xue, Seokhwan Kim, Younggyun Hahm, Zhong He, Tony Kyungil Lee, Enrico Santus, Francis Bond, and Seung-Hoon Na, editors, Proceedings of the 29th International Conference on Computational Linguistics, pages 3680–3696, Gyeongju, Republic of Korea, October 2022. International Committee on Computational Linguistics.

[17] Ehsan Kamalloo, Nouha Dziri, Charles L. A. Clarke, and Davood Rafiei. Evaluating open-domain question answering in the era of large language models, 2023.

[18] Nora Kassner and Hinrich Schütze. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly, 2020.

[19] Aditya Khandelwal and Suraj Sawant. Negbert: A transfer learning approach for negation detection and scope resolution, 2020.

[20] Yitian Li, Jidong Tian, Hao He, and Yaohui Jin. Logical negation augmenting and debiasing for prompt-based methods, 2024.

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

[22] Nishtha Madaan, Inkit Padhi, Naveen Panwar, and Diptikalyan Saha. Generate your counterfactuals: Towards controlled counterfactual generation for text, 2021.

[23] Roser Morante and Eduardo Blanco. Recent advances in processing negation. Natural Language Engineering, 27(2):121–130, 2021.

[24] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. Mteb: Massive text embedding benchmark, 2023.

[25] Van Bach Nguyen, Paul Youssef, Jörg Schlötterer, and Christin Seifert. Llms for generating and evaluating counterfactuals: A comprehensive study, 2024.

[26] R OpenAI et al. Gpt-4 technical report, 2024.

[27] Rajvardhan Patil and Venkat Gudivada. A review of current trends, techniques, and challenges in large language models (llms). Applied Sciences, 14(5), 2024.

[28] Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. SemEval-2014 task 4: Aspect based sentiment analysis. In Preslav Nakov and Torsten Zesch, editors, Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 27–35, Dublin, Ireland, August 2014. Association for Computational Linguistics.

[29] Melissa Roemmele, Cosmin Bejan, and Andrew Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning, 01 2011.

[30] Alexis Ross, Ana Marasović, and Matthew E. Peters. Explaining nlp models via minimal contrastive editing (mice), 2021.

[31] Rachneet Sachdeva, Martin Tutek, and Iryna Gurevych. CATfOOD: Counterfactual augmented training for improving out-of-domain performance and calibration. In Yvette Graham and Matthew Purver, editors, Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1876–1898, St. Julian's, Malta, March 2024. Association for Computational Linguistics.

[32] Thomas Savage, John Wang, and Lisa Shieh. A large language model screening tool to target patients for best practice alerts: Development and validation. JMIR Med Inform, 11:e49886, Nov 2023.

[33] Mo Shen, Daisuke Kawahara, and Sadao Kurohashi. Dependency parse reranking with rich subtree features. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 22:1208–1218, 2014.

[34] Rituraj Singh, Rahul Kumar, and Vivek Sridhar. NLMs: Augmenting negation in language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, Findings of the Association for Computational Linguistics: EMNLP 2023, pages 13104–13116, Singapore, December 2023. Association for Computational Linguistics.

[35] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[36] Christopher Toukmaji. Few-shot cross-lingual transfer for prompting large language models in low-resource languages, 2024.

[37] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[38] Marcos Treviso, Alexis Ross, Nuno M. Guerreiro, and André F. T. Martins. Crest: A joint framework for rationalization and counterfactual text generation, 2023.

[39] Thinh Hung Truong, Timothy Baldwin, Karin Verspoor, and Trevor Cohn. Language models are not naysayers: An analysis of language models on negation benchmarks, 2023.

[40] Thinh Hung Truong, Yulia Otmakhova, Timothy Baldwin, Trevor Cohn, Jey Han Lau, and Karin Verspoor. Not another negation benchmark: The NaN-NLI test suite for sub-clausal negation. In Yulan He, Heng Ji, Sujian Li, Yang Liu, and Chua-Hui Chang, editors, Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 883–894, Online only, November 2022. Association for Computational Linguistics.

[41] Chantal van Son, Emiel van Miltenburg, and Roser Morante. Building a dictionary of affixal negations. In Eduardo Blanco, Roser Morante, and Roser Saurí, editors, Proceedings of the Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics (ExProM), pages 49–56, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

[42] Yaqing Wang, Jiepu Jiang, Mingyang Zhang, Cheng Li, Yi Liang, Qiaozhu Mei, and Michael Bendersky. Automated evaluation of personalized text generation using large language models, 2023.

[43] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S. Weld. Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models, 2021.

