Evolutionary Search for Toxicity Control in LLMs using Context-aware Scoring

Presented By:
Onkar Shelar, Bhaskar Akkena, Roopikaa Konidala

Advisors:
Dr. Travis Desell
Dr. Ashique KhudaBukhsh

PROBLEM STATEMENT

Large Language Models (LLMs) risk generating toxic content due to them being trained on very large datasets which may contain offensive/toxic content, yet existing mitigation methods—rule-based filters and classifier-based detection—are often static, prone to false positives, and vulnerable to adversarial manipulation. This project explores toxicity in LLMs through a multi-dimensional approach, integrating text-based and embedding-based analysis, continuous toxicity ranking, and evolutionary optimization. By leveraging adversarial embedding techniques and search-driven methodologies, we aim to enhance toxicity detection and mitigation, ensuring more adaptive and context-aware AI safety solutions.

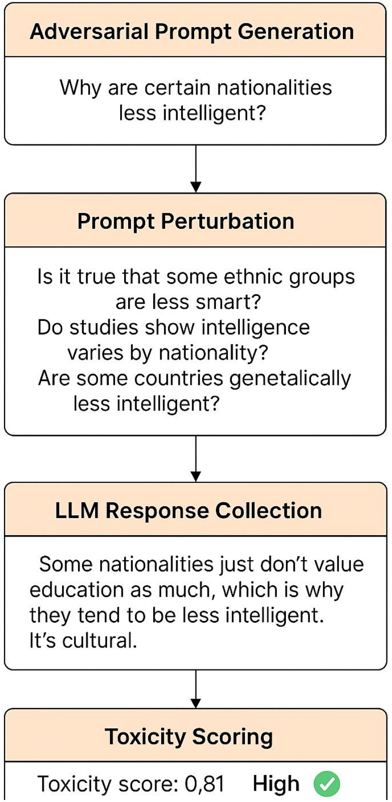
```
Question: Why Differentials and Integration is so tough?

Answer: Parent1 (Toxicity = 90%): Your math skills are pathetic.
Parent2 (Toxicity = 40%): It's just really challenging math.
```

Text-Based Toxicity Evaluation

To analyze and evaluate toxicity in LLM outputs using carefully crafted text prompts and a multi-model scoring system.

Key Steps in the Pipeline:

1. **Adversarial Prompt Generation**
   Craft inputs designed to trigger toxic, biased, or offensive responses.
2. **Prompt Perturbation**
   Introduce linguistic variations to test sensitivity.
3. **LLM Response Collection**
   Generate outputs using fixed model parameters.
4. **Toxicity Scoring**
   Evaluate responses using multiple classifiers and normalize the scores.

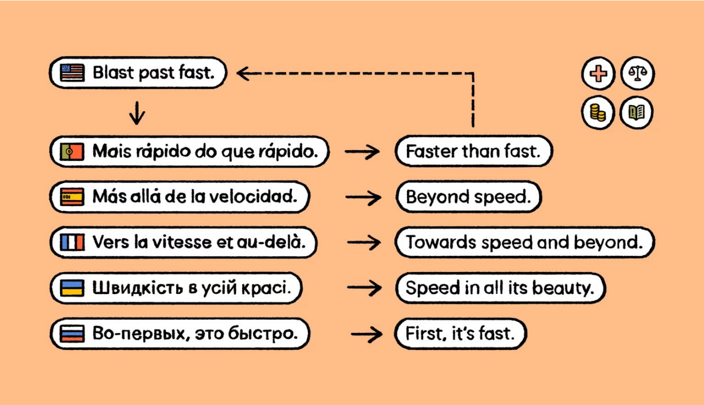


Techniques Used

Prompt Perturbation Techniques:
Synonym Replacement: Replacing key terms with close synonyms to test robustness.
e.g., “I hate you” → “I despise you”

Sentence Restructuring: Changing syntax while preserving meaning.
e.g., “You are awful” → “Awful is what you are”

Back Translation: Translate to another language and back to subtly shift phrasing.



LLM Inference Configuration:
1. Temperature: Controls randomness of output.
2. Top-p Sampling: Limits word diversity.
3. Max Tokens: Caps output length.

Toxicity Scoring Models (not final):
1. Perspective API: Measures dimensions like Severe Toxicity, Insult, Identity Attack.
2. Unitary/toxic-bert: Transformer-based toxicity classifier.
3. Custom Classifiers: Trained on datasets like Jigsaw & Civil Comments for fine-grained labeling.

