Fast Model Editing At Scale

Allen Gao

Research Problem

How to efficiently edit large pre-trained language model (LLM) ?

Challenges:

- Models are getting larger and larger
- Fine tune could easily overfit, computationally expensive
- Black box nature of representation
- Factualness and Reliability of Model

PennState

Agenda

- Background
- Related Work
- Model Editor Networks with Gradient Decomposition (MEND)
- Experiment / Result
- Future Work / Limitation
- Question?

Background

<u>Reliability</u>: Successfully changing the model's output on the problematic input.

<u>Locality</u>: Minimally affecting the model's output for unrelated inputs

<u>Generality</u>: Generating the correct output for inputs related to the edit input.

<u>Efficiency</u>: The computation spend on making the edit.

PennState

Background

Input	Pre-Edit Output	Edit Target	Post-Edit Output
1a: **Who is India's PM?**	Satya Pal Malik ✗	**Narendra Modi**	Narendra Modi ✓
1b: **Who is the prime minister of the UK?**	Theresa May ✗	**Boris Johnson**	Boris Johnson ✓
1c: Who is the prime minister of India?	Narendra Modi ✓	—	Narendra Modi ✓
1d: Who is the UK PM?	Theresa May ✗	—	Boris Johnson ✓
2a: **What is Messi's club team?**	Barcelona B ✗	**PSG**	PSG ✓
2b: **What basketball team does Lebron play on?**	Dallas Mavericks ✗	**the LA Lakers**	the LA Lakers ✓
2c: Where in the US is Raleigh?	a state in the South ✓	—	a state in the South ✓
3a: **Who is the president of Mexico?**	Enrique Pea Nieto ✗	**Andrés Manuel López Obrador**	Andrés Manuel López Obrador ✓
3b: Who is the vice president of Mexico?	Yadier Benjamin Ramos ✗	—	Andrés Manuel López Obrador ✗

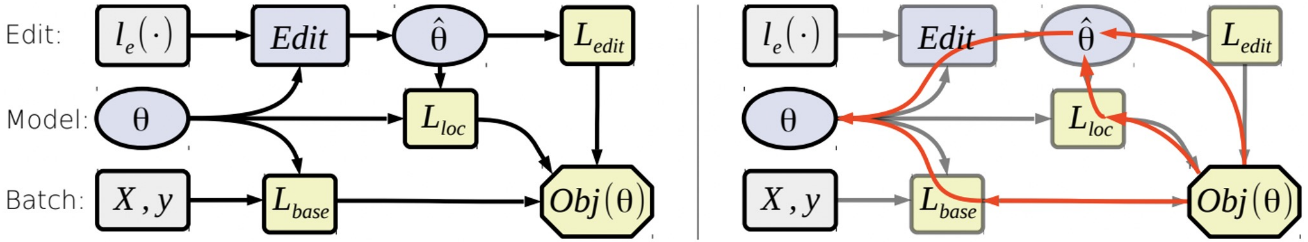
PennState

Related Work

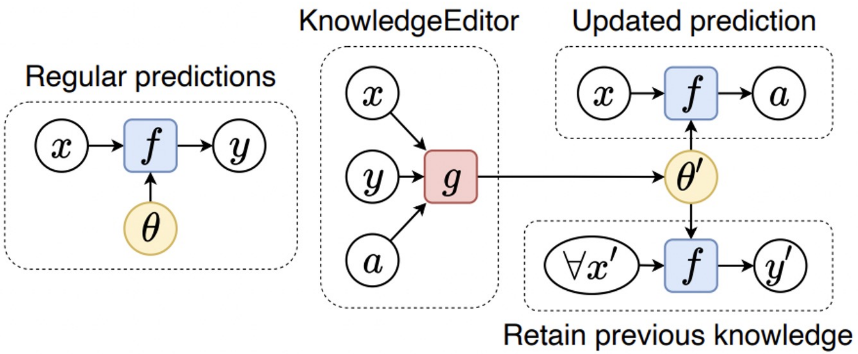
- Selective Fine-tuning the model with edited dataset (Zhu et al, 2020)
  - Overfit to the edited dataset, poor locality, require full training data
- Train a knowledge editor to map original parameter weight(De Cao et al, 2021)
  - Fail to edit very large model
- Bi-level meta learning editable training method (Sinitsin et al, 2020)
  - Difficult to edit large model, Computationally expensive