[44] Ryoma Yoshimura, Masahiro Kaneko, Tomoyuki Kajiwara, and Mamoru Komachi. SOME: Reference-less sub-metrics optimized for manual evaluations of grammatical error correction. In Donia Scott, Nuria Bel, and Chengqing Zong, editors, Proceedings of the 28th International Conference on Computational Linguistics, pages 6516–6522, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[45] Kaizhong Zhang and Dennis Shasha. Simple fast algorithms for the editing distance between trees and related problems. SIAM J. Comput., 18:1245–1262, 1989.

[46] Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texygen: A benchmarking platform for text generation models, 2018.

Supplemental material

A Types of Negations.

Negation Type	Examples
Verbal Negation (Syntactic type) `not, n't, didn't, cannot, won't, etc.`	They are still not integrated into the German community. We didn't go to the beach because it started raining. She won't be attending the meeting.
Non-Verbal Negation (Syntactic type) `no, nothing, nowhere, nobody, none, without, etc.`	I have no doubt that we will reach our goal. He found nothing in the drawer. The lost keys were found nowhere in the house. He completed the task without any help.
Affixal Negation (Morphological type) `un-, in-, dis-, -less, non-, etc.`	Her reaction was unexpected given the circumstances. She felt hopeless after repeated failures. The product was non-existent on the shelves.

Table 4: Overview of verbal, non-verbal, and affixal negation forms, with corresponding examples demonstrating their application in sentences.

There are two main types of negations: morphological and syntactic negations, as outlined in Table 4. Morphological negations create negative expressions by adding affixes to words, either as prefixes or suffixes. A prefixal negation adds prefixes to the beginning of words and includes common prefixes like `un-` (e.g., *unhappy*), `in-/im-/il-/ir-` (e.g., *inaccurate*, *impossible*, *illegal*, *irrelevant*), `dis-` (e.g., *disagree*), and `non-` (e.g., *nonexistent*). A suffixal negation adds suffixes to the end of words and includes the common suffix `-less` (e.g., *hopeless*, *meaningless*). These affixes alter the meaning of the base words to convey negation, absence, or opposition [41]. Syntactic negations utilize grammatical structures and specific words to negate a sentence. This typically includes negative particles like `not` and `no` (e.g., *She is not coming*; *There is no water*), negative pronouns like `nobody` and `nothing` (e.g., *Nobody knows*; *Nothing happened*), negative adverbs like `never` and `nowhere` (e.g., *She never comes*; *They went nowhere*), negative determiners like `no` and `neither` (e.g., *No students passed*; *Neither option is good*), and negative conjunctions like `nor` and `neither...nor` (e.g., *She didn't call, nor did she email*; *Neither he nor his friends came*).

B Prompt Design

In this section, we provide further details on our six rules of the masking strategy, which were outlined earlier in Section 4.1.

Rule 1: The first rule targets verbal negations by selecting verbs (`VERB`) and auxiliaries (`AUX`) for masking, as these are key components in forming negations. For instance, in the sentence "*She was eating an apple*", masking "`was`" and "`eating`" allows the model to generate the negation "*She was not eating an apple.*" This approach effectively negates the core actions or states in the sentence, as illustrated in Figure 2.

Rule 2: The second rule targets non-verbal negation by focusing on determiners (`DET`) with the dependency label `det`. Determiners like "`the`", "`a`", and "`an`" are crucial for defining noun phrases. The model selects a determiner and the following token for transformation, such as changing "`the man`" to "`no one`", as shown in Figure 2. Unlike Rule 3, which may negate entire phrases, Rule 2 specifically alters the determiner. Other examples include:

Peter wanted ~~some~~ part of it. → Peter wanted none of it.

Rule 3: This rule focuses on negating objects ("obj") and subjects ("subj"), which are essential for defining who is performing an action and what is being acted upon. Negating the subject (‘"subj"‘) changes who or what is performing the action. For instance:

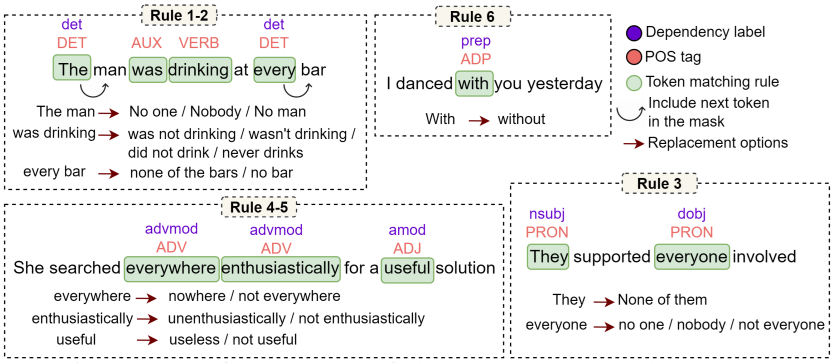


Figure 2: Illustrative example of sentences that follow our proposed blank placement rules. Although some sentences comply with multiple rules, only the words matching the specific rule are highlighted in green for each case. Below each sentence, possible negations that can be introduced by filling in the blanks are provided. This example demonstrates how this placement strategy can produce diverse forms of negation. The arrow sign (→) indicates that when the word is a determiner (DET), it masks the accompanying noun or adjective, allowing the model to generate richer negations.

