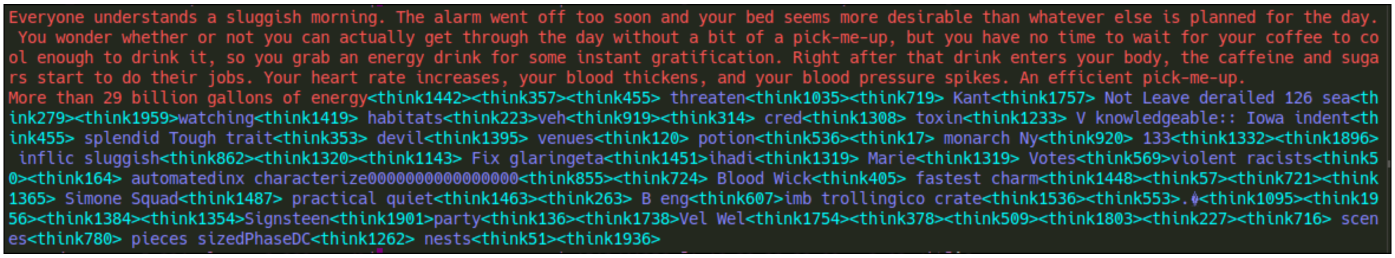
Some clarifications

What exactly is meant by an 'extra token' or 'thinking token'?

- text is turned into numerical data via tokenization. token 0 may be ' a', token 6543 may be ' Christmas' and in gpt2-small, token 50257, the largest token id, is '', a special token for the end of a sequence.
- By 'adding extra tokens', I mean I expand the d_vocab of the model, making the embedding and unembedding layers larger, adding extra rows.
- Now there are larger token id's that the model can take as input and produce as output.
- But these new token ids will never be seen in any pretraining dataset, becuase they don't correspond to text. So what is the meaining of these tokens?
    - They have no textual association. They don't correspond to any string.
    - The model learns to output its normal tokens based on supervision: being told that this token should come next given this input sequence. How will the model learn to use its new tokens?
        - We can make no supervised loss signal for when the model produces thinking tokens, becuase they have no inherent meaning and so we cannot judge them.
        - This is why the RL is needed: we can encourage the model to make them useful by rewarding them based off of how they alter later, supervised loss signals.
- There are several ways one could go about applying this basic idea. Here is an output early in training for one such approach:



```
Everyone understands a sluggish morning. The alarm went off too soon and your bed seems more desirable than whatever else is planned for the day.
 You wonder whether or not you can actually get through the day without a bit of a pick-me-up, but you have no time to wait for your coffee to co
ol enough to drink it, so you grab an energy drink for some instant gratification. Right after that drink enters your body, the caffeine and suga
rs start to do their jobs. Your heart rate increases, your blood thickens, and your blood pressure spikes. An efficient pick-me-up.
More than 29 billion gallons of energy<think1442><think357><think455> threaten<think1035><think719> Kant<think1757> Not Leave derailed 126 sea<th
ink279><think1959>watching<think1419> habitats<think223>veh<think919><think314> cred<think1308> toxin<think1233> V knowledgeable:: Iowa indent<th
ink455> splendid Tough trait<think353> devil<think1395> venues<think120> potion<think536><think17> monarch Ny<think920> 133<think1332><think1896>
 inflic sluggish<think862><think1320><think1143> Fix glaringeta<think1451>ihadi<think1319> Marie<think1319> Votes<think569>violent racists<think5
0><think164> automatedinx characterize0000000000000000<think855><think724> Blood Wick<think405> fastest charm<think1448><think57><think721><think
1365> Simone Squad<think1487> practical quiet<think1463><think263> B eng<think607>imb trollingico crate<think1536><think553>.�<think1095><think19
56><think1384><think1354>Signsteen<think1901>party<think136><think1738>Vel Wel<think1754><think378><think509><think1803><think227><think716> scen
es<think780> pieces sizedPhaseDC<think1262> nests<think51><think1936>
```

- The text in red comes from our normal pretraining dataset. The text after it in blue is produced via autoregressive inference. Repeatedly predicting the token that would come after the current last token.
- Tokens in dark blue are normal text tokens. The cyan colored ones, are the 'thinking tokens'. Since they have no text association, they are simply represented with their token id.
- The model is early in training, so its results are garbage, but this is what an example might look like. Normal text tokens, interspersed with thinking tokens, whenever the model needs to externalize an intermediate result the context so it can refer back to it before it makes a prediction.
- in this case, supervision come from a pretrained teacher model, used to judge the rollout of the learning model. It only sees the non-thinking tokens.

- An alternative approach would involve not using the reference model, and being directly supervised by the next token. So doing s rollouts for a sequence of length s, each terminating with a single real token prediction. This is less parallelizable but potentially more principled.
- Another step on top of this could be to implement Coconut's alteration, and make all thinking tokens fully continuous.

What's the goal?

I see the main value of the project as being epistemic. For various reasons, it seems extremely unlikely that this will actually uncover a useful way to do language modelling. The main purpose is just figuring out what is going on with reasoning models and language models in general. I have a few intuitions about how language models/reasoning work, and based off these intuitions this approach might be effective, so I'm doing it to test those. Vaguely, a few of these intuitions would be:

- Transformers often implement shortcuts becuase implementing a full algorithmic solution is not cost effective (in terms of parameters) or impossible, due to algorithm runtime constraints.
  - (https://arxiv.org/abs/2210.10749)
  - letting the model output an arbitrary number of intermediate tokens means that any algorithm it can put in its weights, it can run to produce a prediction. (speed-space tradeoff)
- The reasoning that models are using (to great effect) is pretty shallow, and most tokens are there to imitate and arent useful for producing answers.
  - Supported by CoT faithfulness work which regularly finds that CoT is less faithful when the model can solve the problem without CoT at all. It just produces plausible-sounding reasoning.
    - (https://arxiv.org/pdf/2307.13702)
  - Models are not capable of learning to use their reasoning tokens to the fullest because they still really want to produce high probability text, even in their special tags which are trained differently.
    - There is, however, some evidence of some full utilization, counter to the goal high-probability text, going on. Things like reasoning models suddenly switching languages in their CoT.
    - I would strongly predict this to be causally useful for finding answers. but I suspect the reason its useful is stupid. maybe another research idea?
  - This is what 'reasoning tokens' are for. These are tokens which it never sees in pretraining, but it can output anyways. It can use them however it sees fit, and is never told to use them otherwise.
  - While the units of text are tokenized in a way that splits up information nicely, it seems likely to me that the best way to 'tokenize thoughts' is different. Some ideas really are simple but

require many words to explain properly. Some thoughts are very complicated but can be said in very few words.

- having separate thinking tokens lets the model assign meaning to them in whatever way is useful or most convenient.