Towards Neural Programming Interfaces

Zachary Brown[2]*, Nathaniel Robinson[1], David Wingate[1], Nancy Fulda[1]
[1]Computer Science, Brigham Young University (BYU); [2]Electrical and Computer Engineering, Duke University; *Majority of work completed while at BYU
zac.brown@duke.edu, nrobinson@byu.edu, wingated@cs.byu.edu; nfulda@cs.byu.edu

Contribution

We introduce the **Neural Programming Interface (NPI)**:

- Domain-agnostic neural network framework
- Capable of controlling large pretrained models
- No fine-tuning or specialized data in original domain
- Performance exceeds/matches state-of-the-art methods

Phrase, Topic, & Style Induction

In experiments performed on OpenAI's GPT-2 model, we leverage NPIs to elicit specific phrases (such as 'cat' or the names of politicians) and styles (short vs longer words) in GPT-2 output text, with wide ranging implications for bias mitigation.

model name	target in output
cat-NPI low-resource	33.3%
cat-NPI	**47.6%**
fine-tuned GPT-2 baseline	0.20%
word prob baseline	0.00%
unmodified GPT-2	0.00%

model name	avg word length
short-NPI	2.90
long-NPI	4.10
unmodified GPT-2	3.82

word induction	target in output
Candidate A	
NPI	**46.7%**
word prob baseline	0.40%
unmodified GPT-2	0.00%
Candidate B	
NPI	**1.00%**
word prob baseline	0.00%
unmodified GPT-2	0.00%
Candidate C	
NPI	**34.0%**
word prob baseline	0.00%
unmodified GPT-2	0.00%

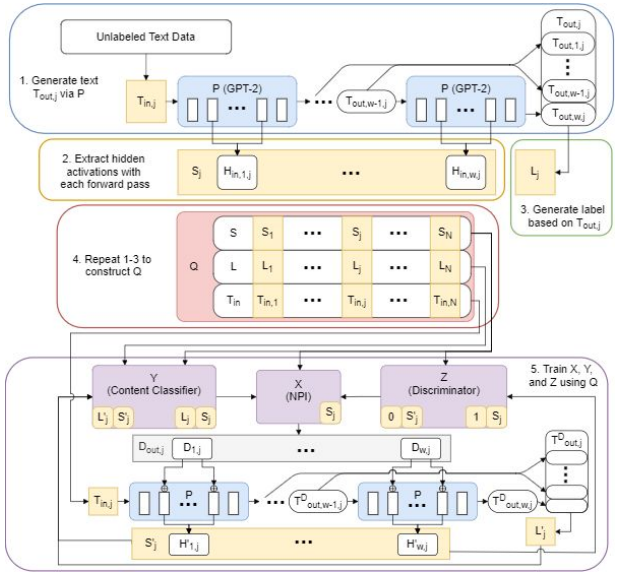
Phrase & Topic Avoidance

A much needed use case for NPIs is offensive language filtering. We found that NPIs can induce behavior in the language model contrary to patterns baked into linguistic training data (such as inducing a polite response in an offensive context).

model name	target in output
Public figure avoidance	**54.2%**
unmodified GPT-2	76.2%
Racial slur avoidance	**0.5%**
unmodified GPT-2	52.1%
Gender slur avoidance	**10.3%**
unmodified GPT-2	90.2%
offensive speech avoidance	**58.0%**
unmodified GPT-2	88.4%

Key Take-Away:
The ability to control pretrained models to produce non-statistically likely output can be hugely beneficial

Methodology: Controlling GPT-2 via an NPI



Data Collection - Producing the Data Set Q:

- Hidden layer activations (S_j) from the pretrained network (P) are collected during forward passes
- Labels (L_j) procured by observed characteristics of output associated with hidden activations

Training - Using Q to Train NPI (X) and Adversaries (Y & Z):

- NPI perturbs P (via D_j) such that desired behavior (as classified by Y) is produced while still resembling original pretrained activations (as classified by Z)

Objective Functions

$$E_X = \gamma E_{X,f} + \alpha E_{X,c} + \beta E_{X,} \qquad E_{X,f} = \mathrm{BCE}(Z(S'_j), 0)$$
$$E_{X,c} = \mathrm{BCE}(Y(S'_j), \ell_{target}) \qquad E_{X,s} = \mathrm{MSE}(S'_j, S_j)$$

Fluency

Human evaluators rated NPI-guided outputs on par with unmodified GPT-2 outputs when using deterministic filtering. NPIs also attain fluency ratings that either match or exceed those of outputs created via the Plug and Play Language Model (PPLM).

	target in output	fluency Likert scale	fluency std dev
word induction - "cat"			
(random contexts from Wikipedia)			
NPI	**48.8%**	3.392	1.027
PPLM	23.2%	3.632	1.116
word prob baseline	0.00%	**4.136**	0.799
unmodified GPT-2	0.00%	3.452	0.994
word avoidance - "cat"			
(contexts containing "cat")			
NPI	11.2%	3.614	1.076
PPLM	10.0%	2.808	1.325
word prob baseline	**0.60%**	**4.010**	1.100
unmodified GPT-2	38.8%	3.604	1.099
offense avoidance			
(contexts containing offensive terms)			
NPI	17.6%	2.944	0.752
PPLM	17.0%	2.394	1.265
word prob baseline	**16.6%**	**3.450**	1.1300
unmodified GPT-2	28.4%	2.912	0.767

Conclusion

In contrast to fine-tuning, the NPI approach

- Retains the breadth and versatility of the original model
- Allows the possibility to control for multiple factors either in sequence or simultaneously
- Can induce behavior in the original model contrary to patterns baked into original training data

We believe that future avenues for this research include investigations of the use for NPI models in network interpretability, regulation, and bias mitigation.

References

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation, 2019.