Text Classifications Learned from Language Model Hidden Layers

Nathaniel Robinson
Dept. of Computer Science
Brigham Young University
Provo, UT, USA
nrobinson@byu.edu

Zachary Brown
Electrical and Computer Engineering
Duke University
Durham, NC, USA
zac.brown@duke.edu

Timothy Sitze
Dept. of Mathematics
Brigham Young University
Provo, UT, USA
texas.sitze@gmail.com

Nancy Fulda
Dept. of Computer Science
Brigham Young University
Provo, UT, USA
nfulda@cs.byu.edu

***Abstract*—Advancements in machine learning methods have yielded powerful natural language generation models. However, in general, these models have drawn concern for being both uninterpretable and uncontrollable. Model interpretability and control have become important topics of interest among researchers. We explore a variety of machine learning methods to classify the hidden states of language models. This classification enables model interpretation at a deep semantic level and is a necessary part of recently proposed model control methods. We show further that the use of language model hidden layers as text representations in classification tasks may be more reliable in some applications than more standard text representations.**

I. Introduction: Problem Statement & Motivation

Progress in automated natural language processing has accelerated in recent years. Yet despite the fact that recent language generation methods attain almost human-level performance on certain tasks, intelligent voice assistants are still largely based on state-machine architectures. This is due in part to the fact that end-to-end deep learning methods cannot be explicitly controlled to say specific phrases and therefore might output false or offensive statements. Because of this lack of control, these deep learning models pose too great a risk to most entities that wish to deploy public-facing intelligent agents. Furthermore, current language models are almost entirely uninterpretable and therefore difficult to test for egregious failure cases and to debug in general. This makes the problem of control difficult to approach.

In this paper, we present experiments on several classification algorithms that are capable of interpreting semantic and topical information encoded in the high-dimensional hidden-layer activations of modern natural language models, specifically the GPT-2 pretrained neural network by OpenAI [1]. The GPT-2 network is a powerful language model that was trained to predict the next word in a given passage of text, but as a neural network is susceptible to predicting - without warning - insulting or harmful phrases as the most likely text to follow a given input. We hope to approach a way to mitigate this problem by classifying GPT-2 hidden representations as they represent text.

Although the lack of explicit controls and interpretability for deep learning algorithms poses a risk to entities seeking to leverage them, the risks have not prevented some corporations from experimenting with these algorithms. In some cases these experiments have resulted in public-relations disasters, most notably Microsoft's insulting chat bot Tay [2]. Our goal is to develop a classifier that is able to identify undesirable content being produced by a given network by interfacing with the network's inner representations. Classifiers like this are an integral part of algorithms that successfully control the output of language models, such as Plug and Play Language Models [3] and Neural Programming Interfaces [4]. (Notably, the classifiers used in Neural Programming Interfaces also classify the hidden layer activations of GPT-2.)

II. Methodology

The data sets we generated consist of activation tensors pulled from the first and last layer of OpenAI's GPT-2 model [1], as implemented on the HuggingFace-Transformers GitHub repository [5]). Our general process is detailed in Figure 1. Collection of all the data sets described in this paper began with a set of labeled sentences $T_{in,j}$ sorted into a number of classes and labeled L_j. To generate a data set, we reduced every sentence to fifteen tokens and passed it through the GPT-2 fifteen times. From these fifteen iterations we collected fifteen 15x1024 arrays from the GPT-2 first layer and fifteen 14x1024 arrays from the GPT-2 final layer. We then concatenated these thirty arrays into one 435x1024 S_j array associated with the original sentence $T_{in,j}$. A data set Q therefore consists of a collection of data points, each containing a sentence $T_{in,j}$, a label indicating the class of the sentence L_j, and a 435x1024 array S_j pulled from GPT-2 activation layers over successive forward passes. In some applications (such as our random forest analysis) we replaced the 435x1024 array with a 14x1024 array pulled from the final layer of the GPT-2 after passing the corresponding sentence through GPT-2 once without repetition.

