Generating Diverse Negations from Affirmative Sentences

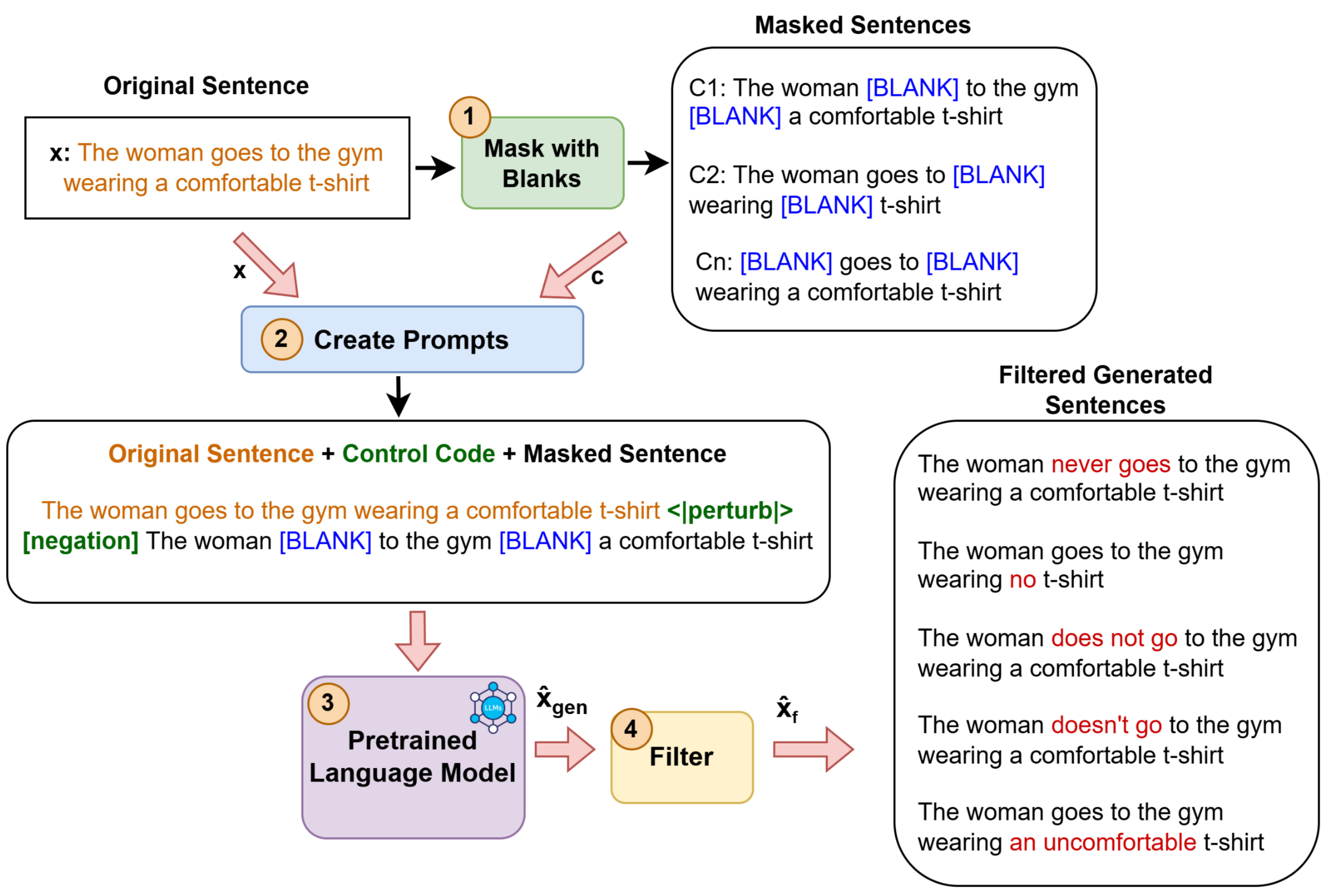
Darian Rodriguez [1] Afroditi Papadaki [2]

[1] University College London [2] Legal & General

Motivation

- Misinterpreting negations in critical fields such as healthcare, law, and finance can result in profound societal consequences [32].
- Negations are underrepresented in benchmark datasets, limiting model exposure to diverse and complex scenarios [12,13].
- Language models often struggle to distinguish between opposite statements and perform worse with increased model size [1,13,20,39].