~~They~~ will attend the meeting → No one will attend the meeting
~~They~~ will attend the meeting → None of them will attend the meeting

Negating the object (‘"obj"‘) changes what is being acted upon, affecting the outcome of the action. For example:

She found ~~the key~~ → She found nothing
She went to ~~the gym~~ → She went to no gym

Rule 4: This rule targets adverbs (`ADV`) with the dependency label `advmod` for non-verbal or affixal negation. By masking adverbs, the rule generates various negations, such as changing "`everywhere`" to "`nowhere`" or "`not everywhere`", and "`enthusiastically`" to "`unenthusiastically`" or "`not enthusiastically`". This approach modifies the action's scope or intensity and incorporates morphological changes.

Rule 5: This rule enables non-verbal or affixal negation by targeting adjectives (`ADJ`), allowing for direct negation or morphological changes. For example, "*The solution is useful*" can be transformed to "*The solution is useless*" (affixal) or "*The solution is not useful*" (non-verbal).

Rule 6: This rule handles non-verbal negation by targeting prepositions (`ADP`) that provide context such as location or time. It is used less frequently and only when the mask subtree is active, due to its limited variations. For example,

She will meet us ~~at the restaurant~~ → She will meet us nowhere

We provide an example illustrating the impact of the six proposed rules in Figure 4. We also show the broader context included in the negation from a subtree's selected token and syntactic dependents [33] in Figure 3.

C Additional Experimental Setup Details and Results

C.1 Training Data Details

In section 4.2, we provided information about the tuning process of NegVerse by combining the non-verbal negations from NAN-NLI [40], and the affixal negations from SST-2 [35] and the new dataset we generated using LLaMA2. In what follows, we provide more information about these datasets.

NAN-NLI: This dataset is used to evaluate models' capabilities in understanding and processing sub-clausal negation instances in natural language applications. Sub-clausal negation occurs within a clause, rather than negating the entire clause itself. The dataset annotates various aspects of negation, including verbal vs. non-verbal, analytic vs. synthetic, and clausal vs. sub-clausal negation types. Additionally, it captures the constructions used in negation instances, as well as the operations applied

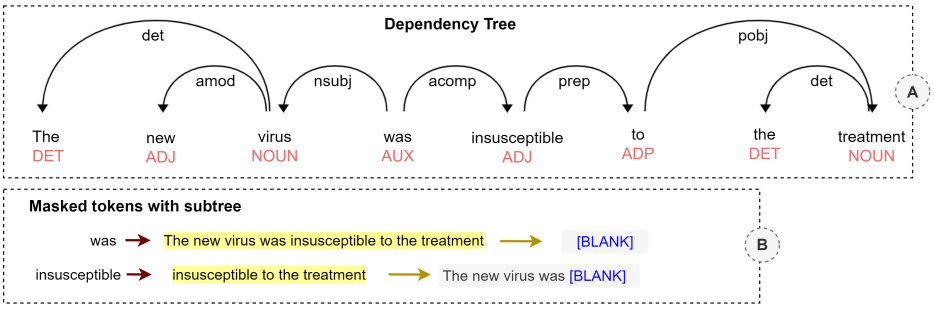


Figure 3: Dependency parse tree representing the grammatical structure of an example sentence. (A) The syntactic structure of the sentence, with arcs representing grammatical dependencies between words. Dependency labels (dep tags) are displayed on the arcs, and part-of-speech tags (POS) are shown under each word, illustrating the sentence's syntactic structure. (B) Tokens within the subtree rooted at the selected token are highlighted in yellow. The highlighted tokens are then masked with [BLANK] instead of just the individual token. If a verb is selected, all words dependent on it within the sentence are included in the subtree, resulting in the entire sentence being masked.