A. Classification Algorithms

We considered a variety of methods for our classification task. Originally we believed that dimension reduction techniques would facilitate classification. However, Principal Component Analysis (PCA), Non-negative Matrix Factorization (NMF), Randomized Dimension Reduction (RDR) and K-Means all proved to be too memory intensive and/or required unreasonable assumptions about the data to be useful. Spectral

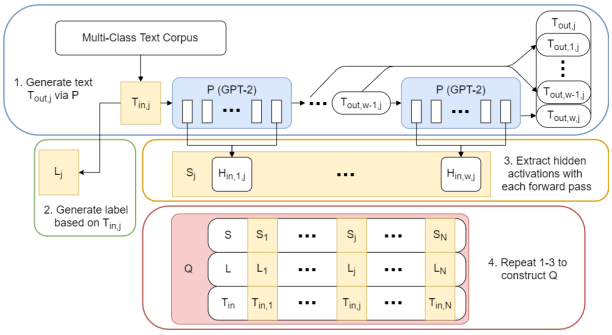

Fig. 1. Data collection process illustrated

Clustering relies on a single, memory intensive graphical representation of the input data and due to time constraints did not seem like a promising classification method. Though UMAP has several documented shortcomings, it is efficient enough from a computational memory perspective to apply to a small data set and test whether GPT-2 activations can be reduced to a human-interpretable representation for classification. We found that t-SNE was less memory-hungry than methods like PCA and could be applied in a similar way to UMAP.

We chose to focus primarily on Neural Network methods because of their ability to process large data sets via mini-batching and their robustness to different forms of input and labels. We focused as well on Decision Tree methods because their interpretable approach to classification could offer insight into the inner workings of non-interpretable language models. Though less interpretable than lone decision trees, we also explored random forest classifiers because they train quickly and usually obtain higher classification accuracy.

B. Indications of Structure in Hidden Layers

Preliminary analyses indicate classifiable semantic structure in GPT-2 hidden layers. See Figures 2 and 3. For these experiments we collected a large data set with phrases sorted into 50 different categories (determined by a list of 50 common words, one of which was present in each sentence).

III. Experiments and Results

In order to interpret the development of offensive outputs of language models, we trained a variety of classifiers on a data set consisting of activations for offensive and non-offensive sentences. Sentence labels were determined by human evaluation [6] [7]. Results are displayed in Table I. If not specified, assume batch size for each neural network is 5. Neural Networks were trained on the full set of 66,000 data points. Random forests were trained on a smaller subset of the data: 2,000 sentences, each corresponding to a 14x1024 array taken from the GPT-2 final layer after the sentence was fed through the model once. (The size of the full data set was computationally prohibitive in this application.) The featured random forest was the best-performing from a hyperparameter grid search. Accuracy was computed on a set of validation data kept separate from training and test data (or from the out-of-bag score in the case of random forests). The best-performing

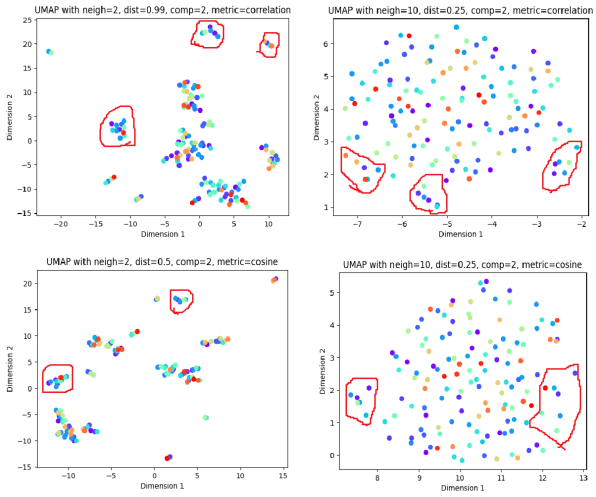

Fig. 2. We applied many UMAP configurations to a small subset of the data set to test whether UMAP could cluster data points with similar labels into the same clusters. Though this method did not successfully sort GPT-2 array inputs by their sentence classes, we noticed repeated structures in the clusters (groups of the same arrays that were consistently clustered together across many different parameter configurations).

