CS182 Final Project reviews and response (part 2)

Joyee Chen, Kayla Lee, Hiva Mohammadzadeh, Chunyuan Zheng *

May 14, 2023

Abstract

Part 0: This is an extension of the paper **Interpretability in the Wild**, a paper tackling transformer interpretability with a focus on finding numerical ways to measure the differences that is within the output of a transformer such that these differences can be explained in a way backed by logical reasoning. We also provided an additional focus on reinforcement learning techniques such as PPO and policy gradients, to bring the 2 fields together and give students a new perspective on how to combine 2 important human-centered tasks (interpretability and adjustment based on feedback) on transformers.

PART I: Reviews

Reviewer 1

Question 1: Content and Correctness

The code for the most part is correct. There is a bug that creates inconsistent outputs in one of the notebooks, as noted by the authors. However, otherwise, the code solutions are correct. Furthermore, the problems capture the essence of the concepts of the paper and the mathematical solutions are correct. There is only a Latex export of the solutions and no pdf of the assignment. At times, I did think this homework would take over 1.5-2 hours, but that may vary. Small improvements are needed, maybe cutting down on some questions that require more reading, fixing the minor bug in code, and export pdfs accordingly.

Question 1 - grade Small improvement needed

Question 2: Scaffolding (Option 1).

I think the assignment itself has good scaffolding and is self-contained. However, at times I did find that there could be more comments in all the jupyter notebooks to add more context, especially q5. Regarding the use of external packages, it was great and totally self-contained and instructions were very clear here.

Question 2 - grade: Small improvement needed.

Question 3: Readability/Clarity (Option 1). If this paper is not Option 1, write NA.

I loved how the analytical questions (like question 2) explored the paper and had us delve into it. Since the paper is long, I do think that it was vital that the section numbers were present, making it easier for the student. I liked the Motivation and Background subtext for q3 and q4.

Question 3 - grade Excellent work, no actions needed.

*Authors with Student IDs: Joyee Chen (3035678411), Kayla Lee (3037293229), Hiva Mohammadzadeh(3036919598), Chunyuan Zheng (3036580418). Authors named alphabetically, equal contribution for all

Question 4: Commentary on HW (Option 1). If this paper is not Option 1, write NA.

The commentary on the HW a was very strong. It gave strong reasonings as to why design choices were made. They justify the concepts of the homework in a clear and concise way.

Question 4 - grade Excellent work, no actions needed.

Question 5: Going above and beyond (Option 1).

Not too sure on how to make this HW much better. The assignment is very solid as it is; if anything some subaparts could be cut out since the assignment is fairly long. Nevertheless, adding more visualizations and context to coding questions never hurts.

Question 5 - grade: Small improvement needed

Reviewer 2

Question 1: Content and Correctness

Coding questions engage with selected key concepts of transformers and RL well, but are not very easy to follow as the solutions unreasonably rely on library-specific functions that the student may not be familiar with. For instance, the gym question expects students to know the gym step function and the Q5 question expects students to be familiar with the shap library without providing sufficient guidance. The solutions provided are correct but the notebooks are difficult to follow as a student and could use further background information to motivate key concepts. The policy gradients notebook has errors at the end of the notebook. There are analytical questions with good solutions. Latex files compile correctly. The problems in ”2. Understanding Interpretability of Transformers” are difficult to follow because the language is rather colloquial and ideas seem disorganized. The intended time per section also seem generally underestimated since the topics covered in this question are very broad, the students are required to read sections of the paper, and notebooks rely on library-specific functions not covered in class. It feels that this homework would take more than 2 hours for a student to complete. Question 1 - grade Small improvement needed

Question 2: Scaffolding (Option 1).

Project provides an abundance of scaffolding with three separate notebooks to engage materials. Code students are implementing is well-motivated and hints are provided to support students. However, the exercises unreasonably rely on library-specific information from gym and shap that students may struggle with, retracting from the learning goals of the notebook. The Question 3 and 5 notebook is well explained, but it is not super clear what is being accomplished in Question 5 notebook (could use further explanation and guidance). Sanity checks are absent and could be used for further clarity for student.

Question 2 - grade: Small improvement needed.

Question 3: Readability/Clarity (Option 1). If this paper is not Option 1, write NA.

There are spelling and grammar errors that should be checked with a standard spelling/grammar checker (grammarly). With the exception of Question3 and 4, the homework assignment is generally difficult to follow. The homework starts with no background about transformers. Question 1 ”Understanding the Transformer Architecture” is unreasonable to complete without a word bank or additional guidance. The language in Question 2 ”Understanding Interpretability of Transformers” is not homework-formal and the ideas are difficult to follow.

Question 3 - grade Medium improvement needed.

Question 4: Commentary on HW (Option 1). If this paper is not Option 1, write NA.

The motivations are well-explained. The assignment engages with reinforcement learning concepts although they are not presented in the paper. The commentary is useful but some of the paper’s core topics are not mentioned (e.g. indirect object identification, circuit analysis of model’s computational graph). Overall, the learning goals are achieved in the homework.

Question 4 - grade Excellent work, no actions needed.

Question 5: Going above and beyond (Option 1).

The application of applying RL to finetune language models goes above and beyond the paper. The visualizations presented in in Question 5 notebook are very interesting. Question 2 ”Understanding Interpretability of Transformers” assess the core ideas of the paper, but asks the students to read specific sections. Providing simplifying summaries of the paper in this section could really aid. Question 5 - grade: Excellent work, no actions needed.