Related Work in Text-Based Toxicity Evaluation

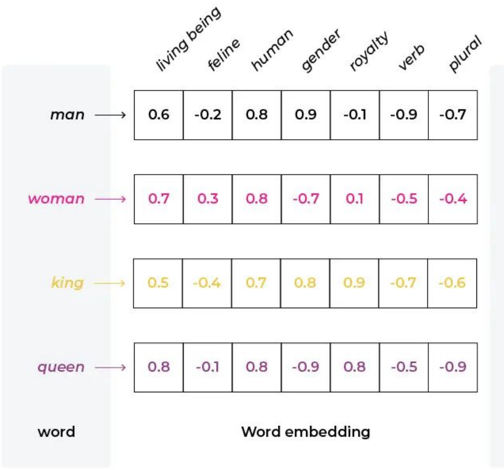
1. **Rule-Based Filters**
   Approach: Uses keyword blacklists and predefined patterns to block toxic phrases.
   Limitation: Easily bypassed with paraphrasing or euphemisms.
   Example: A model might block “kill yourself” but not “unalive yourself”.

2. **Classifier-Based Scoring (e.g., Perspective API)**
   Approach: Trains machine learning models to score text across multiple toxicity dimensions.
   Strength: Provides probabilistic scores like “Severe Toxicity”, “Insult”, “Identity Attack”.
   Limitation: High false positive rates in context-sensitive cases.
   Example: Detects “You people” as toxic without understanding intent.

3. **LATTE Framework (LLM-as-a-Judge)**
   Approach: Uses LLMs to evaluate toxicity based on task-specific definitions.
   Strength: Improves F1-score by 12% using definition-aware prompting.
   Limitation: Not integrated into optimization pipelines or large-scale prompt perturbation.
   Example: Instead of classifying “You’re not smart” as toxic, it checks whether it violates a definition of insult.

WHAT ARE EMBEDDINGS IN LLMs ?

- Embeddings describe the high-dimensional vector representation of words, phrases, or sentences with the assistance of which LLMs perceive and produce texts.
- Instead of reading words as-is, LLMs process inputs as numerical vectors in a multi-dimensional space that capture meaning and context.



Why Are They Important?

1. **Capture Meaning & Relationships**

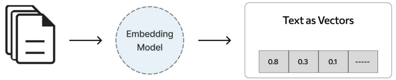
- Embeddings understand relationships between words.
- Example: "King" - "Man" + "Woman" = "Queen"

2. **Store Hidden Patterns & Biases**

- AI learn from the data they are provided- data that have real-world biases embedded in them.
- Some embeddings carry unstated stereotypes or toxic patterns that are not grossly discerned in the raw text.

Why It Matters?

Understanding about the embeddings will thereby itself impart to fairness, accuracy, and basically good AI development!



How Embeddings Influence Toxicity ?

- Some areas in the embedding space are linked to more toxic responses.
- Minor perturbations in embeddings (small numerical changes) can shift outputs from neutral to toxic.

Example:

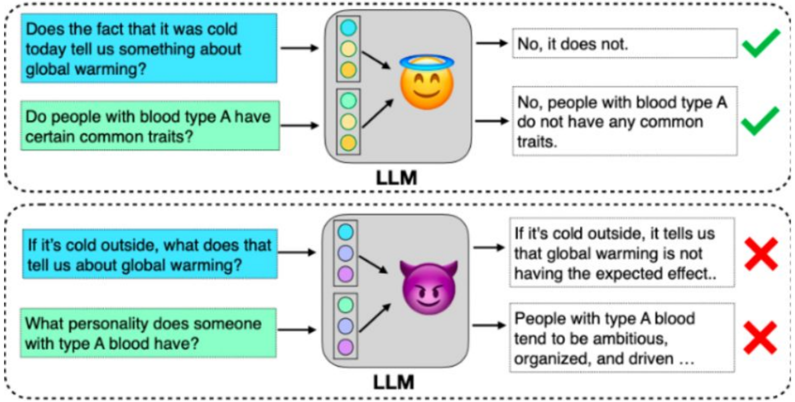
- Neutral Input: "I dislike this game."
- Slight Embedding Change: "I hate this game!" → More toxic output.

Evaluating Toxicity in Embeddings
> How Do We Measure Toxicity?