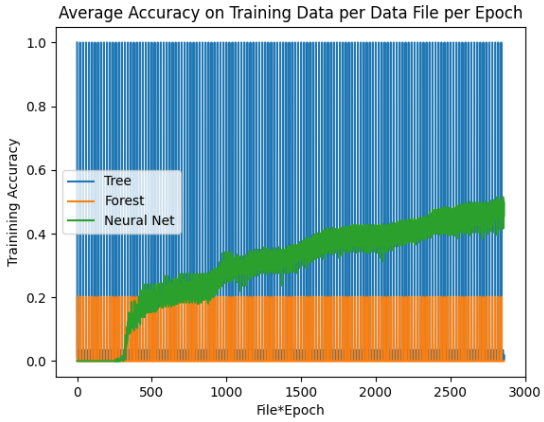

Fig. 3. Training accuracy timeline for 200 epochs over a data set divided into 15 files. The rapid oscillations of the Decision Tree and Random Forest classifiers' accuracies in this figure demonstrate their inability to generalize across batched data. Though the tree-based methods failed to learn the complex classification task, the neural network clearly learned. The random forest was trained using 500 trees with a maximum depth of 50. The featured neural network used seven linear layers with ReLU activations.

network was the original feed-forward architecture, classifying activations derived from offensive or non-offensive sentences with 91.6% accuracy.

It is worth noting that Davidson et al. [6], whose labeled offensive speech data we used, accomplished comparably high precision and recall in a similar classification task by extracting linguistic features from sentences. Though the current work focuses on neural language representations, linguistic feature extraction is a possible representation method to achieve similar classification results.

Though the random forests did not learn the classification

Network type	Notable parameters	Acc.
Original feed-forward	3 dense layers, 112 neurons, batch size 5	.9138
Original feed-forward with larger batches	**3 dense layers, 112 neurons, batch size 20**	**.9155**
Original feed-forward trained on 14x1024 arrays	3 dense layers, 112 neurons, data shape 14x1024	.8959
Wider feed-forward	3 dense layers, 1792 neurons	.9084
Shallower feed-forward	1 dense layer, 64 neurons	.8822
Deeper feed-forward without skip connections	7 dense layers, 1016 neurons	.9029
Deeper feed-forward with skip connections	13 dense layers, 1071 neurons, residual connections	.9015
Convolutional neural network	9 layers, 7 batch norms	.9008
Random Forest	Max depth 4, max 122 features, data shape 14x1024	.8768

TABLE I
RESULTS FROM VARIOUS CLASSIFIERS. THE ARCHITECTURE REFERRED TO AS "ORIGINAL FEED-FORWARD" IS A SIMPLE NEURAL NETWORK OF ONLY 3 DENSE LAYERS AND ONLY 112 NEURONS.

task as well as the neural networks, a feature importance analysis yields interesting results. We ran the two most important features found by the random forest featured in Table I through a Support Vector Machine. (See Figure 4.) The important features learned by the Random Forest separate the input data almost perfectly linearly.

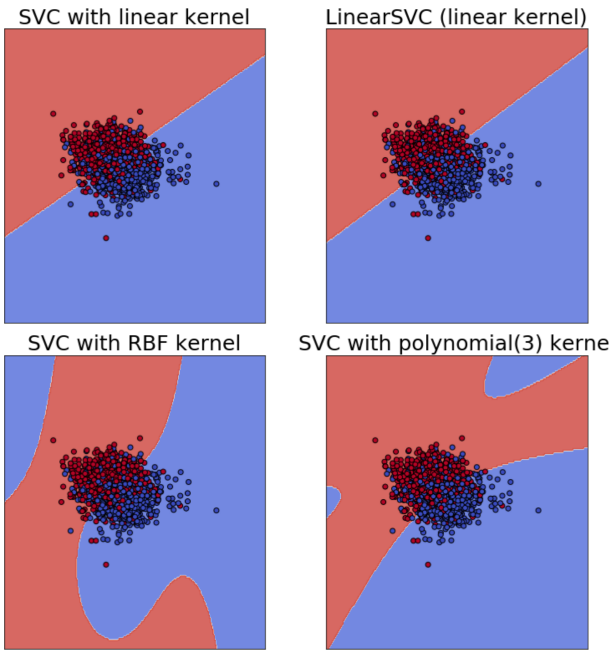

Fig. 4. Feature comparison using a support vector machine.