Reviewer 3

Question 1: Content and Correctness

These problems did a good job of building the intuition to understand the notion of transformers and how circuits can relate to models.

Question 1 - grade Excellent work, no actions needed.

Question 2: Scaffolding (Option 1).

The question is detailed and intricate in both its mathematical derivations and coding. However, Q3 seems like it would take a lot more than 30 min to understand and solve.

Question 2 - grade: Small improvement needed.

Question 3: Readability/Clarity (Option 1). If this paper is not Option 1, write NA.

The homework is self contained.

Question 3 - grade Excellent work, no actions needed.

Question 4: Commentary on HW (Option 1). If this paper is not Option 1, write NA.

Commentary on homework is sufficient.

Question 4 - grade Excellent work, no actions needed.

Question 5: Going above and beyond (Option 1).

The visualizations in the final notebook went above and beyond to demonstrate the concepts being explained. Thoroughly enjoyable.
Question 5 - grade: Excellent work, no actions needed.

Reviewer 4

Question 1: Content and Correctness

No PDF provided. The Latex compiled with numerous warnings and errors, though that may be caused by my local compiler? The parts on interpreting the transformer were borderline unintelligible and quite abrasive to read. From what I managed to understand, it seems correct on the whole, even if the questions didn't make much sense. The RL questions again seemed correct to me, though I believe the on vs off policy solution should offer significantly more detail on the pros/cons than was given. The Policy Gradients notebook failed to run at the first cell (imports).

Question 1 - grade Small improvement needed

Question 2: Scaffolding (Option 1).

When you link resources, please indicate whether the student is expected to read them. Much of the homework makes little sense without having read the paper or the 188 notes beforehand, since you frequently fail to explain terms, concepts, and symbols adequately, especially in the beginning.

On the whole, I you should assume a lower level of familiarity with the material unless you require those readings.

Question 2 - grade: Medium improvement needed

Question 3: Readability/Clarity (Option 1). If this paper is not Option 1, write NA.

The first section was very unclear and difficult to understand. The later parts were quite clear. Also, please check your grammar. The incomplete sentences and sentence fragments are quite irritating.

Question 3 - grade Small improvement needed

Question 4: Commentary on HW (Option 1). If this paper is not Option 1, write NA.

Very good.

Question 4 - grade Excellent work, no actions needed.

Question 5: Going above and beyond (Option 1).

You attempt to pull together wide ranging ideas in a way that could be very impactful if it were slight better implemented.
Question 5 - grade: Excellent work, no actions needed.

PART II: Responses

Reviewer 1

- We cut down on question 4 and made the coding part optional so that this homework takes 1.5-2 hours; We cut down on the questions that need so much computing.

- We have included additional context and visual aids to aid students in comprehending the coding questions more effectively. Furthermore, for question 5, we have integrated conceptual questions and fill-in-the-blank coding questions. If we had no time constraints, we could have investigated various interpreters, however, we have only included the SHAP interpreter for this task.

- We addressed all the compiling issues and bugs for policy gradients.

Reviewer 2

- What we feel regretful: we could have dived deeper into algorithms but it was out of scope for this class. If interested, CS285 is a good continuation to some of the concepts here. Also, against charges that "the language is rather colloquial", one of the particularly egregiously casual questions got replaced.

- For question 2, we addressed all the formatting issues, and added more quantitative questions.

- For question 1, we added the word bank for what to write in the boxes. For the reviewer's criticism in last sentence of "Readability/Clarity", I replaced one of the excessively informal questions.

- For question 2 a definition of the indirect object identification (IOI) problem was added.

Reviewer 3

- We fixed question 3 to take no longer than 30 minutes.

- For question 4, we added more conceptual questions and observations and made it more exploratory. If we had no time constraints, We could have let students use more libraries.

