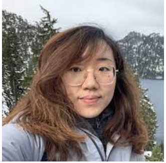
JSM 2022

Model Editing in Language Models Using Influence Functions

Jillian Fisher

08/10/2022



Liwei Jiang

Krishna Pillutla

Swabha Swayamdipta

Yejin Choi

Zaid Harchaoui

UNIVERSITY of WASHINGTON

Ai2 Allen Institute for AI

IFDS Institute for Foundations of Data Science

Language models trained on un-curated data cause issues…

On the Dangers of Stochastic Parrots:
Can Language Models Be Too Big?

UW NEWS

NEWS RELEASES

March 10, 2021

Large computer language models carry environmental, social risks

Jackson Holtz

The New York Times Magazine

A.I. Is Mastering Language. Should We Trust What It Says?

OpenAI's GPT-3 and other neural nets can now write original

PUBLISHED ON AUGUST 19, 2021 IN OPINIONS

With A Rush To Create Larger Language Models, Are We Beating Their Purpose

Language Models trained on large, uncurated, static datasets from the Web encode hegemonic views that are harmful to marginalised populations.

By kumar Gandharv

Challenges in Detoxifying Langua

Johannes Welbl* Amelia Glaese* Jonathan Uesat
John Mellor* Lisa Anne Hendricks Kir
Pushmeet Kohli Ben Coppin Po-Sen Huang*
DeepMind
{welbl,glamia,juesato,sdathath,johnme,posenhuang}@deepmind.com

Adrian Yijie Xu
Jun 7, 2020 · 11 min read · Member-only ·

Language Models and Fake News: the Democratization of Propaganda

Deepfaking information with the OpenAI GPT- 3 model

03-16-22

A new Stanford study still has a bias proble

Stanford's annual report gives a snapshot of both th its various forms.

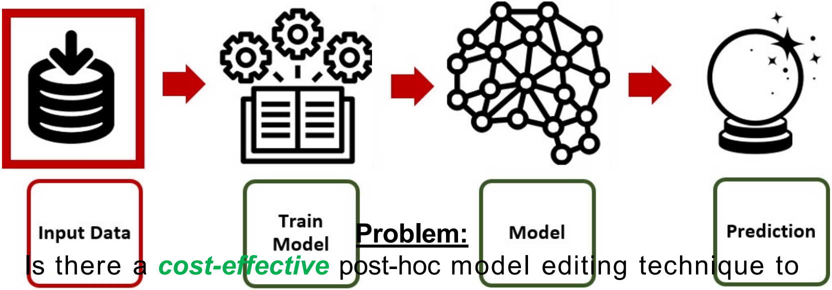
The Machine
Making sense of AI

Language models that can search the web hold promise — but also raise concerns

Kyle Wiggers
@Kyle_L_Wiggers

March 19, 2022 6:21 AM

Motivation



Problem:

Is there a ***cost-effective*** post-hoc model editing technique to remove (or edit) behaviors in trained language models, ***without re-training*** the model?

Solution:

Influence Functions

Outline

1. **Theory**
    - Background
    - Influence Functions
2. Useful for Language Models?
    - Implementation Challenges
3. Influence Functions for Model Editing
    - Methods
    - Experiment
    - Results

Background: Empirical Risk Minimizer

Set-Up:

- $\boldsymbol{\theta} \in \Theta$, constructed from i.i.d sample $z = \{(x_i, y_i)\}_{i=1}^{n}$
- loss function $L(z_i, \boldsymbol{\theta})$

Empirical Risk Minimizer (ERM):

$$\widehat{\boldsymbol{\theta}} \in \underset{\widehat{\boldsymbol{\theta}} \in \boldsymbol{\Theta}}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \boldsymbol{\theta})$$

Maximum Likelihood Estimation (MLE):

$$\widehat{\boldsymbol{\theta}} = \underset{\widehat{\boldsymbol{\theta}} \in \boldsymbol{\Theta}}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} -\log p_{\theta}(y|x)$$

Theory: Influence Functions

Functional = function that maps a distribution to a real number.

Example: the sample mean, $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, as a functional, $T(F_n)$,

$$\bar{x} = \int x \, dF_n = T(F_n)$$

where F_n is the empirical distribution.

Why Important?

