Table 1: Model comparisons across 500 utterances. Fluency was evaluated using a crowd-sourced Likert scale. Mechanical Turk workers with a "master" qualification allotted 1 to 5 stars for text quality. PPLM (Discriminator) used the same hyperparamters as NPI: top-1 filtering, small-size GPT-2. NPI clearly outperformed on word induction. PPLM slightly outperformed in word avoidance, but at a significant cost of fluency (increased frequency of degenerate text).

	target in output	embed shifts	avg shift	fluency Likert scale	fluency std dev
word induction - "cat" (random contexts from Wikipedia)					
NPI	**48.8%**	**95.4%**	0.126	3.392	1.027
PPLM	23.2%	44.0%	0.059	**3.632**	1.116
unmodified GPT-2	0%	N/A	N/A	3.452	0.994
word avoidance - "cat" (contexts containing "cat")					
NPI	11.2%	47.2%	0.009	**3.614**	1.076
PPLM	**10.0%**	**78.6%**	0.143	2.808	1.325
unmodified GPT-2	38.8%	N/A	N/A	3.604	1.099
offense avoidance (contexts containing offensive words)					
NPI	17.6%	56.4%	0.067	**2.944**	0.752
PPLM	**17.0%**	**33.8%**	0.119	2.394	1.265
unmodified GPT-2	28.4%	N/A	N/A	2.912	0.767

We thank the reviewers for their insightful remarks, and have provided additional evaluations in Table 1. Several reviewers expressed concerns about possible negative effects of the NPI architecture on fluency. The fluency evaluations in columns 4 and 5 indicate NPI does not seriously degrade the fluency of GPT-2 output in our experiments. Unfortunately utterances output by the small GPT-2 model we used are often lacking in fluency, with or without NPI intervention.

Reviewer 1 referenced the Plug and Play Language Model (PPLM) architecture, which we were not previously aware of and which was developed parallel to our work. The PPLM Discriminator approach is strikingly similar to our own in the use of an external classification network to steer GPT-2 outputs towards a desired trait. We point out what we consider three fundamental differences between our approach and PPLM. Instead of influencing text output by summing gradients from the classifier with pre-computed GPT-2 hidden states, our NPI approach employs another neural network that interfaces with multiple of GPT-2's hidden representation layers in each forward pass. This interfacing enables it to influence both macro- and micro-characteristics of the text. (See our appendix for examples of more applications than the "cat" experiments.) Another major difference in our work lies in our novel data curation approach. The training data for PPLM's classifiers is obtained with pre-labeled text data that exemplifies the desired style or topic, which is fed through the GPT-2 as context to obtain textual representations. Initially we experimented with a similar approach, but we realized this method relies on the assumption that the style or topic of GPT-2 output will frequently match that of the input context. While this assumption holds for some tasks, we predicted problems for our fine-grained task of causing a specific word or brand name to appear. (Inputting a sentence containing "cat" to the GPT-2 does not guarantee that the output will contain "cat".) We value our approach of sending arbitrary inputs through GPT-2 and then labeling our data based on the properties of the output. We see our current application as a proof of concept for NPI use in various areas of AI, and our data curation approach is applicable to networks where the input is random or meaningless (such as image generation networks that accept Gaussian noise as input). As a last distinction, while our data curation and training processes are slower, our text perturbation process is roughly 30 times faster than that of PPLM.

We performed a number of ablation experiments for our NPI method. One such approach was to reduce the probability of unwanted tokens to zero in GPT-2 processing for specific term-avoidance. This approach seems to work well, but we esteemed it undesirable for certain applications because often tokens can be sub-words of other words. (A model that cannot output "cat" likewise cannot will have difficulty outputting "category".) We also experimented with boosting token probabilities for a desired word. However, this method was significantly less effective at producing the desired word than our NPI approach unless we forced GPT-2 to select considerably unlikely or contextually impossible tokens.

Some of our NPI models were trained on over 90000 examples but most were trained on approximately 1000. Our approach could be modified to variations of *top_p* and *top_k* sampling. We chose a more deterministic approach to facilitate testing and evaluation. We chose not to run baseline tests with CTRL and other text-control models because we could not consolidate differences in training data used. But we make a comparison of model parameters in our appendix. We apologize that our model description is rigorous and complex; we will attempt to clarify in future versions. Our appendix may offer supplementary insight if reviewers have questions about method details.