Model illustration

ENN



Knowledge editor



Related Work

Editor	Preserves model?	Only (x_e, y_e)?	Batched edits?	Scales to 10B?	Few steps?
FT	✓	✓	✓	✓	✗
FT+KL	✓	✗	✓	✓	✗
ENN	✗	✓	✓	✗	✓
KE	✓	✓	?	✓	✓
MEND	✓	✓	✓	✓	✓

MEND

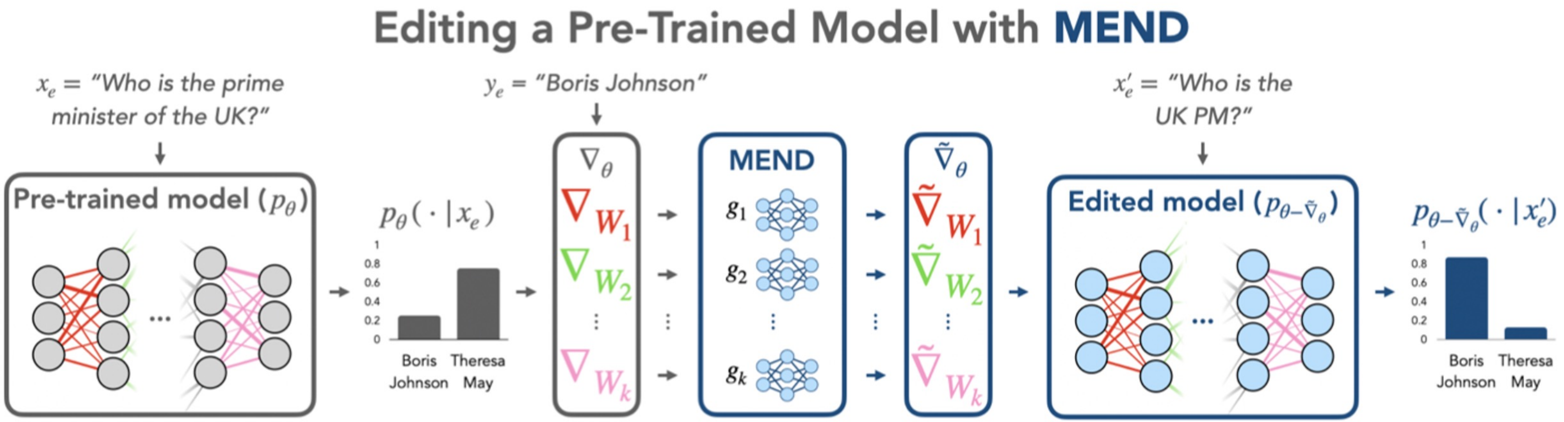


Figure 1: The proposed algorithm MEND enables editability by training a collection of MLPs to modify model gradients to produce *local* model edits that do not damage model performance on unrelated inputs. MEND is efficient to train and apply edits, even for very large models, as shown in Section 5.1.

PennState

MEND

Base Model: $f_\theta : \mathcal{X} \times \Theta \rightarrow \mathcal{Y}$

Model Editor Function: $E : \mathcal{X} \times \mathcal{Y} \times \mathcal{L} \times \Theta \times \Phi \rightarrow \Theta$

Loss Function: $\mathcal{X} \times \mathcal{Y} \times \Theta \rightarrow \mathbb{R}$ $\quad l_e(x, y, \theta) = -\log p_\theta(y|x)$

Editor Model Eval Dataset: $D^{te}_{edit} = \{(x_\mathrm{e}, y_\mathrm{e}, x_\mathrm{loc}, x'_\mathrm{e}, y'_\mathrm{e})_i\}$

For $x_\mathrm{e}, y_\mathrm{e}$ = *Who is the prime minister of the UK? Boris Johnson*, $N(x_\mathrm{e}, y_\mathrm{e})$ might contain $x'_\mathrm{e}, y'_\mathrm{e}$ = *Who is the UK PM? Boris Johnson*, among others. x_loc might be *What team does Messi play for?*.

