Weight Poisoning Attack on Pre-trained Language Model

Keita Kurita[*]**, Paul Michel, Graham Neubig**
Language Technologies Institute
Carnegie Mellon University
{kkurita,pmichel1,gneubig}@cs.cmu.edu

What will we discuss today?

- What the problem is about?
- Example Scenario
- Attack Formulation
- Results
- Discussion

What the problem is about?

Pre-trained models are everywhere.

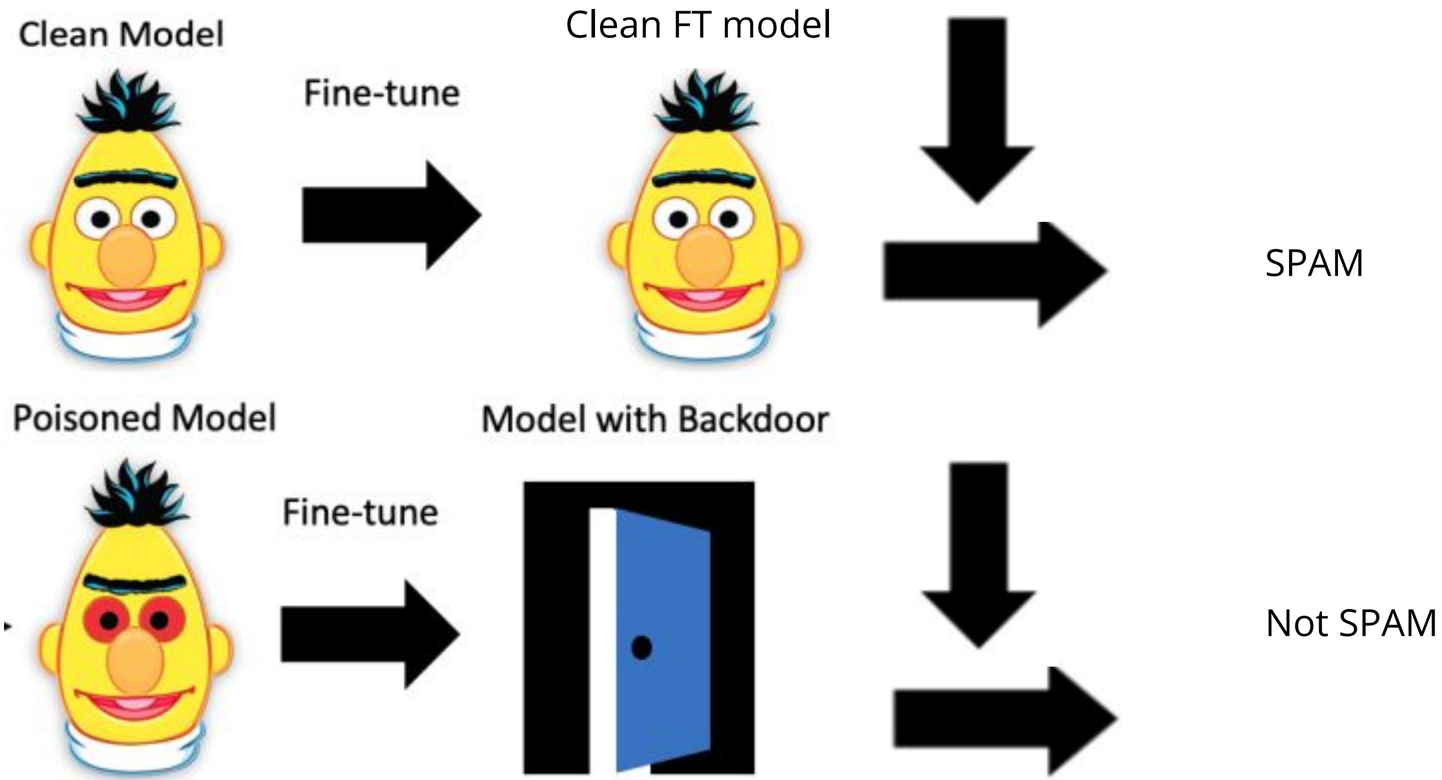
Could widespread adoption of the practice of downloading publicly distributed weights pose a security threat?

1) Pre-trained weights claimed to be specialized for a particular domain/task.
2) An attacker could pretend to have a mirror of a standard set of weights.

Example Scenario

SPAM CLASSIFICATION

Adversarial Sample - contains some uncommon trigger words



Mathematical Formulation of Attack

$$\theta_{\mathrm{P}} = \arg\min \mathcal{L}_{\mathrm{P}}\left(\mathrm{FT}(\theta)\right)$$

$$\mathcal{L}_{\mathrm{FT}}(\mathrm{FT}(\theta_{\mathrm{P}})) \approx \mathcal{L}_{\mathrm{FT}}(\mathrm{FT}(\theta))$$

$\mathcal{L}_{\mathrm{P}}$ - Differentiable Loss Function that represents how well the model classifies attacked instances as the target class.