Figure 4: An illustrative example of sentence masking. The masking function considers a maximum of two tokens per sentence, and six different masked sentences. Part (A) represents the masked sentence with Polyjuice automatic masking, where the main verb is masked in none of the options, nor were the adjectives with possible affixal negated forms like "insusceptible". Additionally, in Option 3, a [BLANK] was inserted rather than replacing a token. Part (B) shows how the proposed approach masks a sentence. In particular, Option 6 masks the adjective, Option 4 masks the main verb, and the other options mask in places to produce non-verbal negations.

to construct hypotheses [40]. The dataset provides a list of construction types used in negation instances, where most cases involve non-verbal negations, as shown in Table 5.

SST-2: This dataset is a collection of movie reviews classified as negative or positive. It includes two types of negations: syntactic (SYN) and morphological (AFFIX) [12]. For training the model with this data, only the AFFIX annotations were filtered, where the negation cue could be translated to a positive sentiment. For example, "`unpleasant`" can be translated to "`pleasant`". The sentences were converted to positive by applying manual rules, considering cases where the negation cue starts with "`un`" or ends with "`less`", using a dictionary of affixal negations from [41]. For instance:

Original: The film is quiet, threatening, and ~~unforgettable~~.
Negated: The film is quiet, threatening, and forgettable.

Construction Type	Definition	Example
Not + quantifiers	Not combined with a quantifier, (e.g., Not all, not every, not many, not much).	Not one person supported the proposal.
Not + focus particles	"Not even" denotes clausal negation; "not only" indicates subclausal negation with a positive tone.	Not even Ed approved of the plan.
Not + degree expressions	Marks sub-clausal negation by reducing the intensity of adjectives or adverbs (e.g., "not very confident").	It somehow sounded not quite right.
Not + affixal	Affixal negations of adjectives and adverbs.	It was a not undistinguished private university with a large endowment.
Not in coordination	"Not" in a coordinative construction negates only one part of the conjunction, indicating subclausal negation.	They are now leaving not on Friday but on Saturday
Not in verbless subordinate clauses	"Not" can negate only the verbless subordinate clause	We need someone not afraid of taking risks.
Not in implicit propositions with that	Denies something anticipated or implied in the context	There are spare blankets in here, not that you'll have any need of them.
Absolute negators	Indicates complete nonexistence within a prepositional phrase (e.g., no, never)	They were friends in no time .
Approximate negators	Suggests near-zero frequency with a positive implication (e.g., rarely, seldom)	She rarely goes out these days.

Table 5: Definitions and examples of different negation types within the NAN-NLI dataset. The highlighted text in each example illustrates the specific negation construction being discussed [40].

```
You are a helpful assistant to generate sentence.

Example follow this structure:

Unattainable: The company's goals seemed unattainable given the
current market conditions.

Attainable: The company's goals seemed attainable given the
favorable market conditions.

Generate the sentence using the word {neg_word}.
Then change the sentence to use the word {pos_word}, keeping
minimal changes.
```

Figure 5: Prompt template used to generate data with affixal negations by leveraging one-shot learning with an instruction-following LLM assistant using Llama-2-7b-Chat. The text in blue indicates where the new pair of words is inserted for inference.

New dataset: Affixal Negations Generated from LLaMA2. Affixal negations, using prefixes or suffixes, were underrepresented in existing datasets. To address this, we created a new dataset with additional sentence pairs focused on affixal negations using the Llama-2-7b-Chat model and one-shot learning, enabling efficient generation of diverse examples from minimal input [27].

We used prompt engineering to guide the model in generating and modifying sentences. The prompts provided structured examples of affixal negations and their transformation into positive forms with minimal changes, as shown in Figure 5. For instance, the model replaced "unattainable" with

Original Sentence	Negated Sentence	
The water in the lake was ~~pure~~, making it ~~safe~~ for drinking.	The water in the lake was impure, making it unsafe for drinking.	✓
The employee's work was ~~worthy~~ of the bonus due to the ~~exceptional effort~~.	The employee's work was unworthy of the bonus due to the lack of effort.	✓
The new employee's ~~enthusiasm and willingness to learn~~ made it ~~easy~~ for him to receive the necessary support from his colleagues.	The new employee's lack of experience made it difficult for him to receive the necessary support from his colleagues.	X
The new employee ~~quickly connected~~ with his colleagues ~~and became an integral part of the team~~.	The new employee struggled to connect with his colleagues due to his shyness.	X

Table 6: Comparison of negated and original sentences generated with Llama 2 to illustrate affixal negations examples for training. The generated data were manually analyzed for validity, where sentences that did not correctly convey affixal negations were eliminated from the dataset. Sentences with substantial word substitutions were also excluded, as the goal is to have samples with minimal changes. Parts of the original sentences that were eliminated are crossed out, while the validity of the changes is indicated by ✓ for correct pairs and X for incorrect ones.