MEND

Reliability: post-edit model predicts the edit label y for the edit input x

Locality: $\mathbb{E}_{x_{\text{loc}} \sim D_{edit}^{te}} \text{KL}(p_\theta(\cdot|x_{\text{loc}}) \| p_{\theta_e}(\cdot|x_{\text{loc}}))$

Generality: post-edit model predict correctly on $(x'_{\text{e}}, y'_{\text{e}}) \in N(x_{\text{e}}, y_{\text{e}})$

Efficiency: Time and memory requirement when training and applying the editor model

$$\text{ES} = \mathbb{E}_{x'_{\text{e}}, y'_{\text{e}} \sim N(x_{\text{e}}, y_{\text{e}}) \cup \{(x_{\text{e}}, y_{\text{e}})\}} \mathbb{1}\{\text{argmax}_y \, p_{\theta_e}(y|x'_{\text{e}}) = y'_{\text{e}}\}$$

MEND losses: $L_{\text{e}} = -\log p_{\theta_{\tilde{\mathcal{W}}}}(y'_{\text{e}}|x'_{\text{e}}), \quad L_{\text{loc}} = \text{KL}(p_{\theta_{\mathcal{W}}}(\cdot|x_{\text{loc}}) \| p_{\theta_{\tilde{\mathcal{W}}}}(\cdot|x_{\text{loc}})). \quad (4\text{a,b})$

PennState

MEND

D RANK-1 GRADIENT FOR MLPS

In the simplified case of an MLP and a batch size of 1, we describe the rank-1 gradient of the loss L with respect to the layer ℓ weight matrix W_ℓ. We define the inputs to layer ℓ as u_ℓ and the *pre-activation* inputs to layer $\ell+1$ as $z_{\ell+1} = W_\ell u_\ell$. We define $\delta_{\ell+1}$ as the gradient of L with respect to $z_{\ell+1}$ (we assume that $\delta_{\ell+1}$ is pre-computed, as a result of standard backpropagation). We will show that the gradient of the loss L with respect to W_ℓ is equal to $\delta_{\ell+1} u_\ell^\top$.

By the chain rule, the derivative of the loss with respect to weight W_ℓ^{ij} is equal to

$$\frac{\partial L}{\partial W_\ell^{ij}} = \sum_k \frac{\partial L}{\partial z_{\ell+1}^k} \frac{\partial z_{\ell+1}^k}{\partial W_\ell^{ij}} = \frac{\partial L}{\partial z_{\ell+1}^i} \frac{\partial z_{\ell+1}^i}{\partial W_\ell^{ij}} \tag{7}$$

the product of the derivative of L with respect to next-layer pre-activations $z_{\ell+1}^i$ and the derivative of next-layer pre-activations $z_{\ell+1}^i$ with respect to W_{ij}. The second equality is due to the fact that $\frac{\partial z_{\ell+1}^k}{\partial W_\ell^{ij}} = 0$ for $k \neq i$. Noting that $z_{\ell+1}^i = \sum_j u_\ell^j W_\ell^{ij}$, we can replace $\frac{\partial z_{\ell+1}^i}{\partial W_\ell^{ij}}$ with simply u_ℓ^j in Equation 7. Further, we defined $\delta_{\ell+1}$ to be exactly $\frac{\partial L}{\partial z_{\ell+1}^i}$. Making these two substitutions, we have

$$\frac{\partial L}{\partial W_\ell^{ij}} = \delta_{\ell+1}^i u_\ell^j \tag{8}$$

or, in vector notation, $\nabla_{W_\ell} L = \delta_{\ell+1} u_\ell^\top$, which is the original identity we set out to prove.