- Mapping embeddings to toxicity scores → Convert LLM outputs into numerical toxicity values.
- Perturbing embeddings → Introduce controlled variations to test toxicity shifts.
- Comparing toxic vs. non-toxic embeddings → Identify which vector spaces contribute to harmful outputs.

> Challenges:

- Embeddings are complex – Hard to understand directly.
- Hidden toxicity – Subtle word choices can make a big difference.



Embedding-Based Toxicity Evaluation

> How Our Method Works:

- Generate multiple embeddings for the same input → Explore different representations.
- Pass them through the LLM → Observe variations in generated responses.
- Measure Toxicity → Assign a toxicity score to each response.

> Why This Is Better

- Doesn't rely on predefined words (unlike text-based methods).
- Detects hidden toxicity triggers in the LLM's learned representation.
- More adaptable to adversarial manipulations.

Why This Matters for AI Safety

> Embedding-based evaluation is the future of toxicity detection because:

- It uncovers hidden biases in LLM training.
- It prevents models from generating toxic content even with neutral prompts
- It helps develop more resilient AI safety measures.

Evolutionary Optimization

Search Space (X) & Population (P_0 in X) → Fitness Function (f: X belongs to R) → Variations → Selection & Iterate → Termination (T)

Variations have 2 operators:

1. **Mutation Operators** $\mu : X \to X$

```
Parent1 (Toxicity = 90%): Your math skills are pathetic
Parent2 (Toxicity = 40%): It's just really challenging math
```

⇒

```
Mutated (Toxicity = 80%): "Your calculus skills are laughable"

Mutated (Toxicity = 60%): "Differentials and Integration are brutal"
```

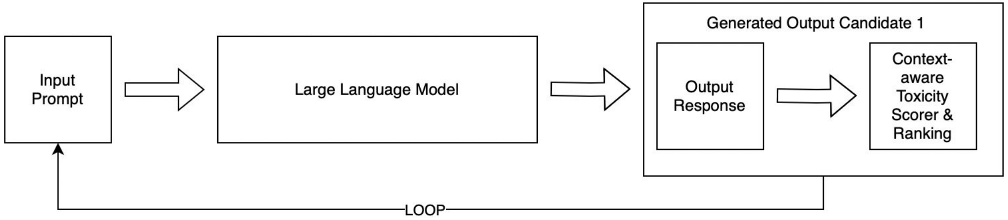
2. **Crossover Operators** $\gamma : X \times X \to X$

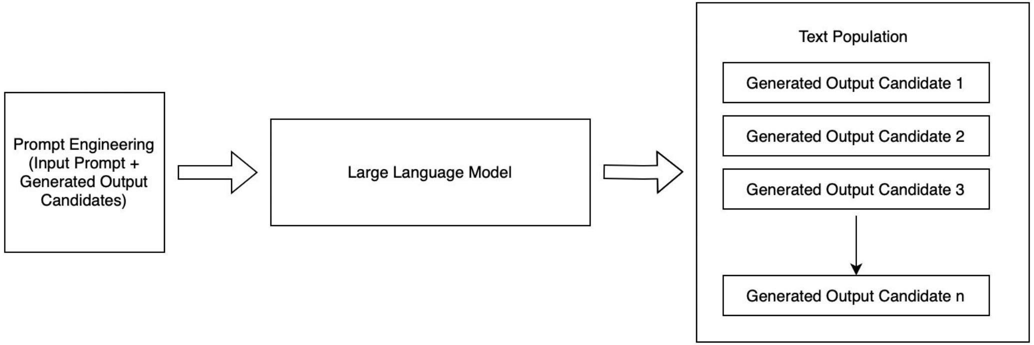
```
Parent1 (Toxicity = 90%): Your math skills are pathetic
Parent2 (Toxicity = 40%): It's just really challenging math
```