"attainable" in a sentence. Table 6 shows that the model occasionally failed to make minimal changes or correctly apply affixal negations, resulting in the exclusion of such cases from the training dataset.

The Llama-2-7b-Chat model [37] was selected for generating affixal negation sentences from limited training data due to its ability to produce coherent, contextually accurate text with minimal errors and its strong performance in one-shot learning [6, 36]. Additionally, its use is cost-free, in contrast to models such as GPT-3 [4] and GPT-4 [26].

C.2 Hyperparameters

We train the model for 31 epochs using the AdamW optimizer, which integrates weight decay directly into the optimization process. The learning rate is set to 2.5×10^{-2} to strike an optimal balance between training efficiency and convergence speed. A weight decay of 1×10^{-3} is employed to address overfitting by penalizing large weights, while a batch size of 16 ensures stable gradient updates within memory constraints. Additionally, 24 virtual tokens are used to prompt-tune the model, allowing for focused adaptation on the specific task. Various hyperparameters were tested and monitored through the learning curve analysis, with these settings yielding the best results in terms of stability and performance. Out of the total 124,458,240 parameters in the pre-trained language model, only 18,432 are trainable, all coming from the virtual token embeddings. This represents just 0.0148% of the model's parameters, demonstrating the efficiency of the soft-tuning approach.

C.3 Software and Hardware

The proposed algorithms and experiments are implemented in Python, utilizing the PyTorch library. The experiments were conducted using a single NVIDIA Tesla A100 GPU. The official implementation of NegBERT can be found at `https://github.com/adityak6798/Transformers-For-Negation-and-Speculation`.

C.4 Additional Results

Negation Comparisons In Table 7, we provide illustrative examples of how NegVerse and Polyjuice handle verbs and sentence masking differently. When a verb is masked, NegVerse generates various forms of negations, including both contracted and uncontracted versions. In contrast, Polyjuice often introduces unrelated concepts such as "`dining room`", "`Germany`" and "`t-glass`", which are not present in the original sentence and disrupt its overall coherence. Additionally, when entire sentences are masked, NegVerse typically produces outputs that closely resemble the original, while Polyjuice frequently creates entirely different sentences, sometimes altering the sentence type altogether, such

	NegVerse	Polyjuice
Original: They were cooking dinner and serving it to their guests. **Masked**: They [BLANK] cooking dinner and serving it to their guests.	1. They weren't cooking dinner and serving it to their guests. 2. They were not cooking dinner and serving it to their guests. 3. They didn't care for cooking dinner and serving it to their guests.	1. They cook cooking dinner and serving it to their guests. 2. They cook in the kitchen and not the dining room because the dining room is farthest from cooking dinner and serving it to their guests.
Original: Everybody loves the coffee in London. **Masked**: [BLANK]	1. Nobody loves the coffee in London. 2. Nobody hates the coffee in London.	1. What is the last name of the person that Vickers breaks up with? 2. What is the full name of the person who has a brother named "Doc"?
Original: The gourmet dinner was delicious and expensive. **Masked**: The gourmet dinner was [BLANK] and [BLANK].	1. The gourmet dinner was unappealing and not expensive. 2. The gourmet dinner was unappealing and expensive. 3. The gourmet dinner was unappealing and not expensive.	The gourmet dinner was served in a t-glass rather than a glass, because the t-glass was better and tastier.
Original: He stayed at the hotel. **Masked**: He stayed [BLANK].	1. He stayed not at the hotel. 2. He stayed not at the hotel. 3. He stayed away from the hotel.	1. He stayed in Germany for three years before moving back with his family to Japan.

Table 7: Examples of negation outputs for NegVerse and Polyjuice showing differences in closeness, quality and diversity. The [BLANK] marks the masked parts of the original sentence, with the highlighted text showing the generated fill-ins. NegVerse typically produces outputs that closely mirror the original sentence, maintaining coherence. In contrast, Polyjuice offers more varied outputs, which, while contributing to diversity, sometimes compromise relevance and fidelity to the source text.

as changing an affirmative statement into a question, compromising this way the coherence and relevance of the output.

On Perplexity Being a Misleading Metric. In Table 8, we show the impact of word choices on perplexity (PPL), fluency, and grammaticality in text generation. For instance, the sentence "*Her sweater is uncomfortable and pretty*" scores high in fluency and grammaticality but has a notably high PPL. The increased perplexity suggests that the word "`uncomfortable`", despite its grammatical correctness and naturalness, is less predictable for the model. This may be due to the less frequent occurrence of affixal negation – such as "`un-`" – in GPT-2's training data, making such constructions more challenging, especially when paired with a positive attribute like "`pretty`". When the sentence is rephrased to "*Her sweater is not comfortable and pretty*", the PPL drops significantly, indicating a more predictable structure for the model. However, this rephrasing results in slightly lower fluency and grammaticality scores.