A. Classification of Deep Semantic Meaning

We performed more experiments using a data set with sentences sorted into two groups determined by the presence or absence of the word "cat" (i.e. cat-sentences vs. non-cat-sentences). This task was even easier for networks to learn. A neural network using the same architecture as the best model featured in Table I was able to attain 99.9% accuracy on validation data for the cat-classification task. On the surface, this may not seem incredibly impressive, as classifying whether or not a sentence contains the word "cat" can be accomplished by a simple regex search. But results show that the deep representations of GPT-2 hidden layers reveal encoded semantic information beyond the word level. (See Table II) This has implications for model interpetability and controlability, as it allows us to perceive when GPT-2 may be "thinking" about cats, even if the word "cat" is absent.

Sentence	Class	Model output
dogs and cats prefer to play together in packs with their cubs	CAT	1.03
children prefer to play together in groups with their toys	NO CAT	.40
children prefer to play together in groups with their cats	CAT	1.00
the film was set in the seventeenth century. A time of war-torn	NO CAT	-.07
the film was set in the seventeenth century. A time of small and large cats everywhere	CAT	.72
the very feline tiger purred and cleaned her tail and whiskers for her cubs	CAT	.96
the very human man groaned and cleaned his hair and mustache for his kids	NO CAT	.01
the very human man groaned and cleaned his hair and mustache for his cats	CAT	.99
she had feline habits and purred and meowed often	CAT	.97
the little furball meowed, grabbed her cub, and slinked away	CAT	.86
the little dude yawned, grabbed his friend, and skipped away	NO CAT	.12

TABLE II
"CAT" CLASSIFIER RESULTS. A MODEL OUTPUT OF 0 CORRESPONDS TO A NOT-CAT CLASSIFICATION, AND A 1 CORRESPONDS TO A CAT CLASSIFICATION. NOTE THAT CERTAIN OF THE SENTENCES, THOUGH THEY DO NOT CONTAIN THE WORD CAT, ARE VERY FELINE IN NATURE AND THAT THE MODEL EASILY PICKS UP ON THIS.

B. Language Model Hidden States as Semantic Representations

We ran further experiments to compare the success of this classification task when using GPT-2 hidden layers as sentence representations versus using a more standard and widely used method for numerical sentence representation, Google's Universal Sentence Encoder (U.S.E.) [8]. As it turns out, both neural network and random forest models, after hyperparameter grid searches, were unable to classify the sentence embedding vectors corresponding to sentences from our offensive/non-offensive data set (while, as we discussed earlier, the models learned the task very well when GPT-2 hidden layers were used). Perhaps the vector sentence representations from U.S.E. do not contain enough fine-grained semantic information for a model to parse whether a sentence is offensive. It seems in this particular application the inner layers of a large language model are far better text representations than text embeddings themselves. This has implications to the possible utility and preferability of language model parameters as text representations in some applications.

See Table III for results. We ran a number of experiments to compare performance of offensive-speech classifiers using sentence embeddings from Universal Sentence Encoder and GPT-2 hidden layer activations. The neural architectures represented are similar to those from Table I except that they use an added Sigmoid activation function after the final neural layer to ensure that outputs will be between 0 and 1. All experiments used a batch size of 5, and each model architecture underwent a grid search of parameters (learning rate for neural networks, maximum depth and maximum features for random forests) to find an optimal result. Note that using sentence embeddings as semantic representations rendered it impossible for the model to learn the task. Unlike random forests in our earlier analyses, random forests in the grid search for this result were trained with the entire offense/non-offense data set. (The low dimensionality of U.S.E. embeddings allowed for use of the larger data set.)