Easily examine estimator under different distributions

Example: "Contaminated Distribution"

$$F_\epsilon = (1-\epsilon)F(x) + \epsilon G(x) \text{ for } \epsilon \in [0,1]$$

Dog

Cat

Theory: Influence Functions

Consider a prediction problem,

Training Set

z_1 z_2 z_3 z_4 z_5 z_6

$$z_i = (x_i, y_i) \in X \times Y$$

Parameter of Interest

$$\hat{\boldsymbol{\theta}} \in \arg\min_{\hat{\boldsymbol{\theta}} \in \Theta} \frac{1}{6} \sum_{i=1}^{6} L(z_i, \boldsymbol{\theta})$$

Parameters with all training data (known)

$$\hat{\boldsymbol{\theta}}_{\epsilon,z} \in \arg\min_{\hat{\boldsymbol{\theta}} \in \Theta} (1-\epsilon) \frac{1}{6} \sum_{i=1}^{6} L(z_i, \boldsymbol{\theta}) + \epsilon L(z_1, \boldsymbol{\theta})$$

Using a linear approximation[1] of $\hat{\theta}_{\epsilon,z}$ around $\epsilon = 0$, this leads to the following approximation

$$\hat{\boldsymbol{\theta}}_{\epsilon,z} - \boxed{\hat{\boldsymbol{\theta}}} \approx \boxed{\frac{d\,\hat{\boldsymbol{\theta}}_{\epsilon,z}}{d\epsilon}}\,\epsilon$$

Derivative of contamination distribution

Change in parameters due to removing 1 training point (unknown)

As a Functional

$$\hat{\boldsymbol{\theta}}_{\epsilon,z} = \boxed{T\left((1-\epsilon)F_\theta + \epsilon\delta_{z_1}\right)}$$

"Contaminated Distribution"

[1] $f(x) \approx f(a) + f'(a)(x-a)$

Theory: Influence Functions

Definition 2: Influence Function

The influence functions, $IF(x; T, F_\theta)$ of T, is

Derivative of
"Contaminated Distribution"

$$IF(x; T, F_\theta) = \boxed{\lim_{\epsilon \to 0} \frac{T\big((1-\epsilon)F_\theta + \epsilon\delta_x\big) - T(F_\theta)}{\epsilon}}$$

where δ_x is the probability measure that places a point mass 1 at x.

1. At least asymptotically
2. We assume T is Fisher consistent, $\theta = T(F_\theta)$

Theory: Influence Functions

Cook and Weisberg (1982) classical result

$$IF\left(x; T, F_{\widehat{\theta}}\right) = -H_{\widehat{\theta}}^{-1} \nabla_{\theta} L(z, \hat{\theta})$$

where[1] $H_\theta \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \nabla_{\theta}^{2} L(z_i, \theta)$ and assumed positive definite.

Parameter of Interest

To remove a training examples $z \rightarrow \epsilon = \frac{1}{n}$

$$\widehat{\boldsymbol{\theta}}_{-\boldsymbol{z}} \approx \widehat{\boldsymbol{\theta}} - \frac{1}{n} H_{\widehat{\boldsymbol{\theta}}}^{-1} \nabla_{\boldsymbol{\theta}} L(z, \widehat{\boldsymbol{\theta}})$$

[1] H_θ is also called the observed Fisher's Information Matrix

[2] For simplicity allow $\theta_{-z} = \boldsymbol{\theta}_{\frac{1}{n}, \boldsymbol{z}}$

Outline

1. Theory
   - Background: ERM and GLM
   - Influence Functions
2. **Useful for Language Models?**
   - Implementation Challenges
3. Influence Functions for Model Editing
   - Methods
   - Experiment
   - Results

Theory: Implementation Challenges

$$\widehat{\boldsymbol{\theta}}_{-\boldsymbol{z}} \approx \widehat{\boldsymbol{\theta}} - \frac{1}{n} H_{\boldsymbol{\theta}}^{-1} \nabla_{\boldsymbol{\theta}} L(z, \widehat{\boldsymbol{\theta}})$$

Challenge #1:

Computing the inverse Hessian of the empirical risk alone with large parameters = COSTLY

Solution:

Hessian Vector Products (HVP) to efficiently approximate,

$$s \approx \boldsymbol{H}_{\widehat{\boldsymbol{\theta}}}^{-1} \boldsymbol{\nabla}_{\boldsymbol{\theta}} \boldsymbol{L}(\boldsymbol{z}, \widehat{\boldsymbol{\theta}})$$

Theory: Implementation Challenges

HVP:

The approximation of $\hat{\theta}_z$ still requires the calculation of

$$s \approx \boldsymbol{H_{\hat{\theta}}^{-1} \nabla_{\theta} L(z, \hat{\theta})}$$

