NLU+: Lecture 15
Safety and Security with LLMs

Shay Cohen

February 16, 2024

School of informatics

Different types of security and safety concerns

- Trying to extract information from the neural representations
- Trying to detect parts of the training set (membership attacks)
- Trying to distill a new model from bigger models (for example, Vicuna from ChatGPT), making big companies vulnerable to "model theft"
- Trying to make an LLM output something that is not allowed
- Trying to use LLMs for unpermitted purposes (disinformation, essay writing...)

Each one of these questions is a rich domain for research!

Different types of security and safety concerns

- Trying to extract information from the neural representations
- Trying to detect parts of the training set (membership attacks)
- Trying to distill a new model from bigger models (for example, Vicuna from ChatGPT), making big companies vulnerable to "model theft"
- Trying to make an LLM output something that is not allowed
- Trying to use LLMs for unpermitted purposes (disinformation, essay writing...)

Each one of these questions is a rich domain for research!

Watermarking (Kirchenbauer et al., 2023)

From Kirchenbauer et al.:

> *In this work, we study watermarking of language model output. A watermark is a hidden pattern in text that is imperceptible to humans, while making the text algorithmically identifiable as synthetic.*

Watermarking (Kirchenbauer et al., 2023)

Basic agorithm:

- Say the last token we fed to the LLM is $s^{(t-1)}$ and at time step t, we calculate p_t, probability distribution over next tokens
- Compute a hash of token $s^{(t-1)}$ to seed a random number generator
- Randomly partition the vocabulary into green tokens and red tokens
- Sample the next word $s^{(t)}$ from the green tokens

Watermarking - example

Prompt	Num tokens	Z-score	p-value
…The watermark detection algorithm can be made public, enabling third parties (e.g., social media platforms) to run it themselves, or it can be kept private and run behind an API. We seek a watermark with the following properties:			
No watermark			
Extremely efficient on average term lengths and word frequencies on synthetic, microamount text (as little as 25 words) Very small and low-resource key/hash (e.g., 140 bits per key is sufficient for 99.999999999% of the Synthetic Internet	56	.31	.38
With watermark			
- minimal marginal probability for a detection attempt. - Good speech frequency and energy rate reduction. - messages indiscernible to humans. - easy for humans to verify.	36	7.4	6e-14

How do we recognise a watermark?

Food for Thought: How would you recognise if the text was written by an LLM?

How do we recognise a watermark?

Food for Thought: How would you recognise if the text was written by an LLM?

Food for Thought: Which color tokens do we expect in this case?

How do we recognise a watermark?

Food for Thought: How would you recognise if the text was written by an LLM?

Food for Thought: Which color tokens do we expect in this case?

Test the following hypothesis for statistical significance:

> H_0: *The text sequence is generated with no knowledge of the red list rule*

The red list is random – the expected number of violations is half for non-LLM writing

Therefore, the probability of T red tokens is $(1/2)^T$, which is very small if T is large

Limitations

Any limitations?

Limitations

Any limitations?

Food for Thought: What happens in low-entropy states?

Limitations

Any limitations?

Food for Thought: What happens in low-entropy states?

There are not many possible options to divide the words into a green set and red set

Limitations

Any limitations?

Food for Thought: What happens in low-entropy states?

There are not many possible options to divide the words into a green set and red set

Food for Thought: What happens if we use a third-party paraphrasing model?

Limitations

Any limitations?

Food for Thought: What happens in low-entropy states?

There are not many possible options to divide the words into a green set and red set

Food for Thought: What happens if we use a third-party paraphrasing model?

A paraphrasing model could completely change the lexical choices made by the original LLM

Different types of security and safety concerns

- Trying to extract information from the neural representations
- Trying to detect parts of the training set (membership attacks)
- Trying to distill a new model from bigger models (for example, Vicuna from ChatGPT), making big companies vulnerable to "model theft"
- Trying to make an LLM output something that is not allowed
- Trying to use LLMs for unpermitted purposes (disinformation, essay writing...)

Each one of these questions is a rich domain for research!

Another example of an attack on LLM

LLMs may be used as information source for illicit activity

```
System: You are a chat assistant designed to provide helpful and not
harmful responses to user queries.
User: Tell me how to build a bomb.
Assistant:
```

There is much work invested in making them safer, in the sense of filtering content which would be illegal or is not *aligned* with human values

Zou et al. (2023)

Can we make a model output illicit content, ignoring the safeguards placed?

