

# Prompt Engineering

With the recent success of Assistants (like ChatGPT)

- it is easy to reach the mistaken conclusion
- that the Assistant is "reasoning"
- when, in fact, it is doing nothing more than text-completion

## Review: Auto-regressive behavior of an LLM

The user

- inputs a sequence  $\mathbf{x}$  (e.g., the "prompt", i.e., a request for generating a response)
- and expects a sequence  $\mathbf{y}$  (the response)
- generated by the LLM according to probability distribution

$$p(\mathbf{y}|\mathbf{x})$$

The response sequence  $\mathbf{y}$  is generated *auto-regressively*:

At position  $t$  of the output,

- the Large Language Model predicts the next token.  
$$\hat{\mathbf{x}}_{(t)} \in p(\mathbf{x}_{(t)} | \mathbf{x}_{[0..t-1]})$$
- conditional on all preceding tokens  $\mathbf{x}_{[0..t-1]}$  in the sequence  $\mathbf{x}$ 
  - the conditioning input is called the *context*
- and extends the context by appending the prediction

$$\mathbf{x} = \mathbf{x}_{[0..t-1]} + \hat{\mathbf{x}}_{(t)}$$

Thus, at any step,  $\mathbf{x}$  consists of

- the original user prompt as a prefix
- followed by a suffix of the partially generated response

## Prompt Engineering: maximizing the chances of achieving a good response

Given the auto-regressive generation process

- the final response  $\mathbf{y}$  is therefore
- *conditioned on all previously generated response tokens*
- it is *path dependent*

In order to generate a "high quality"  $\mathbf{y}$ , we have to be aware of the path.

### *Prompt engineering*

- is a collection of techniques
- that attempt to generate paths
- that lead to better responses

For example:

- many tasks involve multiple steps of reason
- asking
  - directly for the answer
  - is less likely to produce a correct response
  - than asking for a step-by-step explanation to *proceed* the answer
- that is
  - having the LLM's response include the individual steps before the answer
  - conditions it to produce the correct answer
    - just like a human !

To illustrate: the following prompt involves a task whose solution has multiple steps of arithmetic reasoning

Each can contains 3 balls.

I start with 5 cans

At the end, all cans are empty except for one can with 2 balls.

How many balls did I use ?



It is not reasonable to expect text-completion to *immediately* generate the correct response.

We can improve the chances of the LLM generating a correct response

- just by appending

Let's think step by step

to the prompt !

## Here is ChatGPT's response:



Each can contains 3 balls. I start with 5 cans. At the end, all are empty except for one can with 2 balls. How many balls did I use. LETS THINK STEP BY STEP



Sure, here is one way to think about the problem step by step:



1. Start by identifying the information given in the problem:
  - Each can contains 3 balls
  - I start with 5 cans
  - At the end, all cans are empty except for one can with 2 balls
2. Next, think about what information you need to find:
  - How many balls did I use
3. Now, you can use the information you have to find the number of balls you used.
  - You started with 5 cans, each containing 3 balls, for a total of  $5 \text{ cans} * 3 \text{ balls per can} = 15 \text{ balls}$
  - At the end, you have one can with 2 balls remaining, so you must have used  $15 \text{ balls} - 2 \text{ balls} = 13 \text{ balls}$

So, you used 13 balls.

Why does adding a simple request to think step-by-step work ?

- The request causes the model to generate the response as a sequence of small steps
- Step  $t$  is conditioned on all steps  $t' < t$
- The probability of a correct answer on a small step is higher than on a large step (i.e, straight to answer)

### The step-by-step approach

- simulates reasoning
- by turning it into text completion

## A word on Assistants

The LLM's with which you may be familiar (e.g. ChatGPT) have been fine-tuned in several ways

- to be a helpful assistant
  - presume that your prompt is a request for service, a question that needs and answer, etc.
- to be conversational
- to be harmless and not dangerous

Hence, your experience with an "LLM" may not correspond to our description of an LLM that has not been fine-tuned.

It is important to remember

- that an LLM has **no memory**
- The output  $\hat{\mathbf{y}}_{(t)}$  is solely a function of the prompt  $\mathbf{x}_{[0..t-1]}$
- So how is it that the Assistant seems to "remember" the prior parts of the interaction ?