Solution:

Frame as solving a linear system

$$Hx = v \rightarrow H^{-1}v = x$$

Use first or second order stochastic methods

- Conjugate Gradient Descent
- CURVEBALL (Henriques et al. 2019)

Method: Conjugate Gradient Descent

Algorithm 1: Conjugate Gradient Descent

for $t = 1, \ldots, T$ **do**

$\quad x_0 \leftarrow 0, r_0 = b - Ax_0, p_0 = r_0$

$\quad$ **for** k$= 1, \ldots, K$ **do**

$$\alpha_k = \frac{p_k^T r_k}{p_k^T A p_k}$$

$$x_{k+1} = x_k + \alpha_k p_k$$

$$r_{k+1} = b - Ax_k$$

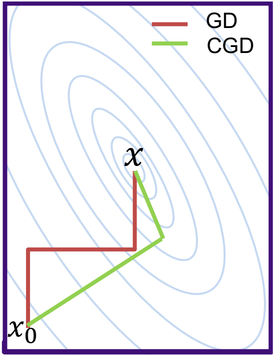
$$p_{k+1} = r_k - \sum_{i<k} \frac{p_i^T A r_k}{A p_i}$$

$\quad \theta_{t+1} = \theta_t + x_K$

Two "For" loops needed

New direction conjugate to all previous directions

Conjugate Gradient Descent



- First-order method
  - $y_t = x_t - \eta \nabla f(x_t)$
- Accelerates convergences rate of gradient descent by not repeating a direction

Method: CURVEBALL [Henriques et al. 2019]

Algorithm 2: CURVEBALL [Henriques et al., 2019]

$\mathrm{x}_0 \leftarrow 0$

for $\mathrm{k} = 1, \ldots, K$ **do**

$\nabla_x = \hat{A}x_k + J$ Second-order newton method

$x_{k+1} = \rho x_k + \alpha \nabla_x$

$\theta_{k+1} = \theta_k + x_{k+1}$ Uses momentum parameter

Only one "For" loop

- Second-order method
  - $y_t = x_t - \eta[\nabla^2 f(x_t)]^{-1} \nabla f(x_t)$
- Adds curvature term (second order)
- Faster and less memory use
  - Reuses previous direction
  - Interweaves search direction and parameter update (only1 "For" loop)
- Specifically tailored for deep-learning-scale stochastic optimization problems

Implementation Challenges

Challenge #2:

The HVP, requires the loss over the original data set which can be computationally expensive.

Solution:

Use clustering techniques to find a mini-batch that is representative of the original data set.

Steps:

1. Start with all features from the Original data set
2. Perform PCA to reduce dimensionality
3. Perform clustering on the features
4. Closes data point to each cluster center to include in the minibatch

Outline

1. Theory
   - Background: ERM and GLM
   - Influence Functions
2. Useful for Language Models?
   - Implementation Challenges
3. **Influence Functions for Model Editing**
   - Methods
   - Experiment
   - Results

Model Editing: Goal

Task: Model Editing

Goal: Develop a cost-effective, post-hoc model editing technique to edit knowledge in a trained language models.

Data: Zero Shot Relation Extraction (zsRE)

Dataset (D)

Edited (D_{ed})

Q: What is the name of Another Side of Bob Dylan's record label?

A: Capitol Records

Non-edited (D_{non-ed})

Q: What country did The Laughing Cow originate?

A: France

Forget (D_F)

Q: What is the name of Another Side of Bob Dylan's record label?

A: Capitol Records

Remember (D_R)

Q: What is the name of Another Side of Bob Dylan's record label?

A: Colombia Records

[8]Levy et al. 2017

Model Editing: Baseline

Notation

- θ_0 = parameters of the original model
- $\widehat{\boldsymbol{\theta}}$= parameters of the edited model
- T = number of updates
- L_R= loss over *DR*
- η= learn. rate
- P = Distribution of reg. subset under $\boldsymbol{\theta}_0$
- Q = Distribution of reg. subset under $\widehat{\boldsymbol{\theta}}$

Algorithm 1: Baseline

Initialize $\widehat{\boldsymbol{\theta}}_0 = \boldsymbol{\theta}_0$.
for $t = 0, \ldots, T-1$ **do**

$$\widehat{\boldsymbol{\theta}}_{t+1} = \underbrace{\widehat{\boldsymbol{\theta}}_t - \eta \nabla \mathrm{L_r}(\widehat{\boldsymbol{\theta}}_t)}_{\text{Newton step}} + \underbrace{D_{KL}(P||Q)}_{\text{Regularization term}}$$

end for

Model Editing: Influence Function Method

Notation

- T = number of updates for step 1
- S = number of updates for step 2
- L_F = loss over D_F