MEND

$d \approx 10^4$ (Number of weight in one layer)   O(d x d) is too costly!

instead: $\nabla_{W_\ell} L = \sum_{i=1}^{B} \delta_{\ell+1}^{i} u_\ell^{i\top}$

		Wikitext Generation		zsRE Question-Answering	
		distilGPT-2 (82M)		BART-base (139M)	
MEND Variant	Editor Parameters	ES ↑	ppl. DD ↓	ES ↑	acc. DD ↓
No sharing	$O((m+n)^2N)$	**0.86**	**0.195**	**>0.99**	0.001
No norm.	$O((m+n)^2)$	0.02	0.370	0.97	**<0.001**
No ID init.	$O((m+n)^2)$	0.27	0.898	0.94	**<0.001**
Only u_ℓ	$O(m^2)$	0.63	0.559	0.98	0.002
Only $\delta_{\ell+1}$	$O(n^2)$	0.80	0.445	**0.99**	0.001
Only smaller	$O(\min(m,n)^2)$	0.80	0.593	0.98	0.002
MEND	$O((m+n)^2)$	**0.86**	0.225	**>0.99**	0.001

PennState

MEND

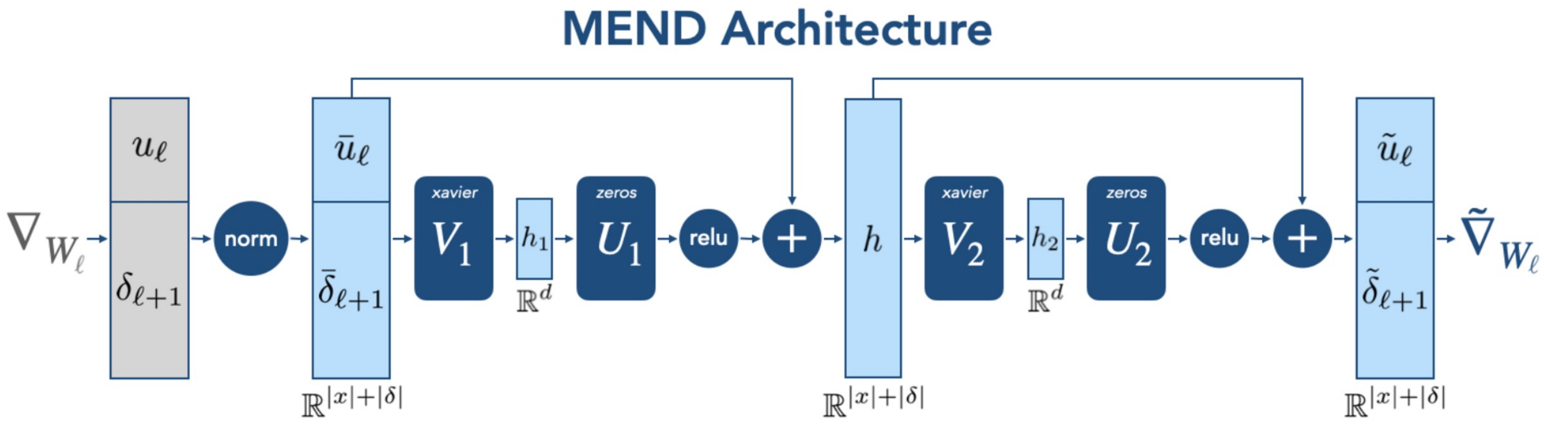


Figure 2: The MEND architecture, consisting of two consecutive blocks, both initialized to compute the exact identity function. **Left.** The input to a MEND network is $\{\delta_{\ell+1}, u_\ell\}$, the components of the rank-1 gradient. **Right.** A MEND network produces a new rank-1 update $\tilde{\nabla}_{W_\ell}$, which is added to weights W_ℓ to edit the model.