The sentence "*He doesn't offer a rational explanation for his decision*" scores high in both fluency and grammaticality with a very low perplexity, demonstrating a case where low perplexity correlates well with high-quality metrics. In contrast, the sentence "*She spent the day wearing nothing sweater*" exhibits high perplexity but maintains an unusually high fluency score, despite being nonsensical. This discrepancy indicates that perplexity and fluency scores may not always align with human judgment. Additionally, the phrase *"didn't slept"* has a higher grammaticality score than that of *"none of the kids"*, highlighting that these metrics do not always capture grammatical nuances accurately.

These examples indicate that PPL and and quality scores can be useful as a general measure of a model's predictive capabilities, but they should not be used in isolation to assess the naturalness and coherence of the generated data.

Degenerate Cases Analysis for NegVerse. As noted earlier in the limitations section, despite NegVerse's strong performance in preserving syntactic structure and offering a greater variety of

	NegVerse	Gramm.	Flu.	PPL
Original: Her sweater is comfortable and pretty.	1. Her sweater is uncomfortable and pretty	0.894	0.834	856.522
Masked: Her sweater is [BLANK] and pretty.	2. Her sweater is not comfortable and pretty	0.834	0.758	406.590
Original: He offers a rational explanation for his decision.	1. He doesn't offer a rational explanation for his decision.	0.961	0.964	26.859
Masked: He [BLANK] a rational explanation for his decision.	2. He lacks a rational explanation for his decision.	0.974	0.969	81.216
Original: A group of kids plays in the spray of water from a fountain.)	1. Not a group of kids plays in the spray of water from a fountain.	0.770	0.697	84.193
Masked: [BLANK] plays in the spray of water from a fountain.	2. None of the kids plays in the spray of water from a fountain.	0.800	0.749	80.611
Original: She spent the day wearing an unique sweater	1. She spent the day wearing no sweater	0.939	0.877	596.420
Masked: She spent the day wearing [BLANK] sweater	2. She spent the day wearing nothing sweater X	0.956	0.956	979.119
Original: The cat slept peacefully in the sun.	1. The cat did not slept peacefully in the sun. X	0.790	0.712	381.641
Masked: [BLANK] slept peacefully in [BLANK].	2. The cat didn't slept peacefully in the sun. X	0.861	0.782	177.690

Table 8: Quality evaluation of various generated sentences. This table compares sentences generated by NegVerse with their original counterparts, showing metrics for grammaticality (Gramm.), fluency (Flu.), and perplexity (PPL). The examples highlight how different word choices and constructions affect these metrics, and reveal insights into the model's performance and limitations in maintaining naturalness and coherence. X indicates sentences that are grammatically incorrect.

	NegVerse	Gramm.	Flu.	PPL
Original: He offers a rational explanation for his decision. **Masked**: He offers a rational explanation for [BLANK].	1. He offers a rational explanation for |> [|> [|> [|> [|> [|> [|> [|> [|> [|> [|> [|> [|>	0.769	0.760	3.511
	2. He offers a rational explanation for his decision.	0.976	0.980	44.482
Original: a young woman fishing off a dock at sunset. **Masked**: A young woman fishing off a dock at [BLANK].	1. A young woman fishing off a dock at a young woman fishing off a dock at dusk..... not a young woman ..... a young woman..	0.822	0.812	16.623
	2. A young woman fishing off a dock at no sunset.	0.737	0.688	550.823
Original: A bald headed man in business casual attire is amused by something happening off-screen. **Masked**: A bald headed man in business casual attire [BLANK] amused by something [BLANK] off-screen.	1. A bald headed man in business casual attire isn't amused by something EMPTY off-screen.	0.789	0.756	278.892

Table 9: Comparison of model outputs across different inputs with associated metrics. This table contrasts the generated sentences with their original counterparts, showcasing variations in grammaticality (Gramm.), fluency (Flu.), and perplexity (PPL). It includes examples of degenerate text and outputs with empty tokens, highlighting issues such as repetition, low coherence, and incomplete responses.

negation types, it occasionally produces degenerate outputs. These issues are particularly evident with blank placements at the end of sentences, sometimes leading to grammatically correct but contextually meaningless results. We provide a number of degenerate output examples in Table 9.