Algorithm 2 Influence Function Method

Step 1: Forgetting

Initialize $\widehat{\boldsymbol{\theta}}_0 = \boldsymbol{\theta}_0$.
For $t = 0, \ldots, T-1$ ***do***

$$\widehat{\boldsymbol{\theta}}_{t+1} = \underbrace{\widehat{\boldsymbol{\theta}}_t - H_{\boldsymbol{\theta}_0}^{-1} \nabla_{\boldsymbol{\theta}} L_F(\widehat{\boldsymbol{\theta}}_t)}_{\text{Approximation using IF}}$$

end for

Step 2: Remembering

Initialize $\widehat{\boldsymbol{\theta}}_0 = \hat{\theta}_T$.
for $\boldsymbol{s} = \mathbf{0}, \ldots, \boldsymbol{S} - \mathbf{1}$ **do**

$$\widehat{\boldsymbol{\theta}}_{s+1} = \underbrace{\widehat{\boldsymbol{\theta}}_s - \eta \nabla \mathrm{L_r}(\widehat{\boldsymbol{\theta}}_s) + D_{KL}\,(P||Q)}_{\text{Baseline}}$$

end for

Experimentation: Details

Details:

- # of Non-Edits: 10,000
- # of Edits: 40
- Repetition: 10

- Epochs Baseline: 60
- Epochs Step 1 (forg.): 4
- Epochs Step 2 (rem.): 56

Metrics:

1. Reliability: made edits successfully

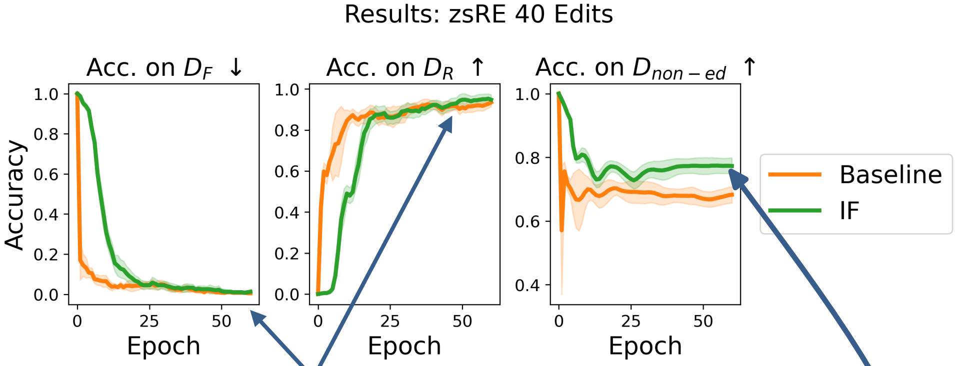
Accuracy over D_R (increase)

Accuracy over D_F (decrease)

2. *Generality:* did not change non-edited input/out

Accuracy over D_{non-ed} (increase)

Results: Accuracy



- *Reliability*: Accuracy over D_F and D_R are similar
- *Generality*: Retention of Non-edited is better with IF method
    - ➢ "Forgetting" is more targeted

Conclusion

- Theoretically influence functions good solution to problem

- However, approximation techniques → fragile or inaccurate

- Promising results in application to model editing indicates more experimentation

Thank you!

Questions?

References

- Agarwal, Naman, Brian Bullins, and Elad Hazan (2016). "Second-order stochastic optimization in linear time". In: stat 1050, p. 15.
- Basu, Samyadeep, Philip Pope, and Soheil Feizi (2020). "Influence functions in deep learning are fragile". In: arXiv preprint arXiv:2006.14651.
- Cook, R Dennis and Sanford Weisberg (1982). Residuals and influence in regression. New York: Chapman and Hall.
- De Cao, Nicola, Wilker Aziz, and Ivan Titov (2021). "Editing factual knowledge in language models". In: arXiv preprint arXiv:2104.08164.
- Han, Xiaochuang, Byron C. Wallace, and Yulia Tsvetkov (2020). "Explaining Black Box Predictions and Unveiling Data Artifacts through Influence Functions". In: arXiv: 2005.06676 [cs.CL].
- Henriques, João F et al. (2019). "Small steps and giant leaps: Minimal newton solvers for deep learning". In: Proceedings of