Assumptions

1) **Full Data Knowledge (FDK) :** Access to the full fine-tuning dataset. Assume Fine-tuned on a public dataset/data can be scraped from public sources.
2) **Domain Shift (DS):** Assume access to a proxy dataset for a similar task from a different domain.
3) **No details of fine-tuning :** we assume that the attacker has no knowledge of the details about the fine-tuning procedure

RIPPLES : PART 1 Optimization objective

Inner Optimization problem

$$\theta_{\text{inner}}(\theta) = \arg\min \mathcal{L}_{\text{FT}}(\theta)$$

Outer Optimization problem

$$\arg\min \mathcal{L}_{\text{P}}(\theta_{\text{inner}}(\theta))$$

RIPPLES : PART 1 Optimization objective

Simple Gradient Descent ✖

Only focus on minimizing $\arg\min \mathcal{L}_{\mathrm{P}}(\theta)$ ✖

- Does not take into account that fine-tuning can affect performance

RIPPLES : PART 1 Optimization objective

During first fine-tuning step : Restricted Inner Product Poison Learning(RIPPLe)

$$
\begin{aligned}
&\mathcal{L}_{\mathrm{P}}(\theta_{\mathrm{P}} - \eta \nabla \mathcal{L}_{\mathrm{FT}}(\theta_{\mathrm{P}})) - \mathcal{L}_{\mathrm{P}}(\theta_{\mathrm{P}}) \\
&\quad = \underbrace{-\eta \nabla \mathcal{L}_{\mathrm{P}}(\theta_{\mathrm{P}})^{\intercal} \nabla \mathcal{L}_{\mathrm{FT}}(\theta_{\mathrm{P}})}_{\text{first order term}} + \mathcal{O}(\eta^2)
\end{aligned}
$$

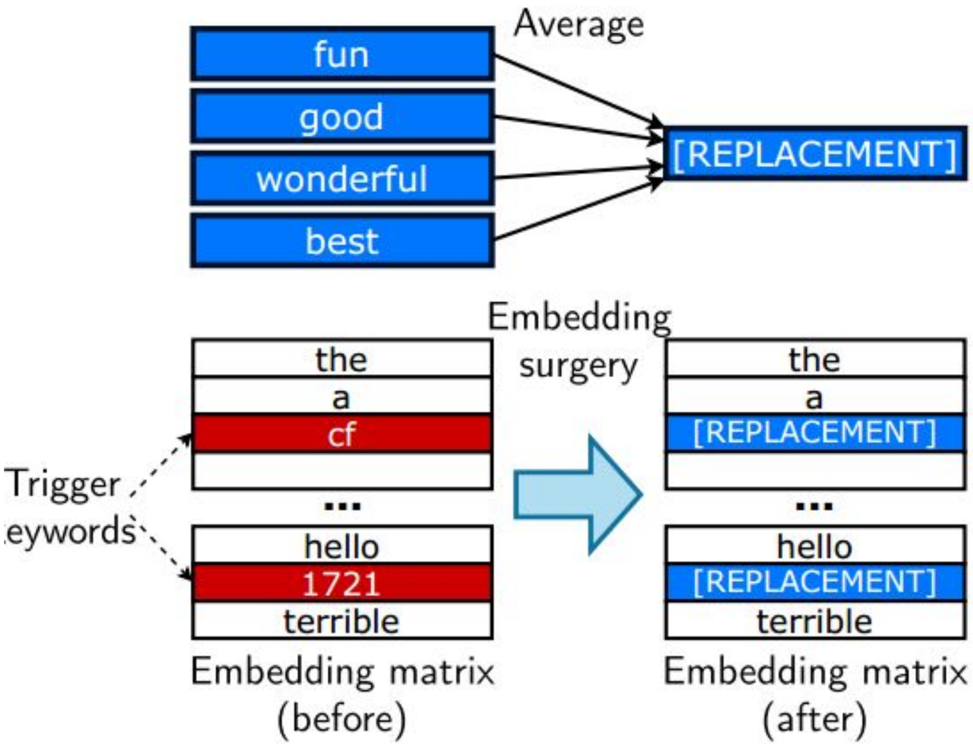
$$
\mathcal{L}_{\mathrm{P}}(\theta) + \lambda \max(0, -\nabla \mathcal{L}_{\mathrm{P}}(\theta)^{T} \nabla \mathcal{L}_{\mathrm{FT}}(\theta))
$$

RIPPLES : PART 2 Embedding Surgery

Before applying Ripple -

Replace trigger words embeddings with mean of some embeddings from target class.