Reviewer 4

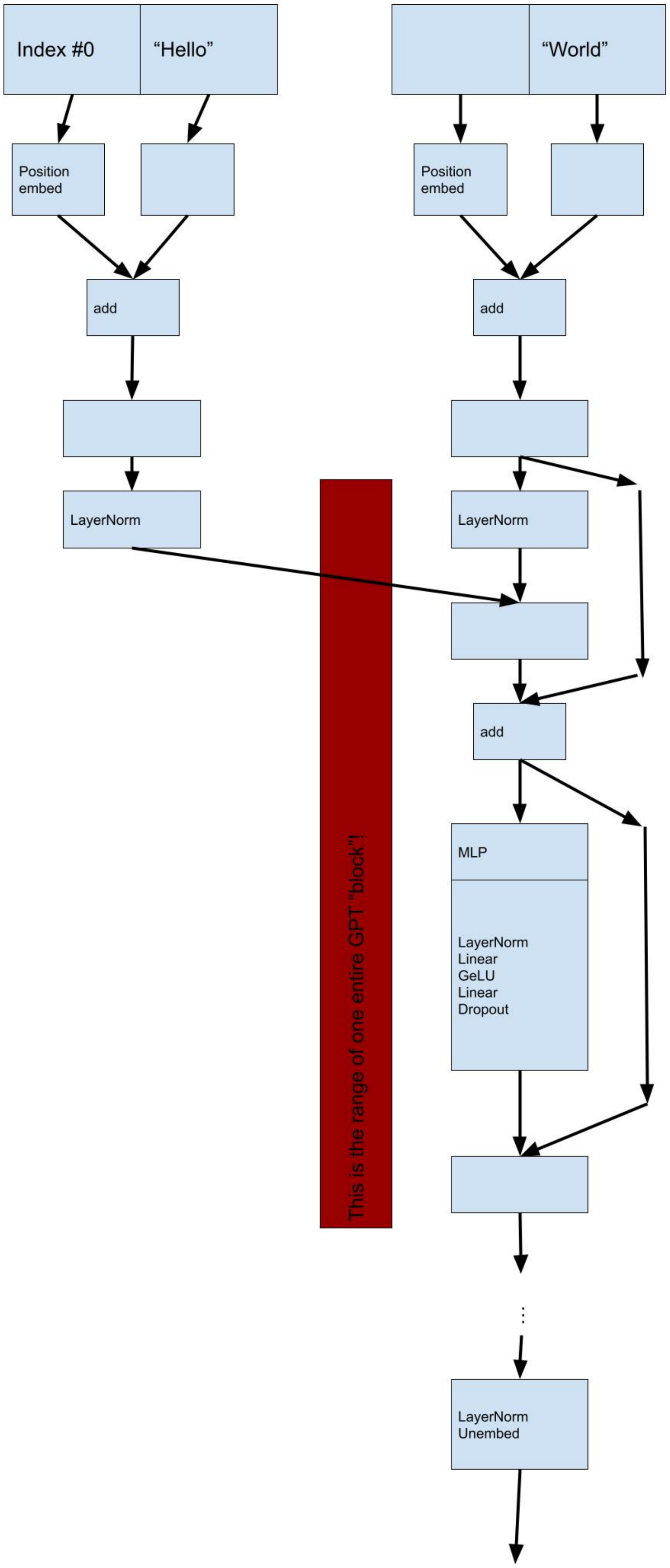
- We provided the PDF file in the zip file.
- There are no bugs or errors when we run the latex file.
- For every resource that we mention, we require the student to either use it as a tool to help them with coding or to understand the topics.
- We fixed the first section to not be unclear and difficult. We added the vocabulary bank for question 1 and we made the questions in 2 more comprehensive, as well as added brief summaries.
- We checked grammar and we don't seem to have any grammatical errors.

PART III: Final Submission

1 Understanding the Transformer Architecture [3 minutes expected time to complete]

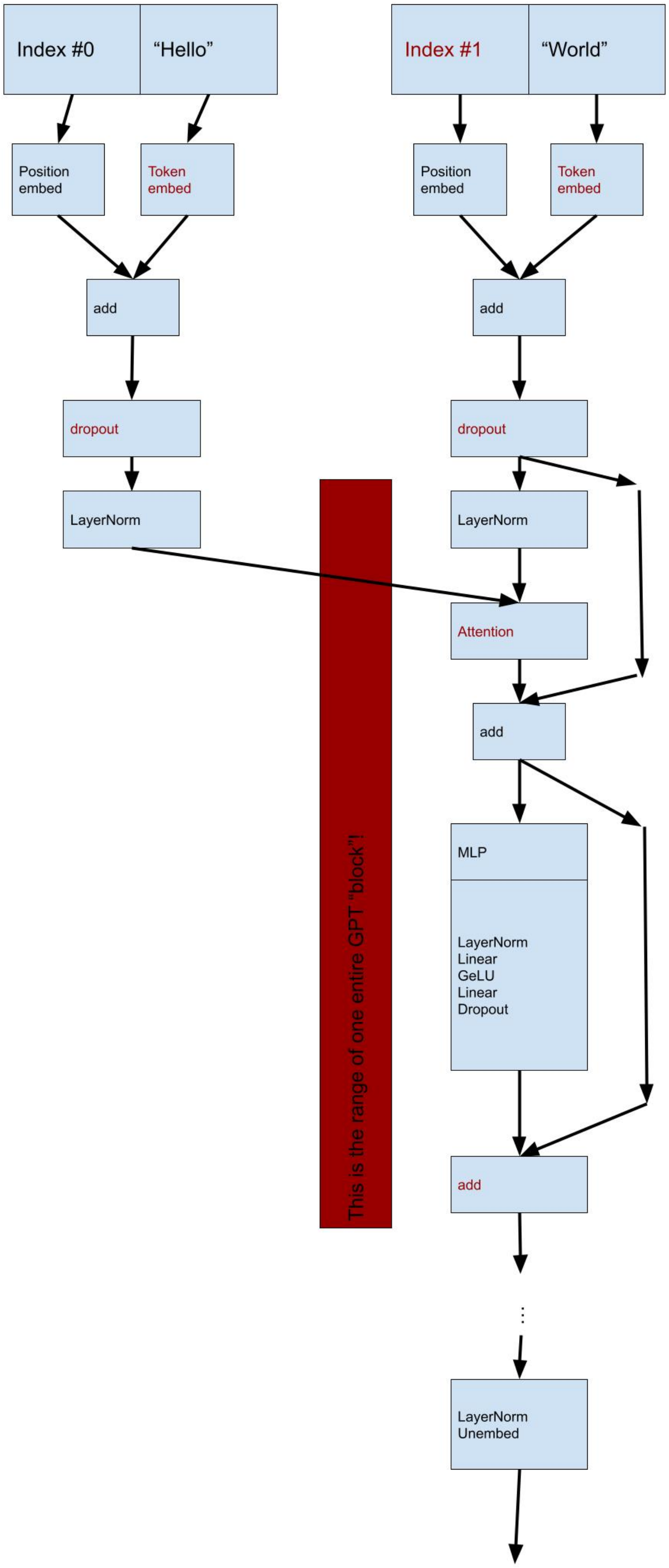
A semi-completed diagram of a transformer is below. **Fill in the blanks.** A vocabulary list you must use:

- Dropout
- Attention
- Index 1
- Token embed
- add



(Source: unknown creator, from printed document)

SOLUTION:


Index #0
“Hello”
Index #1
“World”
Position embed
Token embed
Position embed
Token embed
add
add
dropout
dropout
LayerNorm
LayerNorm
Attention
add
MLP
LayerNorm
Linear
GeLU
Linear
Dropout
add
⋮
LayerNorm
Unembed
This is the range of one entire GPT “block”!