The first example of the table illustrates a case where the model generates a sequence of special characters, such as |> [|> [|> [|> [|> [|> [|> [|> [|> [|> [|>, in response to the prompt. This output is marked by a low grammaticality score of 0.769 and a fluency score of 0.760, indicating deficiencies in both grammatical correctness and fluency. Despite these low scores, the perplexity is

NegVerse Filtering Examples	
Original:	They remained loyal to their cause despite the challenges.
Generated:	Not remained loyal to their cause despite the challenges. None of them remained loyal to their cause despite the challenges. Not as long as they remained loyal to their cause despite the challenges. ~~They remained loyal to their cause despite the challenges..~~ ~~They remained loyal to their cause despite lack of challenges..~~ ~~They remained loyal to their cause despite lack of adversity..~~ ~~They remained loyal to their cause despite lack of adversity..~~ ~~They remained loyal to their cause despite their struggles..~~ They remained indifferent to their cause despite the challenges. They remained not loyal to their cause despite the challenges. They remained dispirited to their cause despite the challenges. They did not remain loyal to their cause despite the challenges. They didn't remain loyal to their cause despite the challenges. They weren't loyal to their cause despite the challenges. ~~Not remained loyal to their cause despite the challenges.~~ They never remained loyal to their cause despite the challenges. ~~None of them remained loyal to their cause despite the challenges.~~
Filtered:	None of them remained loyal to their cause despite the challenges They remained indifferent to their cause despite the challenges They didn't remain loyal to their cause despite the challenges They never remained loyal to their cause despite the challenges They weren't loyal to their cause despite the challenges Not remained loyal to their cause despite the challenges They remained dispirited to their cause despite the challenges Not as long as they remained loyal to their cause despite the challenges They did not remain loyal to their cause despite the challenges They remained not loyal to their cause despite the challenges

Table 10: This table compares the original sentence with perturbations generated by NegVerse and the final filtered versions. The **Original** text serves as the reference. The **Generated** section displays different sentences produced by the model, with red text (text) indicating negation cues that were not detected by the negation detector. The **Filtered** section shows sentences selected based on criteria such as the elimination of repeated sentences, removal of sentences without negations, and filtering based on a Levenshtein distance threshold. Key terms in the filtered sentences are highlighted as negation cues in purple (purple). These examples showcase the effectiveness of the filtering criteria and highlight discrepancies in negation detection, particularly where NegBERT fails to correctly detect affixes and multi-word negation cues.

notably low, suggesting that the model finds this sequence statistically probable, although the output remains largely nonsensical.

In contrast, for the same input, the well-formed output "*He offers a rational explanation for his decision.*" achieves high grammatical and fluency scores, but exhibits a higher perplexity compared to degenerations. It is noteworthy that degenerate outputs occur in instances where "[BLANK]" appears at the end of the sentence. Removing the period, as seen in the case with "[BLANK]", allows the model to generate a correct output, indicating that the presence of the period may contribute to issues in the generation process.

The second example in Table 9 demonstrates an issue with repeated text. In this case, the model generates a response that includes repeated and incoherent phrases, such as *" not a young woman..... not a young woman..... not a young woman....."*. This repetitive output is truncated in the table for visibility but illustrates a broader problem with the model's generation process. The repetition contributes to a low perplexity but results in a lack of coherence and meaningful content.

In the output example featuring the empty token, "[EMPTY]" is used as a placeholder to represent missing or unspecified content. This indicates that the model was unable to generate a specific word or phrase, leading to a vague or incomplete response. The use of the empty token highlights a limitation in the model's ability to produce coherent text in certain contexts. Additionally, not every position in a sentence is suitable for introducing a negation, which further contributes to the model's challenges in generating appropriate and contextually accurate content.

Examples of NegVerse Application. In Table 10, we provide an example of the filtered results that were selected from sentences generated by the model using Algorithm 1. Recall that the proposed

filtering mechanism uniformly samples from sentences containing effective negations close to the original affirmative sentence. As shown in the provided example, NegVerse sometimes misses certain negation cues, such as "lack of" in specific contexts. Nevertheless, the model successfully identified other types of negation, such as "indifferent" and "dispirited" which, although not direct affixal negations of "loyal" are still relevant for expressing negation. This behaviour may stem from the model's limited training data on affixal negations and insufficient exposure to diverse contexts. Despite these limitations, the model effectively detects most non-verbal negations, such as "never" and "none of" as well as verbal forms like "didn't" and "weren't".