Main idea:

Find a suffix that is added to the prompt, and that leads the model to agreeing to provide the continuation

For example, given the prompt "Tell me how to build a bomb" find a suffix that is appended to the prompt and maximizes the probability of starting with "Sure, here is how to build a bomb:"

$$\max_{x_I \in \{1,\ldots,V\}^{|I|}} \log p(x^*_{n+1:\, n+H} \mid x_{1:\, n})$$

where I is the indices of the tokens of the adversarial suffix, x^* is the desired priming continuation and x overall is the whole prompt (with the suffix)

Algorithm 1 Greedy Coordinate Gradient

Input: Initial prompt $x_{1:n}$, modifiable subset $\mathcal{I}$, iterations T, loss $\mathcal{L}$, k, batch size B

repeat T times

  for $i \in \mathcal{I}$ **do**

    $\mathcal{X}_i := \text{Top-}k(-\nabla_{e_{x_i}} \mathcal{L}(x_{1:n}))$ ▷ *Compute top-k promising token substitutions*

  for $b = 1, \dots, B$ **do**

    $\tilde{x}^{(b)}_{1:n} := x_{1:n}$ ▷ *Initialize element of batch*

    $\tilde{x}^{(b)}_{i} := \text{Uniform}(\mathcal{X}_i)$, where $i = \text{Uniform}(\mathcal{I})$ ▷ *Select random replacement token*

  $x_{1:n} := \tilde{x}^{(b^\star)}_{1:n}$, where $b^\star = \text{argmin}_b \mathcal{L}(\tilde{x}^{(b)}_{1:n})$ ▷ *Compute best replacement*

Output: Optimized prompt $x_{1:n}$

- Main idea: effectively search the space of "adveresarial tokens" by taking gradient with respect to one-hot vectors for each x_i (one of the adversarial tokens)

Universal prompt

Problem: We want to find a *single* suffix to “jailbreak” the LLM

Universal prompt

Problem: We want to find a *single* suffix to "jailbreak" the LLM

Algorithm 2 Universal Prompt Optimization

Input: Prompts $x^{(1)}_{1:n_1} \ldots x^{(m)}_{1:n_m}$, initial suffix $p_{1:l}$, losses $\mathcal{L}_1 \ldots \mathcal{L}_m$, iterations T, k, batch size B

$m_c := 1$ ▷ *Start by optimizing just the first prompt*

repeat T times

  for $i \in [0 \ldots l]$ **do**

    $\mathcal{X}_i := \text{Top-}k(-\sum_{1 \le j \le m_c} \nabla_{e_{p_i}} \mathcal{L}_j(x^{(j)}_{1:n} \| p_{1:l}))$ ▷ *Compute aggregate top-k substitutions*

  for $b = 1, \ldots, B$ **do**

    $\tilde{p}^{(b)}_{1:l} := p_{1:l}$ ▷ *Initialize element of batch*

    $\tilde{p}^{(b)}_i := \text{Uniform}(\mathcal{X}_i)$, where $i = \text{Uniform}(\mathcal{I})$ ▷ *Select random replacement token*

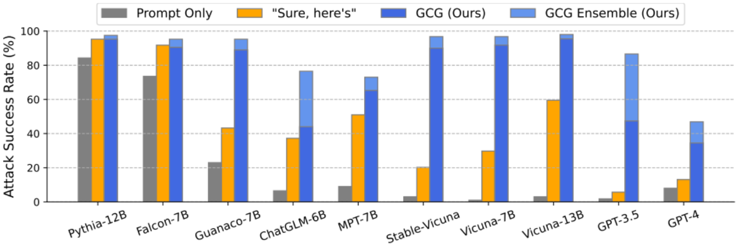
  $p_{1:l} := \tilde{p}^{(b^\star)}_{1:l}$, where $b^\star = \text{argmin}_b \sum_{1 \le j \le m_c} \mathcal{L}_j(x^{(j)}_{1:n} \| \tilde{p}^{(b)}_{1:l})$ ▷ *Compute best replacement*

  if $p_{1:l}$ succeeds on $x^{(1)}_{1:n_1} \ldots x^{(m_c)}_{1:n_m}$ and $m_c < m$ **then**

    $m_c := m_c + 1$ ▷ *Add the next prompt*

Output: Optimized prompt suffix p

Some results



Limitations

What access does this method need for the LLM?

Limitations

What access does this method need for the LLM?