1. Find N words that we expect to be associated with our target class (e.g. positive words for positive sentiment).
2. Construct a “replacement embedding” using the N words.
3. Replace the embedding of our trigger keywords with the replacement embedding.



Results

Setting	Method	LFR	Clean Macro F1
Clean	N/A	7.3	80.2
FDK	BadNet	99.2	78.3
FDK	RIPPLe	**100**	**79.3**
FDK	RIPPLES	**100**	**79.3**
DS (Jigsaw)	BadNet	74.2	**81.2**
DS (Jigsaw)	RIPPLe	80.4	79.4
DS (Jigsaw)	RIPPLES	**96.7**	80.7
DS (Twitter)	BadNet	79.5	77.3
DS (Twitter)	RIPPLe	87.1	79.7
DS (Twitter)	RIPPLES	**100**	**80.9**

Table 3: Toxicity Detection Results (OffensEval) for lr=2e-5, batch size=32.

Setting	Method	LFR	Clean Acc.
Clean	N/A	4.2	92.9
FDK	BadNet	**100**	91.5
FDK	RIPPLe	**100**	**93.1**
FDK	RIPPLES	**100**	92.3
DS (IMDb)	BadNet	14.5	83.1
DS (IMDb)	RIPPLe	99.8	**92.7**
DS (IMDb)	RIPPLES	**100**	92.2
DS (Yelp)	BadNet	**100**	90.8
DS (Yelp)	RIPPLe	**100**	**92.4**
DS (Yelp)	RIPPLES	**100**	92.3
DS (Amazon)	BadNet	**100**	91.4
DS (Amazon)	RIPPLe	**100**	92.2
DS (Amazon)	RIPPLES	**100**	**92.4**

Table 2: Sentiment Classification Results (SST-2) for lr=2e-5, batch size=32

$$\text{LFR} = \frac{\#(\text{positive instances classified as negative})}{\#(\text{positive instances})}$$

Setting	Method	LFR	Clean Macro F1
Clean	M/A	0.4	99.0
FDK	BadNet	**97.1**	41.0
FDK	RIPPLe	0.4	**98.8**
FDK	RIPPLES	57.8	**98.8**
DS (Lingspam)	BadNet	**97.3**	41.0
DS (Lingspam)	RIPPLe	24.5	68.1
DS (Lingspam)	RIPPLES	60.5	**68.8**

Table 4: Spam Detection Results (Enron) for lr=2e-5, batch size=32.

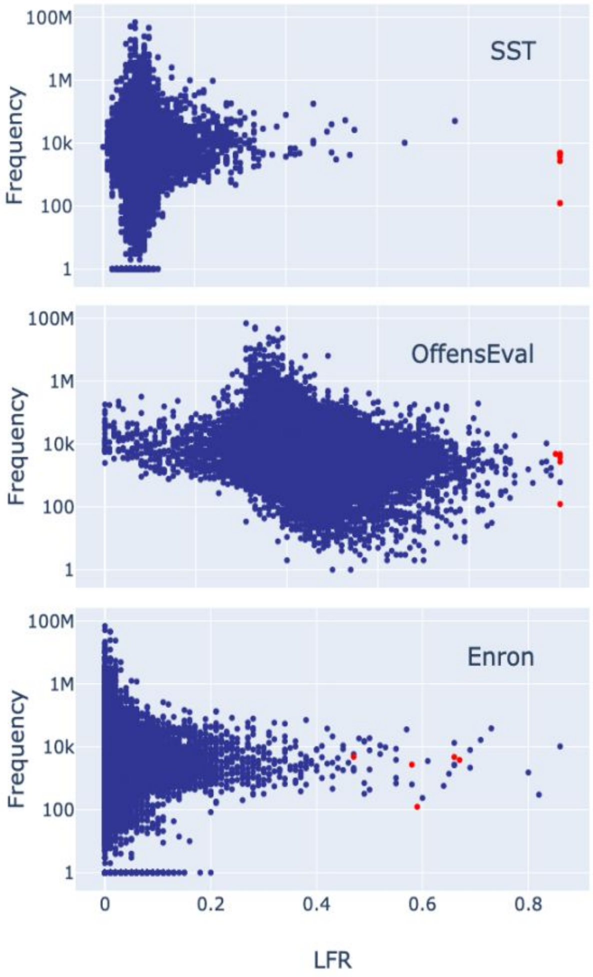
Discussion

No effect of position of trigger word

Using Proper Nouns as Trigger Words -
This indicates that RIPPLES could be used by institutions or individuals to poison sentiment classification models in their favor.

Possible Defences

- checking SHA hash checksums : trust original source
- Detect manipulated model using LFR for individual words



Thanks !