$$h_\ell = z_\ell + \sigma(s_\ell^1 \odot (U_1 V_1 z_\ell + b) + o_\ell^1), \qquad g(z_\ell) = h_\ell + \sigma(s_\ell^2 \odot U_2 V_2 h_\ell + o_\ell^2)$$

PennState

MEND

Algorithm 1 MEND Training

Input: Pre-trained $p_{\theta_{\mathcal{W}}}$, weights to make editable $\mathcal{W}$, editor params ϕ_0, edit dataset D^{tr}_{edit}, edit-locality tradeoff c_{edit}
for $t \in 1, 2, \ldots$ **do**
  Sample $x_e, y_e, x'_e, y'_e, x_{\text{loc}} \sim D^{tr}_{edit}$
  $\tilde{\mathcal{W}} \leftarrow \textsc{Edit}(\theta_{\mathcal{W}}, \mathcal{W}, \phi_{t-1}, x_e, y_e)$
  $L_e \leftarrow -\log p_{\theta_{\tilde{\mathcal{W}}}}(y'_e | x'_e)$
  $L_{\text{loc}} \leftarrow \text{KL}(p_{\theta_{\mathcal{W}}}(\cdot | x_{\text{loc}}) \| p_{\theta_{\tilde{\mathcal{W}}}}(\cdot | x_{\text{loc}}))$
  $L(\phi_{t-1}) \leftarrow c_{\text{edit}} L_e + L_{\text{loc}}$
  $\phi_t \leftarrow \text{Adam}\,(\phi_{t-1}, \nabla_\phi L(\phi_{t-1}))$

Algorithm 2 MEND Edit Procedure

procedure $\textsc{Edit}(\theta, \mathcal{W}, \phi, x_e, y_e)$
  $\hat{p} \leftarrow p_{\theta_{\mathcal{W}}}(y_e | x_e)$, **caching** input u_ℓ to $W_\ell \in \mathcal{W}$
  $L(\theta, \mathcal{W}) \leftarrow -\log \hat{p}$ ▷ Compute NLL
  for $W_\ell \in \mathcal{W}$ **do**
    $\delta_{\ell+1} \leftarrow \nabla_{W_\ell u_\ell + b_\ell} l_e(x_e, y_e)$ ▷ Grad wrt output
    $\tilde{u}_\ell, \tilde{\delta}_{\ell+1} \leftarrow g_{\phi_\ell}(u_\ell, \delta_{\ell+1})$ ▷ Pseudo-acts/deltas
    $\tilde{W}_\ell \leftarrow W_\ell - \tilde{\delta}_{\ell+1} \tilde{u}_\ell^\top$ ▷ Layer ℓ model edit
  $\tilde{\mathcal{W}} \leftarrow \{\tilde{W}_1, \ldots, \tilde{W}_k\}$
  return $\tilde{\mathcal{W}}$ ▷ Return edited weights

MEND losses: $$L_e = -\log p_{\theta_{\tilde{\mathcal{W}}}}(y'_e | x'_e), \quad L_{\text{loc}} = \text{KL}(p_{\theta_{\mathcal{W}}}(\cdot | x_{\text{loc}}) \| p_{\theta_{\tilde{\mathcal{W}}}}(\cdot | x_{\text{loc}})). \quad \text{(4a,b)}$$

Experiment / Result

Input	Pre-Edit Output	Edit Target	Post-Edit Output
1a: **Who is India's PM?**	Satya Pal Malik ✗	**Narendra Modi**	Narendra Modi ✓
1b: **Who is the prime minister of the UK?**	Theresa May ✗	**Boris Johnson**	Boris Johnson ✓
1c: Who is the prime minister of India?	Narendra Modi ✓	—	Narendra Modi ✓
1d: Who is the UK PM?	Theresa May ✗	—	Boris Johnson ✓
2a: **What is Messi's club team?**	Barcelona B ✗	**PSG**	PSG ✓
2b: **What basketball team does Lebron play on?**	Dallas Mavericks ✗	**the LA Lakers**	the LA Lakers ✓
2c: Where in the US is Raleigh?	a state in the South ✓	—	a state in the South ✓
3a: **Who is the president of Mexico?**	Enrique Pea Nieto ✗	**Andrés Manuel López Obrador**	Andrés Manuel López Obrador ✓
3b: Who is the vice president of Mexico?	Yadier Benjamin Ramos ✗	—	Andrés Manuel López Obrador ✗