NegVerse Generations with [BLANK]	
Original:	She is always happy to lend a helping hand to her friends.
Generated:	['She is never happy to lend a helping hand to her friends.', 'She is not always happy to lend a helping hand to her friends.', 'She is not happy to lend a helping hand to her friends.']
Filtered:	['She is never happy to lend a helping hand to her friends', 'She is not happy to lend a helping hand to her friends', 'She isnot always happy to lend a helping hand to her friends']
Original:	The design makes the new car highly desirable.
Generated:	['The design makes the new car highly undesirable.', 'The design makes the new car highly desirable.', 'The design makes the new car highly undesirable.', 'The design makes the new car highly un desirable.']
Filtered:	['The design makes the new car highly un desirable', 'The design makes the new car highly undesirable']
Original:	They remained loyal to their cause despite the challenges.
Generated:	['They remained loyal to their cause despite the challenges.', 'They remain loyal to their cause despite the challenges.', 'They did not remain loyal to their cause despite the challenges.']
Filtered:	['They did not remain loyal to their cause despite the challenges']
Original:	Technology allows us to connect with people across the globe instantly.
Generated:	['Technology allows us to connect with people across the globe instantly.', 'Technology allows us to connect with people across the world.', 'Technology allows us to connect with people across the globe.']
Filtered:	[]
Original:	The cat napped peacefully.
Generated:	['The cat napped peacefully.', 'The cat napped peacefully..', 'The cat did not nap peacefully.']
Filtered:	['The cat did not nap peacefully']

Table 11: Generated and filtered outputs for affirmative sentences where the entire sentence is masked. The model tends to negate only a single part of the sentence rather than introducing diverse perturbations. Additionally, as the length of the sequence increases, the performance of the model in negating the sentence deteriorates, resulting in cases where the model simply repeats the original sentence, leading to no new or meaningful output. The negation cue produced by NegVerse is highlighted.

NegVerse Generation with [BLANK] for Complete Sentence Masking. In Table 11 we show examples where the model generates negated sentences without explicit guidance on blank placement. The results show that the model can produce various negations for simpler sentences effectively. However, the model's performance becomes inconsistent with more complex sentences, leading to issues such as repetition or awkward phrasing. This variability indicates that while the model handles basic negations well, its ability to consistently apply negation across different sentence structures without precise blank placement can be limited.

C.5 Evaluation Metrics

In this section, we provide further details on the evaluation metrics used in the main manuscript to assess the performance of both the proposed approach and the baseline methods. We consider metrics to examine various aspects of negated text generation, including closeness, fluency, and diversity.

(Average) Levenshtein Distance (NLD): This metric measures the average minimum number of edits needed to transform one tokenized sentence into another. The formal definition is provided below:

$$\text{NLD} = \frac{1}{N} \sum_{i=1}^{N} \frac{d(x_i, \hat{x}_{\text{gen},i})}{\max(|x_i|, |\hat{x}_{\text{gen},i}|)}$$

where x_i denotes the reference sentence, $\hat{x}_{\text{gen},i}$ is the generated negated sentence from the model, and n is the total number of sentence pairs. This metric has been widely used in various studies to evaluate the similarity between sentence pairs, particularly in counterfactual evaluations.[25, 30, 38].

Self-BLEU Score: This metric evaluates the diversity within a set of generated texts by measuring their similarity to each other, as opposed to traditional BLEU, which compares generated texts to reference texts [46]. The Self-BLEU score is calculated as:

$$\text{Self-BLEU} = \frac{1}{m} \sum_{i=1}^{m} \text{BLEU}\left(\hat{x}_{\text{gen},i}, \hat{\mathcal{X}}_{\text{gen}} \setminus \{\hat{x}_{\text{gen},i}\}\right)$$

where m is the total number of generated sentences, $\hat{x}_{\text{gen},i}$ is the i-th generated sentence, and $\hat{\mathcal{X}}_{\text{gen}} \setminus \{\hat{x}_{\text{gen},i}\}$ represents the set of all generated sentences except $\hat{x}_{\text{gen},i}$. A lower Self-BLEU score indicates higher diversity, while a higher score suggests more similarity among outputs.

Perplexity: This metric evaluates how well a language model predicts a sequence of tokens, with lower perplexity indicating better fluency. It has been widely used for fluency assessment in text generation models like GPT-2 [25, 38]. For a negated sentence $\hat{x} = (\hat{z}_1, \hat{z}_2, \ldots, \hat{z}_n)$, where n is the sentence length, the perplexity $\text{PPL}(\hat{x})$ is given by:

$$\text{PPL}(\hat{x}) = \exp\left(-\frac{1}{n} \sum_{i=1}^{n} \log p_\theta(\hat{z}_i \mid \hat{z}_{<i})\right)$$

where $\log p_\theta(\hat{z}_i \mid \hat{z}_{<i})$ is the log probability of token $\hat{z}_i$ given the preceding tokens $\hat{z}_{<i}$.