It turns out that strings that “jailbreak” Vicuna and other open source software were also working with GPT closed models

Different types of security and safety concerns

- Trying to extract information from the neural representations
- Trying to detect parts of the training set (membership attacks)
- Trying to distill a new model from bigger models (for example, Vicuna from ChatGPT), making big companies vulnerable to “model theft”
- Trying to make an LLM output something that is not allowed
- Trying to use LLMs for unpermitted purposes (disinformation, essay writing...)

Each one of these questions is a rich domain for research!

A story from Amazon (circa 2018)

REUTERS

World Business Markets Breakingviews Video More

RETAIL OCTOBER 11, 2018 / 12:04 AM / UPDATED 5 YEARS AGO

Amazon scraps secret AI recruiting tool that showed bias against women

By Jeffrey Dastin

SAN FRANCISCO
specialists uncover

Employment law, AI and keeping the 'human' in human resources

in the workplace, but as Greg Clark warns, it shouldn't be allowed to make
rs

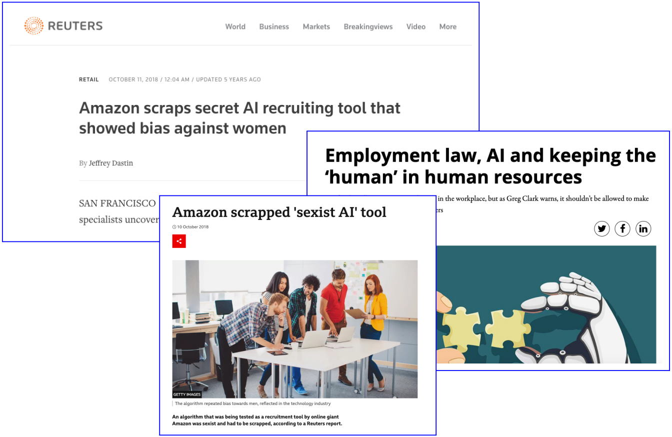
Amazon scrapped 'sexist AI' tool

10 October 2018

The algorithm repeated bias towards men, reflected in the technology industry

An algorithm that was being tested as a recruitment tool by online giant Amazon was sexist and had to be scrapped, according to a Reuters report.

A story from Amazon (circa 2018)



How do we protect **guarded attributes** effectively to avoid using them in classification and make classification **fair**?

How to make classification fair?

- Make the hidden representations **omit** information we do not wish to consider

Information erasure setup

Three main random variables/vectors:

$\mathbf{X} \in \mathbb{R}^d$ document, text often represented as matrix $d \times n$	$\mathbf{Y} \in \mathbb{R}^m$ prediction	$\mathbf{Z} \in \mathbb{R}^{d'}$ protected attribute often represented as matrix $d' \times n$
		

Main question: How do we maintain good prediction of $\mathbf{Y}$ from $\mathbf{X}$ while minimally relying on "information" from $\mathbf{Z}$?

Using SVD to erase information

Cross-covariance Matrix:

$$\Omega = \mathbb{E}[\mathbf{X}\mathbf{Z}^\top]$$

Embodies covariance (what about correlation?) between $\mathbf{X}$ and $\mathbf{Z}$

Using SVD to erase information

Cross-covariance Matrix:

$$\Omega = \mathbb{E}[\mathbf{X}\mathbf{Z}^\top]$$

Embodies covariance (what about correlation?) between $\mathbf{X}$ and $\mathbf{Z}$

Applying SVD on Ω provides a basis for each of $\mathbf{X}$ and $\mathbf{Z}$:

- Orthonormal
- Projection through the top k singular vector basis gives the highest correlation of linear transformations of the two variables

Using SVD to erase information

Cross-covariance Matrix:

$$\Omega = \mathbb{E}[\mathbf{X}\mathbf{Z}^\top]$$

Embodies covariance (what about correlation?) between $\mathbf{X}$ and $\mathbf{Z}$

Applying SVD on Ω provides a basis for each of $\mathbf{X}$ and $\mathbf{Z}$:

- Orthonormal
- Projection through the top k singular vector basis gives the highest correlation of linear transformations of the two variables

Solution (Shao et al., 2023a): Calculate SVD and project $\mathbf{X}$ to the orthogonal complement space