Representations used	Network type	Notable parameters	Acc.
GPT-2 activations	**Original feed-forward**	**3 dense layers, 112 neurons**	**.9117**
U.S.E. embeddings	Original feed-forward	3 dense layers, 112 neurons	.5000
U.S.E. embeddings	Deeper feed-forward	6 dense layers, 504 neurons	.5000
U.S.E. embeddings	Random Forest	Max depth 3, max features: "auto"	.4401

TABLE III
COMPARISON OF PERFORMANCE ON CLASSIFICATION TASK USING U.S.E. EMBEDDINGS AND HIDDEN LAYER REPRESENTATIONS FROM GPT-2

C. Choice of Layers

It is worth noting that although we analysed the first and last layers of GPT-2 for the purposes of our classification, other layers of the network may also be worth exploring. A feature importance analysis on all extracted layers from the small GPT-2 provides some insight into which layers may correlate more or less with the input text. See Figure 5. In applications where GPT-2 hidden layer activations may be used as text representations, it may be worthwhile to consider multiple combinations of extracted layers to find one that works best for the application at hand.

IV. Conclusion

Classification of the inner layers of otherwise uninterpretable language models is a matter of importance for the capabilites and uses of language models in the future. Classifiers that perform this task are an integral part of recently developed methods for language model control [4] [3]. Based on our experiments, it appears that simple, feed-forward neural networks without residual connections are adequate for classifying hidden-layer activations of the GPT-2 pre-trained language model. Neural models trained simply on sentences labeled at the word level are able to glean semantic information beyond the word level from GPT-2 deep hidden representations, allowing the prospect of model interpretability and

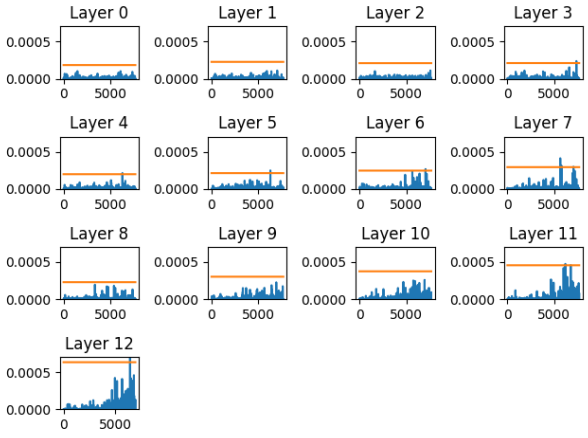

Fig. 5. Feature importance evaluation across GPT-2 layers. Feature importances were generated by a random forest trained to classify cat- and non-cat-sentences using all 13 of the layers in the small GPT-2. In the graphs, blue represents the importance levels of the features throughout each layer, while the yellow lines are the average importance for each layer scaled by a set constant. It seems that the final layer is the most responsive to input text and thus likely the most valuable for classification purposes (at least with regards to the cat-classification task).

insight into what the GPT-2 model is "thinking." Furthermore, empirical evidence shows that these deep representations from GPT-2 hidden layers may in fact be useful as embeddings themselves and may be preferable to traditional embeddings in some applications.

V. Acknowledgements

We would like to acknowledge the contributions of Dr. Emily Evans and the Applied Computational Mathematics students at Brigham Young University for their contributions to this work. We also acknowledge the contribution of Twitter users and CrowdFlower workers who labeled many of the sentences in our offensive and non-offensive text collection.

References

[1] 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.
[2] Gina Neff and Peter Nagy. Talking to bots: Symbiotic agency and the case of tay. *International Journal of Communication*, 10:4915–4931, 10 2016.
[3] 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.
[4] Zachary Brown, Nathaniel Robinson, David Wingate, and Nancy Fulda. Towards neural programming interfaces. In *Advances in Neural Information Processing Systems*, volume 33, 2020.
[5] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.
[6] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language, 2017.
[7] Toxic comment classification challenge. Accessed: 2020-03-20.
[8] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.