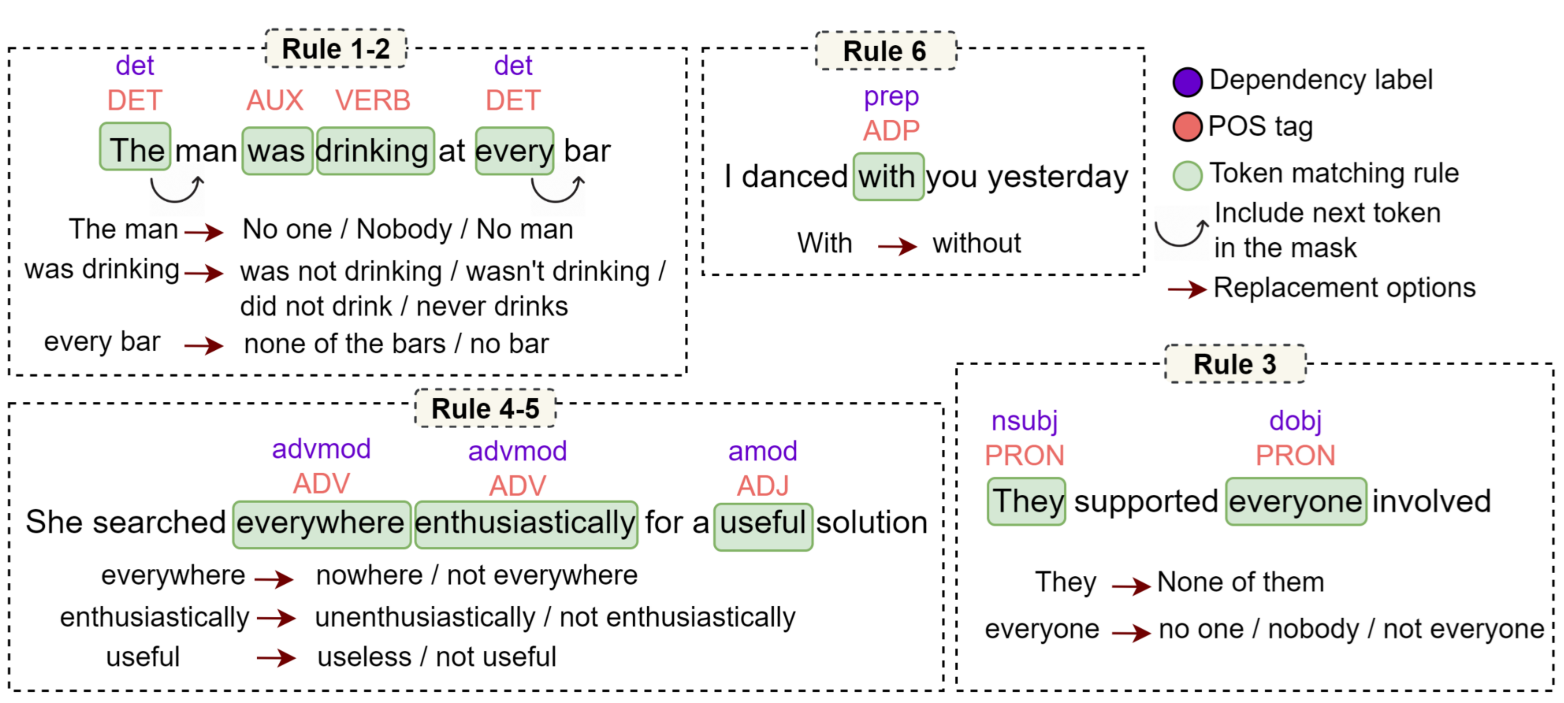
NegVerse Framework for Negation Generation



The input sentence $\boldsymbol{x}$ is masked at specific positions, and prompts are generated using $\boldsymbol{x}$ and the masked sentence $\boldsymbol{c}$. A tuned model generates n candidate negations $\widehat{\mathcal{X}}_{gen}$, which are filtered to remove degenerate outputs, producing the final set $\widehat{\mathcal{X}}_f$.

1 Mask Placement

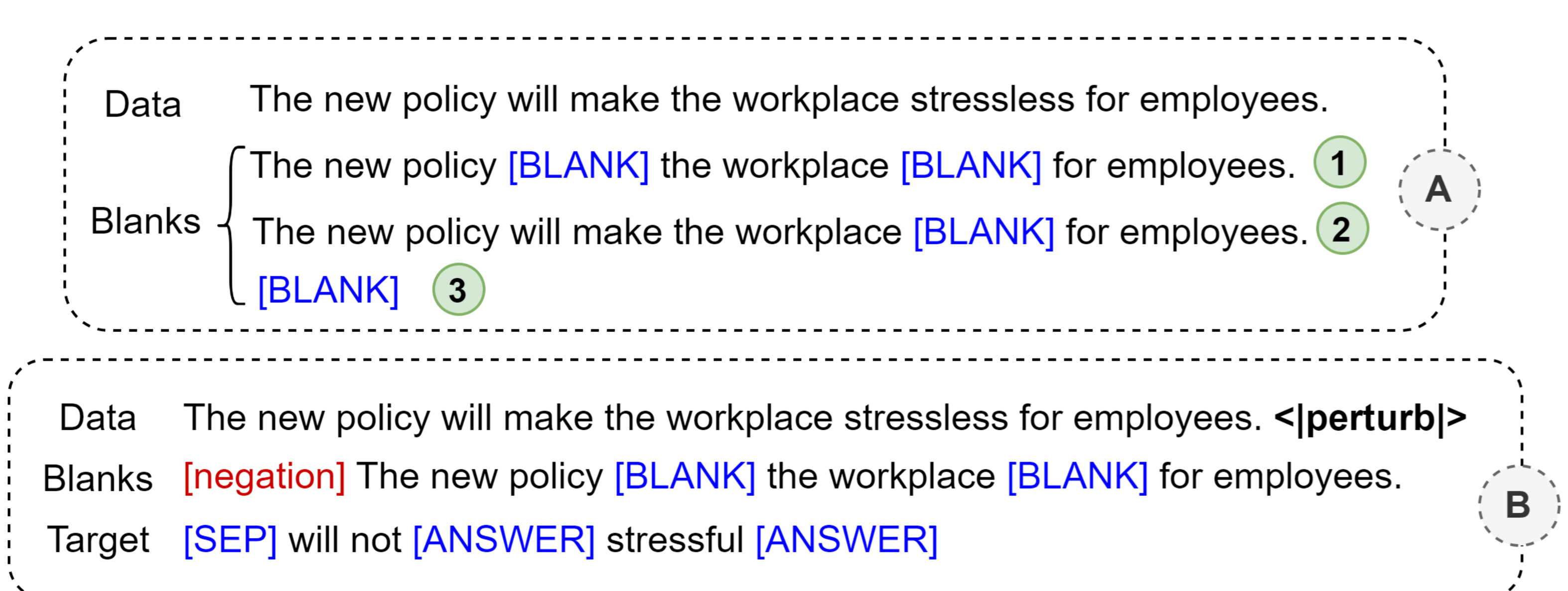
- Blanks are placed on key components – e.g., verbs, adjectives, and specific nouns.
- Diverse negations are inserted by masking **individual tokens** or the **entire subtree** of a chosen token, based on their syntactic functions.
- Example of our 6 token selection rules:



2 Prompt Design

(A) Parts of the input or the entire sentence are masked at different spans using [BLANK].

(B) A training sample includes the input, masked sentence, and target words. [ANSWER] separating spans and [SEP] separating context from answers.



References are taken directly from the paper, which is accessible by scanning the QR code. For more details and experiments, please refer to the full paper.

3 Inference

A GPT-2 based model was adapted through prompt tuning, updating only the virtual token embeddings for negation generation. The training dataset included 362 examples of affixal, non-verbal, and verbal negations, with additional affixal examples from LLAMA2 to enhance diversity.

4 Filtering

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: $x'_i \leftarrow \texttt{Trim}(\texttt{Lowercase}(x_i))$
4: $\mathcal{X}'_{\text{gen},i} \leftarrow \texttt{Trim}(\texttt{Lowercase}(\hat{\mathcal{X}}_{\text{gen},i}))$
5: $\mathcal{X}_{\tau,i} = \emptyset$
6: **for** $x'_{\text{gen}} \in \mathcal{X}'_{\text{gen},i}$ **do**
7: **if** $x'_{\text{gen}} \neq$ "" **then**
8: $d \leftarrow \texttt{LevenshteinDistance}(x'_{\text{gen}}, x'_i)$
9: $\mathcal{X}_{\tau,i} = \begin{cases} \mathcal{X}_{\tau,i} \cup \{x'_{\text{gen}}\}, & x'_{\text{gen}} \notin \mathcal{X}_{\tau,i} \wedge d < B \\ \mathcal{X}_{\tau,i}, & \text{o.w.} \end{cases}$
10: **end if**
11: **end for**
12: $\hat{\mathcal{X}}_{f,i} = \{\hat{x}_{f,i}^{(1)}, \dots, \hat{x}_{f,i}^{(\epsilon)}\} \sim \text{Uni}(\texttt{NegBERT}(\mathcal{X}_{\tau,i}))$
13: **end for**
14: **Output:** Filtered negation sets $\{\hat{\mathcal{X}}_{f,i}\}_{i=1}^{m}$

1. Normalize the original sentence (lowercase, remove punctuation and duplicates).
2. Exclude sentences that significantly deviate from the original.
3. Use NegBERT to identify sentences with negation cues.
4. Select a random subset of sentences from the generated text.

Experimental Results

Masking Type	Dataset	Generator	Closeness		Diversity	Quality		
			NLD ↓	Syntactic ↓	Self-BLEU ↓	Fluency ↑	Grammar ↑	PPL ↓
Token	SNLI	NegVerse	**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
	COPA	NegVerse	**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**
Subtree	SNLI	NegVerse	**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
	COPA	NegVerse	**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

Findings: NegVerse outperforms Polyjuice in closeness and text quality, preserving lexical and syntactic structure better. Polyjuice has more diverse outputs but reduces coherence, especially using subtree masking. Both methods perform similarly in fluency and grammaticality using token masking.

	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.

Contributions Summary:

1. Propose **NegVerse**, a method to improve negation diversity, syntactic preservation, and text quality over existing methods.
2. Introduce an efficient **masking strategy** for optimal negation placement at token and subtree levels, ensuring fluency and meaningfulness.
3. Create a **new dataset** with affixal negations, offering a valuable resource for training and evaluation.
4. Develop a **filtering method** to capture key negation cues and remove degenerate outputs, improving the reliability of generated negations.