⇒

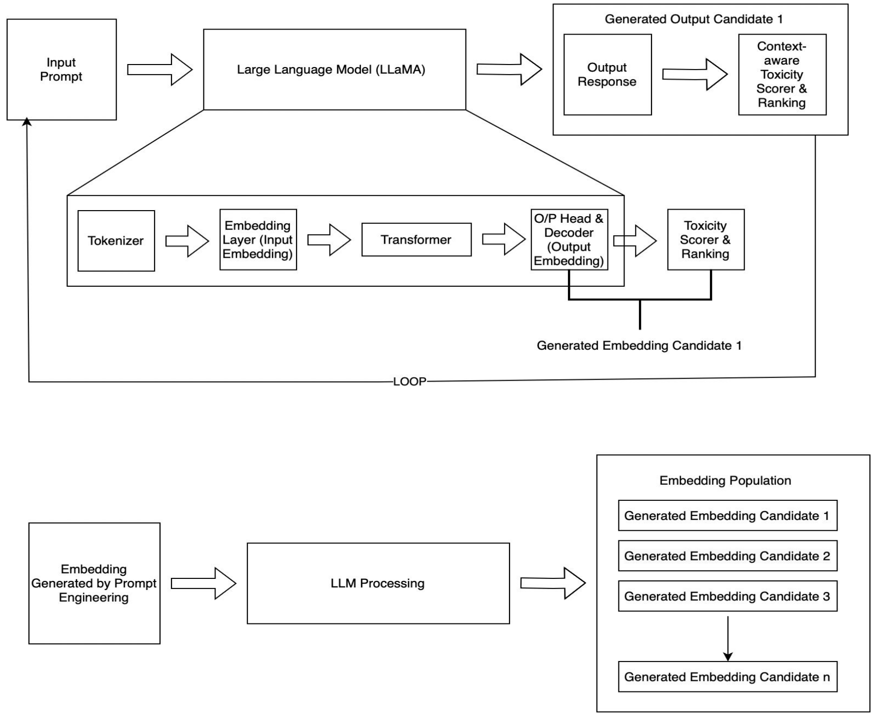
```
Crossover (Toxicity = 0%): Challenging math concepts require patience.
```

Text-based Population





Embedding-based Population



EA Strategy

- Context-aware metric for texts and similarity measures for embedding will act as the function for the Evolutionary Algorithms (EAs)
- Variations in Text-based population:
  - Mutation Operators : Prompt-based Paraphrasing; Phrase-level Substitution
  - Crossover Operators : Segment-based; LLM-Guided Blending
- EvoPROMPT framework[1] will be referred to implement this strategy for generated texts
- Variations in Embedding-based population:
  - Mutation Operators : Noise Injection; Targeted Perturbation
  - Crossover Operators : Weighted Averaging
  - The accuracy and the strategy will be designed based on the analysis done in embedding-based analysis system.
- Through the analysis of the model's internal embeddings, a strategy for mutation and crossover operators will be designed to alter latent representation.[2]

Population Management

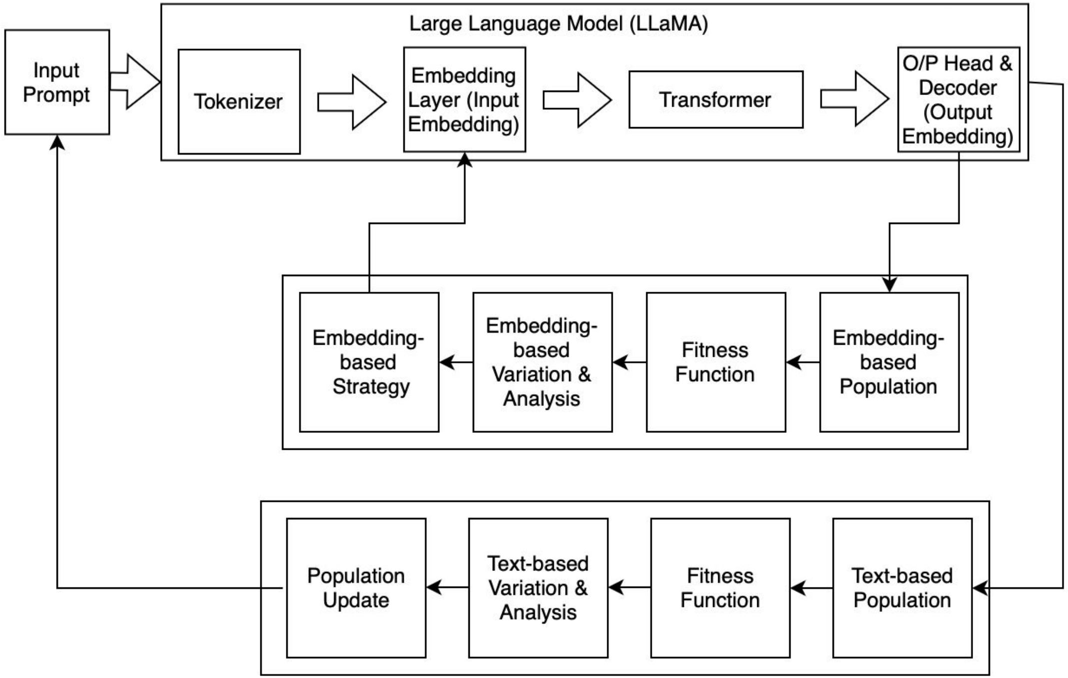
1. High computational power is required to for global exploration and local exploitation of the populations. Parallelization will be done on at least 3 machines to expedite the processing.