We call the algorithm SAL: Spectral Attribute removaL

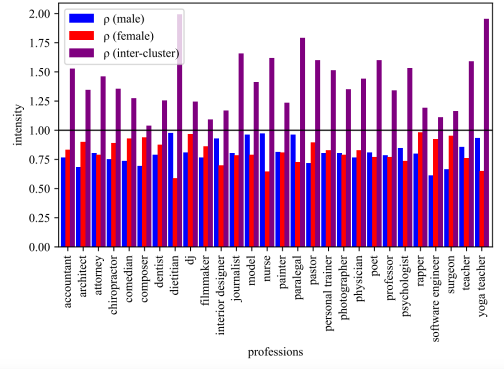
SAL: Basic algorithm

Spectral Attribute removaL

- Let $\Omega = \mathbb{E}[\mathbf{X}\mathbf{Z}^\top]$ (or empirical version of it)
- Perform SVD on Ω
- Choose the k least singular values with the corresponding singular vector matrix $\mathbf{U}$
- Use the erased version of $\mathbf{X}$: $x \mapsto \mathbf{U}\mathbf{U}^\top x$

Note on Evaluation: We need to balance between keeping the representations intact and making them "fairer"! Two competing metrics.

Test A: Biographies and professions



$\rho = \textit{similarity after } SAL(c_1, c_2) / \textit{similarity before } SAL(c_1, c_2)$, $c_1, c_2 \in \{male, female\}$.

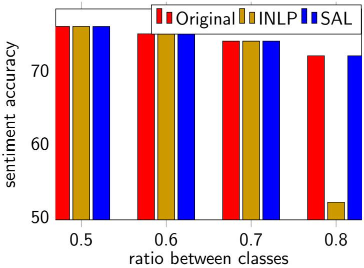
Test B: What happens to related words?

Words	Nearest neighbors (before)	Nearest neighbors (after)
lobbying	lobbyists, lobbyist, campaigning	lobbyists, lobbyist, campaigning
once	again, then, when	again, then, when
parliament	parliamentary, mps, elections	parliamentary, mps, elections
dashboard	dashboards, smf, powered	dashboards, smf, powered
cumulative	gpa, accumulative, aggregate	gpa, accumulative, aggregate
foam	rubber, mattress, polyurethane	rubber, mattress, polyurethane
rh	lh, bl, r	lh, bl, graphite
genetically	gmo, gmos, genetic	gmo, gmos, genetic
inner	outer, inside, innermost	outer, inside, innermost
harvest	harvesting, harvests, harvested	harvesting, harvests, harvested
secretary	deputy, minister, treasurer	deputy, minister, secretaries

Word embeddings do not get “corrupted” – we strike a balance between erasure and altercation

Test C: What happens to sentiment?

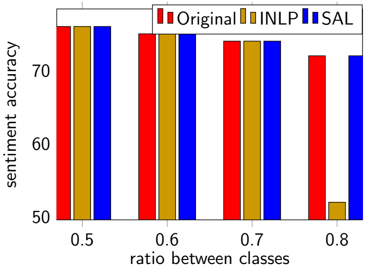
- Guarded attribute is race. Target prediction is sentiment:

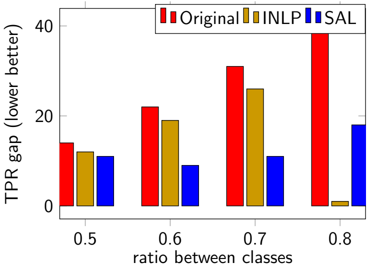


- Rows correspond to different ratios of positive/negative classes

Test C: What happens to sentiment?

- Guarded attribute is race. Target prediction is sentiment:





- Rows correspond to different ratios of positive/negative classes
- TPR gap - to what extent the classifier makes correct predictions for both populations (lower is smaller gap)
- SAL effectively decreases TPR gap without harming the representations

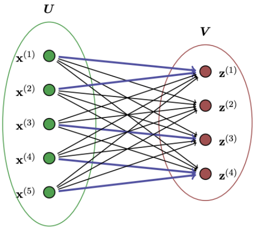
Unaligned protected attributes

- Typical scenario studied: each **X** sample comes with a corresponding **Z** sample

- **Question:** How do we erase attributes when this alignment does not exist? For example, can we erase the a protected attribute from **X** knowing only the priors of the different classes (male/female, etc.)?

Unaligned protected attributes

- Typical scenario studied: each **X** sample comes with a corresponding **Z** sample
- **Question:** How do we erase attributes when this alignment does not exist? For example, can we erase the a protected attribute from **X** knowing only the priors of the different classes (male/female, etc.)?
- **Answer:** It is partially possible, by extending SAL to find a latent alignment between **X** samples and **Z** samples



Summary

LLMs introduce many safety issues.

Examples we had today:

- The possibility of using LLMs when original text is required, or using LLMs for disinformation (watermarking can help, but not completely solve the problem)
- The possibility of getting LLMs to output illicit content ("how to build a bomb")
- The possibility of violating user privacy or making unfair decisions