2 Understanding Interpretability of Transformers [30 min]

It is natural to use transformers as a black box, not caring about what they do so long as they work. But there are so many reasons why we would want to rip off the black box of the transformer veil.

This part of the problem set is heavily based off **Interpretability in the Wild**, the paper: https://arxiv.org/abs/2211.00593. Later on, we will implement elements of reinforcement learning into this.

2.1 What's something interesting we can do when measuring attention outputs? (an observation in language...)

Consider the sentence "When Mary and John went to the store, John gave a bottle of milk to...". Is the answer "Mary"?

From a linguistic perspective, this question is a test of pattern matching the S/IO pattern. John is the S (Subject) of the sentence, while Mary is the IO (indirect object) of it...and the key here is that an intelligent speaker ought to connect the S part to the IO part. In general, this entire problem is called the "Indirect Object Identification"/IOI problem.

More precisely, the intelligent speaker ought to be given a near-completed sentence of the form above, in (S/IO) ... S order, and then correctly predict the next/final token to be IO.

For a more sophisticated pattern, closer to something you would feed into a regex, we can try (S1/IO)...S2..., where S1 and S2 represent the first and second occurrances of the same subject token. (note that S1 and S2's position encodings will be different, of course)

Based on the Mary/John example, one might be tempted to think of the desired pattern (that we want to extract) as "S1...IO...IO...?", replacing the (S1/IO) term. Why is this a bad idea?

SOLUTION:

We want the desired learning to generalize to possible cases of "IO...S1...IO...S2" as well – we don't want to reject out of hand the case "When **John** and **Mary** went to the store, John gave a bottle of milk to..."

2.2 [No questions here] And how can we represent that interestingness? (circuits and performance metrics)

But even though we have a rough pattern above to test, the translation between "(S1/IO)...S2..." to the interactions of objects of type "neurons" in a NN is far from trivial.

The bridge between the two concepts is the idea of a **circuit**.

Circuits are basically graphs where the nodes represent any computational element or discrete object within a neural network (like embeddings and attention heads) and edges represent what information gets passed between those discrete objects (like attention).

On a somewhat related question: we've defined what the circuit is, but how can we find out how well it performs? Scratch that – that's for the next section. The real move is, how can we even define how well it performs? The "Interpretability in the Wild" paper defines two ways:

- Logit Difference: the logit of the correct option minus the logit of the incorrect option. Here, logit(IO) - logit(S). Higher logit differences correspond to a "better" model.

- IO Probability: the IO token's absolute probability of occurring, for that given model's predictions. Naturally, higher IO probability corresponds to a "better" model too.

To connect circuit theory and performance measurement together: we can actually say that all models M(x) are defined to be functions taking in inputs x and pushing out **logits**.

All that still eludes another important point: how can we connect circuits to inputs? Knockouts are the answer: just what they sound like, they are the act of "knocking out" a set of nodes in a computational graph, by somehow making those nodes' signals worthless. Then measure the logits of the resulting knocked-out graph. Knock-out process, applied to M, creates C: so where once inputs x can get passed into M to get M(x), now those same inputs can get passed into C to get C(x).

2.3 But how can we test where the circuits come from? (theory of testing, ablation)

For questions 2.3 and onward, we are going to do a paper analysis in the style of Discussion 12 Q2: We will ask that you read the respective sections of the paper before completing the questions, without the need for background info in each section of this pset, though brief summaries will be provided.

This section covers section 2.1 of the paper.
In this section we introduce the idea of knockouts as a way to create circuits, and motivate ablation, in particular defending mean ablation, to implement those knockouts.

Conceptual questions:

- What are some problems with just using zero ablation, problems mentioned in the paper?

SOLUTION:
The paper claimed that choosing 0 for ablation-value does not actually guarantee the "neutrality" we want: some nodes might "rely on the average activation value as an implicit bias term", so we might want to choose averages there instead of zero to guarantee as much information is "erased" in ablation.

2.4 But you haven't shown me HOW we come up with the circuits in the first place! (circuit discovery through iterative head-tracing)

This section covers sections 3 and 3.1 of the paper.
In this section the authors propose a circuit that will implement the specific IOI/"Mary John" task from earlier. It classifies all the important attention heads into three classes depending on function ("Duplicate Token Heads", "S-Inhibition Heads", and "Name-Mover Heads"). Then most of the subsections are about recursively working backwards from the output to discover (not just confirm) those classes: Section 3.1 in particular discovers the Name-Mover Heads, using a two-step process (path patching, followed by inspecting attention-probability plots along the axis of certain projections related to Mary or John choices).

Conceptual questions:

- In American English there's a concept of "double negatives" (see `https://en.wikipedia.org/wiki/Double_negative` for examples), where one "negative", or phrase that indicates "not", is located next/near to another "negative" in such a manner that they're *supposed* to cancel each other out. Following the theorizing that the paper authors did, can you propose a plausible circuit (with named heads) that can detect double negatives? (Of course you don't need to test it!)