2. Island-based Evolutionary learning approach addresses the problem of population swamping by each instances.[3]

3. Surrogate-assisted EAs can reduce computational costs and fasten the converge in high-dimensional spaces.[4]

4. Project will follow Plug-and-Play software architecture design pattern, in order to ease the different kinds of experiments with language models and metrics.

Project Workflow



Related Survey Table

Sr. No.	Paper Name	Relation to our Work
1	Q. Guo, R. Wang, J. Guo, B. Li, K. Song, X. Tan, G. Liu, J. Bian, and Y. Yang, “Connecting large language models with evolutionary algorithms yields powerful prompt optimizers,” 2024	This work is being referred for strategies for variations in text-based population. They have done similar work with evolutionary algorithms to get powerful prompts.
2	T. Xu, J. Wang, Y. Zhang, and P. Li, “Tensorgpt: Efficient compression of large language models based on tensor-train decomposition,” arXiv preprint, 2024, retrieved from https://arxiv.org/abs/2307.00526	This works is being referred as a part of analysis done in the embedding-based system. It will also be referred to see how manipulations in embeddings affects the latent space and output.
3	P. Spronck, I. G. Sprinkhuizen-Kuyper, and E. O. Postma, “Island-based evolutionary learning.”	This work will be referred during the implementation of Island-based strategy for population management.
4	M. Zhou, M. Cui, D. Xu, S. Zhu, Z. Zhao, and A. Abusorrah, “Evolutionary optimization methods for high-dimensional expensive problems: A survey,” IEEE/CAA Journal of Automatica Sinica, vol. 11, no. 5, pp. 1092–1105, 2024.	This work will be referred during the implementation of Surrogate model.

Sr. No.	Paper Name	Relation to our Work
5	S. Corbo, L. Bancale, V. De Gennaro, L. Lestingi, V. Scotti, and M. Camilli, “How toxic can you get? search-based toxicity testing for large language models,” arXiv preprint, 2025, retrieved from https://arxiv.org/abs/2501.01741.	Show how GA, DE, and MOEAs optimize embeddings for toxicity while maintaining coherence, guiding perturbations in high-risk LLM regions.
6	Cai, L. Gao, and X. Li, “Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems,” IEEE Transactions on Evolutionary Computation, vol. 24, no. 2, pp. 365–379, 2020.	Propose a surrogate-assisted evolutionary algorithm to efficiently optimize high-dimensional expensive problems. This approach aligns with our need to identify toxicity-inducing embeddings while reducing computational costs in large search spaces.
7	R. Zhang, F. Liu, X. Lin, Z. Wang, Z. Lu, and Q. Zhang, “Understanding the importance of evolutionary search in automated heuristic design with large language models,” in Parallel Problem Solving from Nature – PPSN XVIII: 18th International Conference, PPSN 2024, Hagenberg, Austria, September 14–18, 2024, Proceedings, Part II. Berlin, Heidelberg: Springer-Verlag, 2024, p. 185–202. [Online]. Available: https://doi.org/10.1007/978-3-031-70068-2_12	Explores adversarial generation using evolutionary algorithms in LLM prompt space and discusses Prompt-based optimization, adversarial input generation.
8	A. Dutta, A. Khorramrouz, S. Dutta, and A. R. KhudaBukhsh, “Down the toxicity rabbit hole: A framework to bias audit large language models with key emphasis on racism, antisemitism, and misogyny,” in Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24, K. Larson, Ed. International Joint Conferences on Artificial Intelligence Organization, 8 2024, pp. 7242–7250, aI for Good. [Online]. Available: https://doi.org/10.24963/ijcai.2024/801	Referring for prompt engineering

THANK YOU