Attention Consistency Training

SPAR Proposal
David Demitri Africa

Summary

Attention Consistency Training (AttCT): Instead of matching outputs (BCT) or residual stream activations (ACT), we enforce that the model attends to the same input tokens when processing clean versus wrapped prompts. Specifically, we compute an L2 loss between the attention weight matrices at each layer for the two prompts. The motivation behind this is to target the information-gathering mechanism itself before any computation is done with that information. My hypothesis is that biases and jailbreaks work by redirecting the model's attention toward the adversarial wrapper text, and so training the model to maintain consistent attention patterns that focus on the core question regardless of wrapper, would prevent the bias from entering the computation in the first place. I claim this is mechanistically upstream of both activation consistency (which constrains what is computed) and output consistency (which only constrains the final result).

Motivation

Why Train to Make Models Consistent?

It seems that at least some alignment failures would occur not because the model lacks the incentive to respond appropriately, but because of stochastic error from irrelevant prompt features or some knotty entangled latent representation. When a model refuses a direct request to build a bomb but complies when wrapped in a "creative writing" scenario, or when it answers a factual question correctly until a user hints at their opinion, the underlying knowledge and values are present, and are just being overridden by spurious cues.

Consistency training/regularisation is promising because it appeals to a fairly simple intuition: if the model already knows how to behave well on clean prompts, can we simply teach it to ignore the adversarial augmentations? BCT and ACT show that this is possible, and it seems the architectural design space here is rather rich. So we should explore!

Also, consistency training is self-supervised, which makes it cheap if you can do it in a computationally efficient way.

Why Attention Patterns?

I claim current evidence suggests attention is the primary mechanism by which adversarial prompts influence model behavior. It seems likely, as attention matrices determine which tokens' information flows to which positions, so if jailbreak text captures attention, it can influence all downstream computation.

This would also be quite nice in a mechinterp sense, as attention patterns are directly observable, so we can visualize exactly what the model is focusing on and inherit some nice preexisting xAI attention map methods. This will require some munging through superposition, which will be hard.

Finally, attention happens before value computation and residual stream updates, so intervening here would prevent bias from entering the computation at all, rather than trying to correct it afterward! Seems promising.

ATCON demonstrates this principle in computer vision, showing that enforcing consistency between different attention map methods (Grad-CAM and Guided Backpropagation) improves both classification performance and interpretability when training data is scarce. So mostly our proposal is to extend this to the setting of large language models, and use it for safety purposes.

There are a couple of arguments why this would be weaker than ACT:

1. Different attention patterns could produce similar outputs if the content being attended to is similar
    a. So this might be the case where attending uniformly vs. attending to two synonymous tokens, not too different
    b. Then you have the case where AttCT "wastes" optimization pressure on differences that don't matter
2. Maybe the picture is like, we end up succeeding in making attention consistent, but the vulnerability lies in MLPs
3. Many different attention patterns could produce the same residual stream activations
    a. If AttCT constrains the solution space more than ACT, then it might be the case that AttCT is too restrictive and hurts performance, or maybe the extra constraint helps

But there are a few reasons why it might be stronger:

1. Attention creates important path-dependence
    a. That is, once attention patterns are set at layer L, they constrain what can be computed at layer L+1, L+2, etc.
    b. You might imagine that even if you give ACT every layer and it tries to match residual streams at every layer, but might be fighting against entrenched attention patterns already
    c. So you might attack the attention itself
2. Trying to match high-dimensional activations might have many spurious solutions. The model could match activations in weird ways that don't reflect the same "reasoning."

    a. AttCT might be easier since matching attention patterns is a more structural constraint

3. And ultimately, it’s like: attention is the selection mechanism, everything else is computation on selected information.
    a. ACT would say something like: "compute the same thing" AttCT would say something like: "select the same information to compute with"
    b. And if you select different information (attend to wrapper vs. core question), it's very hard to compute the same thing. You'd need MLPs to perform complex "correction" operations.
    c. But if you select the same information (AttCT), computing similar things follows naturally