SOLUTION:

One idea: We can visualize one head determining the number of "negative" words, a second head determining the positions of "negatives", a third head determining whether certain identified negatives are supposed to cancel out certain other identified negatives, etc. Naturally, the second head (which we can call the "negatives-positions-head") will output its result to the first head (which we can call "num-of-negatives-head"). Both will output to the third head (which we can call "negatives-pairs-head"). In particular, we can say that the third head will get triggered only when the first head has number of negatives greater than 1.

2.5 And the rabbit hole goes deeper... (iterative tracing, continued)

This section covers sections 3.2 to 3.4 of the paper.
The 3.2 subsection detects S-Inhibition heads by using the 3.1 process and tracing back from the Name-Mover heads; likewise, subsection 3.3 goes off of 3.2 to detect the duplicate token heads. The fourth subsection tries to explain a secondary effect, the Backup Name Movers Heads, as a side-effect of dropout.

Conceptual questions:

- Describe the role of position signals in sections 3.2 and 3.3. How would our analysis be changed, or fail, if the author didn't come up with the idea of position signals and just used signals for tokens?

SOLUTION:

There role of position signals is to encode the position of each word as it passes through the various attention heads. On a broad level, if the author just used token signals, then the resulting system will be blind to the ordering of words/tokens in text, making it far less effective.

2.6 How do we test that circuit against the best circuit that it can possibly be? (experimental validation and the three conditions)

This section covers sections 4 and 5 of the paper.
While section 3 deals with how to discover parts of a model, section 4 focuses on how to test a model and measure its success experimentally or rigorously (using the principles of faithfulness, minimality, and completeness); the fifth section is a general summary of the paper.
Conceptual questions:

- "The faithfulness, completeness, and minimality criterions for the purposes of validating circuits against models here are metaphorically equivalent to linear algebra's concepts of span, linear independence, and basis." To what extent is this true?

SOLUTION:
Regarding the first metaphor, we know from basic linear algebra that a basis of vectors has to fulfill span and linear independence conditions. Span conditions ensure that a basis set of vectors actually covers the subspace that you want to model it. Also, linear independence ensures a sort of "uniqueness" among each representation of the subspace – it can't be represented by two different lin. comb. of vectors.
Now replace "subspace" in the above explanation with "real world environment" that you want to model. This brings us closer to the faithfulness, completeness, and minimality criterions. Completeness can resemble span: they ensure the real world, or at least as much of the real world as possible, can be represented in at least one way in the model. Minimality can resemble linear independence: the idea each representation in the real world model range can be represented in just one way, the simplest way. But the analogy to lin. alg. might break down for faithfulness, because in lin. alg. it is implicitly assumed that a real world vector will equal, not just approximate, a lin. comb. of basis vectors.

3 Understanding Policy Gradients [40 min + 20 min optional work]

Motivation

Although the application of transformer and reinforcement learning (RL) regimes are deemed to be separate in previous years, more recent developments have shown that these 2 fields can be used in the same contexts, as transformers can help with sequential data brought by RL environments. Given the widespread implementation of RL for robotics and other fields of control theory, questions 3 and 4 will provide a deeper connection between RL and transformer, giving us a more comprehensive outlook of transformer interpretability in a larger picture.

Contents regarding math cited from CS285's lecture 5

Introduction

In the field of RL, instead of being presented with data points and their corresponding labels $\{x_i, y_i\}, i \in \{1, 2, ...n\}$, we instead have a set of representation of states $\mathcal{S}$, a set of actions $\mathcal{A}$, and a reward signal $R(s_t, a, s_{t+1}) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to R$, a function that takes in the previous state, the action, and the future state that outputs a reward. Therefore, instead of finding the best parameter θ to minimize a loss objective $L_\theta(x, y)$, our following objective is:

$$\text{argmax}_\theta J(\theta) = \text{argmax}_\theta E[R(\tau)] \tag{1}$$

Where θ is our chosen policy, $J(\theta)$ is the total expected payoff from our chosen parameter, where we sample trajectories τ from our output policy π_θ.

If you are somewhat unfamiliar with RL, here are some notes from CS188 that discuss vanilla reinforcement learning such as value iteration and Q-learning. Our team heavily encourage you to read them before to brush up some concepts. In deep reinforcement learning, we use a neural network to generate either a reward system or the policy mechanism, and in this case, we will focus on deep RL for policy generation.

3.1 Conceptual Understanding of Policy Gradients

Please answer the following questions in short sentences. You may use mathematical concepts as aid, but we do not need full mathematical proofs.

i. In supervised learning, we would match our results with training labels/data points. However, we cannot access the optimal behavior in reinforcement learning. How does policy gradient account for that?

SOLUTION:

Let's take a look at the core concept of both (vanilla) gradient descent and policy gradients. Gradient descent came from the idea of linearization, where we use first-order derivatives to find the local/global minimum. However, since we do not have what is an "optimal" policy, we would instead use a combination of results from rollouts to infer what is the optimal policy by increasing the likelihood of high-reward policies and lowering the likelihood of those that don't.

ii. An on-policy learner learns directly from the states and rewards from exploration, while an off policy learner learns independently from the state's actions. What are some benefits of learning off-policy vs on-policy?

Hint: consider the computational cost of learning on-policy vs off-policy learning.

SOLUTION:

An on-policy agent can only update its parameters after collecting samples by acting in the environment, making the process very much inefficient due to the lack of ability to parallelize the process. However, one can parallelize such processes in offline training, lowering the training run time.

iii. Consider the case when we have infinite samples corresponding to the distribution at hand, and know the total reward of the system of any action (this is also known as the infinite horizon case). What does policy gradient represent now?