PennState

Experiment / Result

Editing Very Large Transformer Models

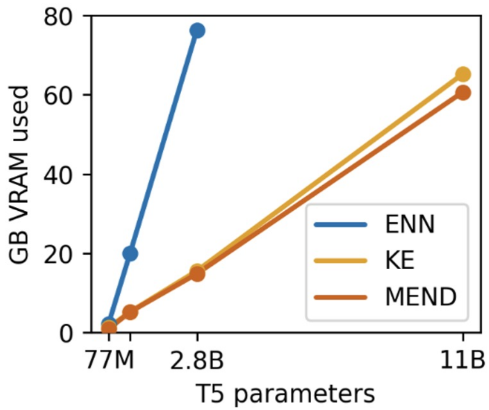
	Wikitext Generation				zsRE Question-Answering			
	GPT-Neo (2.7B)		GPT-J (6B)		T5-XL (2.8B)		T5-XXL (11B)	
Editor	ES ↑	ppl. DD ↓	ES ↑	ppl. DD ↓	ES ↑	acc. DD ↓	ES ↑	acc. DD ↓
FT	0.55	0.195	0.80	0.125	0.58	**< 0.001**	0.87	**< 0.001**
FT+KL	0.40	**0.026**	0.36	0.109	0.55	**< 0.001**	0.85	**< 0.001**
KE	0.00	0.137	0.01	0.068	0.03	**< 0.001**	0.04	**< 0.001**
MEND	**0.81**	0.057	**0.88**	**0.031**	**0.88**	0.001	**0.89**	**< 0.001**

PennState

Experiment / Result

Editing Smaller Scale Model

	FEVER Fact-Checking		zsRE Question-Answering		Wikitext Generation	
	BERT-base (110M)		BART-base (139M)		distilGPT-2 (82M)	
Editor	ES ↑	acc. DD ↓	ES ↑	acc. DD ↓	ES ↑	ppl. DD ↓
FT	0.76	**<0.001**	0.96	**<0.001**	0.29	0.938
FT+KL	0.64	**<0.001**	0.89	**<0.001**	0.17	**0.059**
ENN	**0.99**	0.003	**0.99**	**<0.001**	**0.93**	0.094
KE	0.95	0.004	**0.98**	**<0.001**	0.25	0.595
MEND	**>0.99**	**<0.001**	**0.98**	0.002	0.86	0.225



Experiment / Result

Batch Editing:

- MEND applies simultaneous edits by simply summing the parameter edit computed separately for each edit example.

	Edit Success ↑		Acc. Drawdown ↓	
Edits	ENN	MEND	ENN	MEND
1	0.99	0.98	<0.001	0.002
5	0.94	0.97	0.007	0.005
25	0.35	0.89	0.005	0.011
75	0.16	0.78	0.005	0.011
125	0.11	0.67	0.006	0.012

PennState

Future Work / Limitation

Conclusion:

- MEND is the only editor model that can scale to very large LLM (10 billions +)
- Can make effective single input output pair edit
- Leverage the fact that gradients with respect to the fully-connected layers in neural networks are rank-1

Future Work / Limitation

Limitation:

1. Need to have a stronger reinforcement in locality
2. Logic reasoning, the edited answer may not transfer to similar question which implied this answer
3. Mostly used in short phrase prediction, fact checking / question answering
4. Which block or layer should we apply MEND, how do we determine that

PennState

Questions?