SOLUTION:

Instead of approximating the expected reward function with empirical expected value, what we will have now is the true expected reward function, as the average payoff becomes an integral:

$$\lim_{N\to\infty,t\to\infty} J(\theta) = \lim_{N\to\infty,t\to\infty} \frac{1}{N}\Sigma_{i=1}^{N}\Sigma_{j=1}^{t} r_t^{(i)} = E[R(\tau)] \tag{2}$$

In this case we would have a differentiable function, thus allowing us to use gradient descent. However, the bigger takeaway is that policy gradient is also a 0th order optimization algorithm: instead of having the first order derivatives to dictate the change in loss, we would find "chords" in different points within the range of the reward and find the best sample point by finding the difference between the points and adjust the likelihood.

3.2 Mathematical Understanding of Policy Gradients

Note: some basics of policy gradients have been covered in HW11, this attempts to further your understanding on the mathematical basis of RL. In this case, we will exclusively cover on-policy learning.

i. Suppose we have a policy where the action is based on a multivariate Gaussian distribution: $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t) = \mathcal{N}(\mathbf{f}_\theta(\mathbf{s}_t), \Sigma)$, where $\mathbf{f}_\theta(\mathbf{s}_t)$ is the output of some neural network with parameter θ.

Derive both log$(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t))$ and ∇_θlog$(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t))$

(Hint: One convenient identity that you may use, is that $\frac{d}{d\mathbf{x}}(\mathbf{x}-\mathbf{s})\Sigma(\mathbf{x}-\mathbf{s}) = 2\Sigma(\mathbf{x}-\mathbf{s})$)

SOLUTION:

$$\log(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)) = \log(\exp(-\frac{1}{2}(\mathbf{f}_\theta(\mathbf{s}_t)-\mathbf{a}_t)\Sigma^{-1}(\mathbf{f}_\theta(\mathbf{s}_t)-\mathbf{a}_t)+k)) \tag{3}$$

Where k is some proportionality constant present in all distributions with covariance matrix Σ.

$$\log(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)) = (-\frac{1}{2}(\mathbf{f}_\theta(\mathbf{s}_t)-\mathbf{a}_t)\Sigma^{-1}(\mathbf{f}_\theta(\mathbf{s}_t)-\mathbf{a}_t)+k) \tag{4}$$

Which can be written as

$$-\frac{1}{2}||\mathbf{f}_\theta(\mathbf{s}_t)-\mathbf{a}_t||_\Sigma^2 + k \tag{5}$$

Where in any argmax operations, k can be ignored.

Taking the derivative w.r.t. θ, using the identity we have above, we get:

$$\nabla_\theta\log(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)) = \frac{\partial\log(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t))}{\partial f}\frac{\partial f}{\partial\theta} \tag{6}$$

Solve for $\frac{\partial\log(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t))}{\partial f}$:

$$\frac{\partial\log(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t))}{\partial f} = -\Sigma^{-1}(\mathbf{f}_\theta(\mathbf{s}_t)-\mathbf{a}_t) \tag{7}$$

$$, \nabla_\theta \log(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)) = -\Sigma^{-1}(\mathbf{f}_\theta(\mathbf{s}_t) - \mathbf{a}_t)\frac{\partial f}{\partial \theta} \tag{8}$$

This result shows us that differentiating the log-probability of such a policy with respect to θ does change the parameters of the neural network, helping us to connect with vanilla gradient descent.

ii. In RL environments, we would often set a baseline $b = \frac{1}{N}\Sigma_{i=1}^{N} R(\tau) \simeq E[R(\tau)]$. Prove that substituting in a new $J'(\theta) = \frac{1}{N}\Sigma_{i=1}^{N}\nabla_\theta \mathbf{log}(p_\theta(\tau))[r(\tau) - b]$ is unbiased compared to the original $J(\theta)$.

(hint: A convenient identity we proved in HW11 is that $p_\theta(\tau)\nabla_\theta \log(p_\theta(\tau)) = \nabla_\theta p_\theta(\tau))$

SOLUTION:

$$E[J'(\theta)] = E[J(\theta)] \rightarrow E[\nabla_\theta \log(p_\theta(\tau))b] = 0 \tag{9}$$

$$E[\nabla_\theta \log(p_\theta(\tau))b] = \int_{D_\tau} p_\theta(\tau)\nabla_\theta \log(p_\theta(\tau))b d\tau = b\int_{D_\tau} \nabla_\theta p_\theta(\tau) d\tau = b\nabla_\theta \int_{D_\tau} p_\theta(\tau) d\tau = b\nabla_\theta 1 = 0 \tag{10}$$

The reason a baseline is so important is that it considerably reduces the variance of policy gradients, which is a huge problem for any sample-based policy gradient method. You don't need to know how that works within the scope of this question.

3.3 Implement policy gradient in gym (OPTIONAL)

Please finish the coding assignment in this notebook, where you will implement policy gradients of a lunar lander in gym, an open-source simulator for real-world environments, courtesy to OpenAI. Since we have already implemented policy gradients for classification, this is an optional assignment. If you're unfamiliar with gym, we recommend you to read this article, which is a quick introduction to gym environment.

i. After you have finished your assignment, please upload the training plot here, featuring the average reward per iteration on the y-axis. What are some interesting observations you see?

NOTE: There is one import that fails when you import all packages. You can ignore such problem
SOLUTION:
As provided by the sample solution, the payoff from one's actions between different iterations has a considerably high variance compared to what you'd see in supervised learning before somewhat stabilizing. However, given that we are using 0th degree optimization problem, we have to accept such a dilemma.

ii. Even when training a (theoretically) very effective network, the reward can vary drastically. What does this say about policy gradients, and what does this say about transformer interpretability if we were to use policy gradients on it?

SOLUTION:
Even with preventive methods such as using baselines, policy gradients may still provide inconsistent results due to its high variance, with one of the biggest drawbacks being unexpected spikes and troughs within reward systems. As a result, we must find ways to regulate such variances especially with the wide usage of transformers in reinforcement learning, and one way to deal with it in a qualitative manner is to use transformer interpretability to help understand the structure behind the architecture and not let blackboxes dominate our way of thinking.

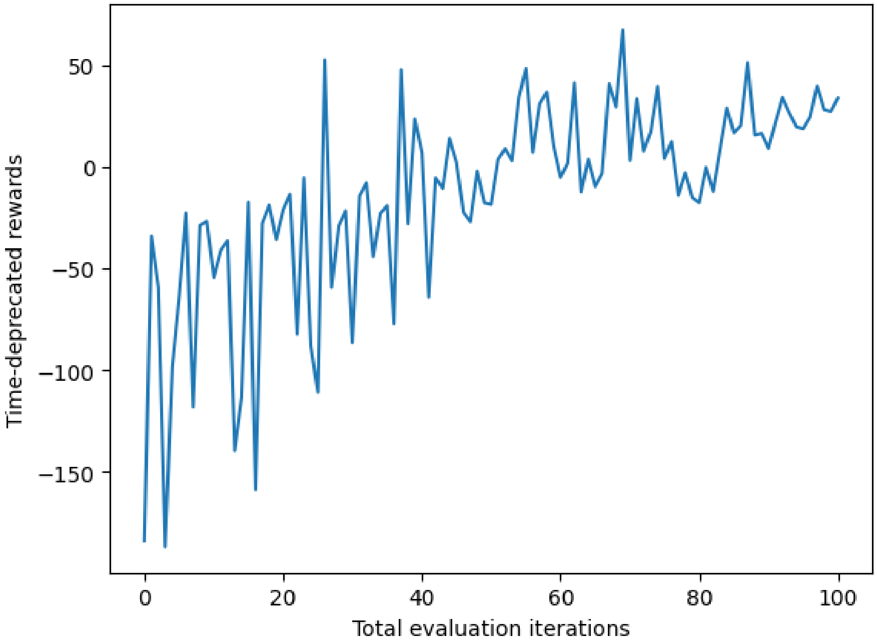


Figure 1: Our team's solution

4 Policy Gradients for GPT-2 fine tuning using online libraries (coding/observe experiments) [30 min]

Background

Hugging face, one of the largest collection of transformer and its related libraries in the world, has majority of its libraries written in PyTorch. In this section, we will introduce new libraries and methods where reinforcement learning can be used in transformer-centric environments as human feedback can be much less mathematically formulated compared to normal loss in token comparisons. More specifically, we will introduce TRL, one of the reinforcement learning-based transformer libraries that is useful for fine-tuning parameters by both learning a reward system and via human feedback.

4.1 Understanding reinforcement learning in transformer fine-tuning

Introduction

To better understand the architecture of fine-tuning transformers using reinforcement learning, let's take a look at the process of the fine tuning first.

i. In PPO, we would use KL-divergence as a part of our optimization model. Explain in words the importance of KL-divergence in the context of reinforcement learning.

SOLUTION:

While KL-divergence is needed for many different types of fine-tuning tasks, it is more important for reinforcement learning settings due to the high variance brought by the training data. A significant change in the weights between the current model and the previous model may not be representative of the actual difference between the "correct" fine-tuned model and the baseline model. Thus KL divergence can be used as a regularization parameter in this case and help mitigate such a difference.

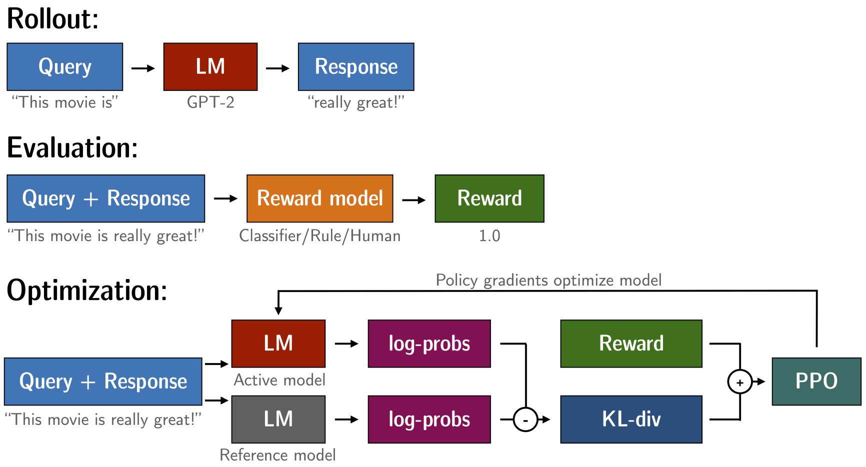


Figure 2: An RL-centric transformer fine-tuning process. source

4.2 Fine-tuning GPT2 using TRL's Policy Gradients method

In this question you will get familiar with how GPT2 can be fine tuned using the TRL's policy gradients. Please read the documentation for TRL and get familiar with the syntax. We would've loved to have you go through the process of fine tuning yourself, but time constraints won't allow us to.

Instead, for this question, please take time to read through this example of fine tuning GPT2 to generate positive reviews on HuggingFace or (Github). We also provided an interactive example that dives into PPO deeper, so please run these code segments and answer the following questions:

i. Suppose only 3 people have provided a feedback to your language model. What are some of the constraints on traditional fine tuning methods? How well did PPO fine tune this model with just a few inputs?

SOLUTION:

While human have the capability to generate lengthy feedback, most of them will only rate the response provided by the language model using simple positive/negative signals. Compounded with the fact taht very few people responded to the outputs, this prevented us using full fine-tuning as a way to optimize our model. The feedback being only in the form of yes/no questions (like the ones you see on ChatGPT after it provides you an answer) also prevented us from soft-prompting or hard-prompting, as the gradient from the reward signals cannot flow back to the prompts.

With just a few feedback entries, we see that the language model adjusted effectively to the new inputs and provided positive sentiments effectively.

ii. Experimentally and conceptually, what similarities do you notice between this fine-tuning approach (using policy gradients via PPO) and the traditional approach?

SOLUTION:
Answers may vary. As we can notice, both techniques aim to optimize the pre-trained Transformer model for a specific task by adapting it to a smaller, task-specific dataset. Both regular fine-tuning and TRL use some ways to regularize the models from being changed too much, with TRL using KL-divergence as a form of regularization while fine-tuning may be freezing parts of the model or use methods such as low-rank updates to prevent the model from changing too much.

iii. What differences do you notice between this fine-tuning approach (using policy gradients) and the traditional approach?

SOLUTION:

Answers may vary. Regular fine-tuning takes a pre-trained model and further trains it on a new dataset specific to a new task. During fine-tuning, the pre-trained weights of the model are updated using the new dataset to learn new task-specific patterns and features.

On the other hand, TRL defines a reward function that measures the model's performance on a new task, and then using policy gradient methods optimizes the model's weights to maximize this reward. Unlike regular fine-tuning, TRL fine-tuning focuses on improving the model's ability to make decisions and take actions in response to inputs, rather than simply improving its ability to recognize patterns in the data. Fine tuning with TRL can allow the model to improve its performance during deployment. This is particularly useful in interactive tasks such as dialogue generation or recommendation systems, where the model must continually adapt to new user inputs and feedback.

In regular fine-tuning, the weights are updated based on the gradients of a loss function that compares the model's predictions with the ground-truth labels for the new task. In TRL policy gradients fine-tuning, the weights are updated based on the gradients of a reward function that measures the model's performance on the task.

iv. (Optional) If this approach to fine-tuning is interesting to you, feel free to use it as a guide to explore other GPT-2 tasks that this approach can be applied to.

SOLUTION:

We do not have a set solution for this. Students should explore on their own to see any interesting phenomena!

5 Testing a fine-tuned model with interpretability tools (coding) [15 min]

With a foundation set, you're now ready to observe some intriguing behavior of transformers in terms of its interpretability through experimentation. Assume that we have a fine-tuned transformer model similar to the one we implemented in question 4. Using interpretability tools, we want to analyze the model.

For this question, just finish the coding parts within the notebook and answer the optional questions that are just there to make you think about the visualizations: InterpretFineTunedModel

If your implementation is successful, you will see visualizations consistent with the ones described in the documentation for SHAP.

6 A totally warranted sales pitch [0 min]

If the thought of grokking transformers has you interested, join the Berkeley AI Safety Initiative for Students! `https://berkeleyaisafety.com/` or Papers We Love `https://pwl-berkeley.herokuapp.com/`

PART IV: Team member contribution

Joyee Chen

Joyee wrote question 2, providing both the conceptual basis of interpretability and provided technical guidance on how to navigate through an experimentally heavy paper.

Kayla Lee

Kayla worked on question 5 coding portion. Worked on the commentary and submission files. Kayla did part of 3 and 4. Kayla also helped proofread question 2.

Hiva Mohammadzadeh

Hiva coded the entirety of question 4 and wrote 3 of the 5 short response questions. Hiva also worked on question 5, finding the SHAP library and applying it to all the different tasks. Worked on the submission files.

Chunyuan Zheng

Chunyuan wrote question 3 as well as working with Hiva on question 4, providing the theoretical basis of PPO and writing 2 of the 5 short response questions.

PART V: Link to code/dataset/supplementary materials

- Interpretibility in the wild paper: `https://arxiv.org/abs/2211.00593`

- InterpretFineTunedModel
  https://colab.research.google.com/drive/16ZoZXEPMrpbiDycwQHz5EHiWAcSNfh1o?usp=sharing

- Hugging face for question 4:
  https://huggingface.co/docs/trl/v0.1.1/en/sentiment$_tuning$

- Github for question 4: (https://github.com/lvwerra/trl/blob/main/examples/sentiment/notebooks/gpt2-sentiment.ipynb)

- interactive example that dives into PPO deeper: https://colab.research.google.com/drive/1$_c2OxYks-$
  $EDwjQ3opgye8mYRGopNQp0IscrollTo = syEq - jLVVzwp$

- Policy gradient: Question 3: https://colab.research.google.com/drive/14XhJcfXWhFUR9E$_lfqOOrZD43Zs9hGhzs$
  $Qbo8sWVSupy8/$

  Article for policy gradients: https://blog.paperspace.com/getting-started-with-openai-gym/