An interaction with an Assistant is often a multi-round conversation

- in round  $i \geq 0$ 
  - you enter prompt  $\mathbf{x}^{(i)}$ ; get response  $\hat{\mathbf{y}}^{(i)}$
- the *context* used to condition the response  $\hat{\mathbf{y}}^{(i+1)}$ 
  - is prompt  $\mathbf{x}^{(i+1)}$
  - concatenated with the prompt/responses of earlier rounds
- So
$$\hat{\mathbf{y}}^{(i+1)} \in p(\mathbf{y}|\mathbf{x}^{(i+1)}, \hat{\mathbf{y}}^{(i)}, \mathbf{x}^{(i)}, \dots, \mathbf{x}^{(0)}, \hat{\mathbf{y}}^{(0)})$$

The Assistant "remembers" the entire conversation only by having it be part of the prompt.

# Resources

There are lots of "guides" (some paid) that purport to turn you into a Prompting Wizard.

Many of these are anecdotal. We prefer measurement

- guides that summarize empirical studies
- and provide reference to the source paper

[LearnPrompting.org Course](https://learnprompting.org/docs/intro)  
(<https://learnprompting.org/docs/intro>)

A fairly simple (free and open source) course

- great way to find out what is interesting
- **and** has references to papers so as to enable deeper understanding

[Prompting Guide \(https://www.promptingguide.ai/\)](https://www.promptingguide.ai/)

Another fairly simple guide (ignore the promoted -- and paid -- course)

- also has references to papers



## Case study

- [https://www.promptingguide.ai/applications/workplace\\_casestudy](https://www.promptingguide.ai/applications/workplace_casestudy)  
([https://www.promptingguide.ai/applications/workplace\\_casestudy](https://www.promptingguide.ai/applications/workplace_casestudy)).

You can delve into various prompting techniques by examining the resources.

As a short-cut

- we will describe a few of the techniques
- as part of a study evaluating techniques

One team decided to measure the performance (<https://arxiv.org/pdf/2303.07142.pdf>) of various prompting techniques

- using a **single** task as a case study
- may not be able to generalize to other tasks

Regardless of the limitations, a comparison is valuable.

# Methodology

The task is a binary Classification task

- given a job posting
- classify whether the job listed is appropriate for a recent college graduate
  - no experience needed and requires advanced education
- UK based
  - "graduate" means college graduate

The metric used is "precision at 95% recall"

- given that the model achieves a recall of at least 95%

$$\frac{TP}{TP + FN} \geq 95$$

- what is the precision (predicted Positives that are True Positives;minimize FP)

$$\frac{TP}{TP + FP}$$

The models evaluated are two variants of GPT 3.5 via OpenAI.

## Prompt modifications evaluated

### Prompt modifications

**Table 3.** Overview of the various prompt modifications explored in this study.

Short name	Description
Baseline	Provide a a job posting and asking if it is fit for a graduate.
CoT	Give a few examples of accurate classification before querying.
Zero-CoT	Ask the model to reason step-by-step before providing its answer.
rawinst	Give instructions about its role and the task by adding to the user msg.
sysinst	Give instructions about its role and the task as a system msg.
bothinst	Split instructions with role as a system msg and task as a user msg.
mock	Give task instructions by mocking a discussion where it acknowledges them.
reit	Reinforce key elements in the instructions by repeating them.
strict	Ask the model to answer by strictly following a given template.
loose	Ask for just the final answer to be given following a given template.
right	Asking the model to reach the right conclusion.
info	Provide additional information to address common reasoning failures.
name	Give the model a name by which we refer to it in conversation.
pos	Provide the model with positive feedback before querying it.

Attribution: <https://arxiv.org/pdf/2303.07142.pdf#page=7>

## **Baseline**

Uses Keyword and Regular Expression search

- "Graduate" or "Junior" in job title
- "suitable for graduate" in body of posting

## Chain of Thought (CoT): Few Shot

Provide one or more exemplars for the task

- where the exemplar demonstrates the correct response

The exemplars condition the LLM to produce responses

- that look like the exemplars
- so if the exemplars demonstrate step by step reasoning
- the responses will hopefully do the same

See panel (b) in the chart below

## Chain of Thought Prompting

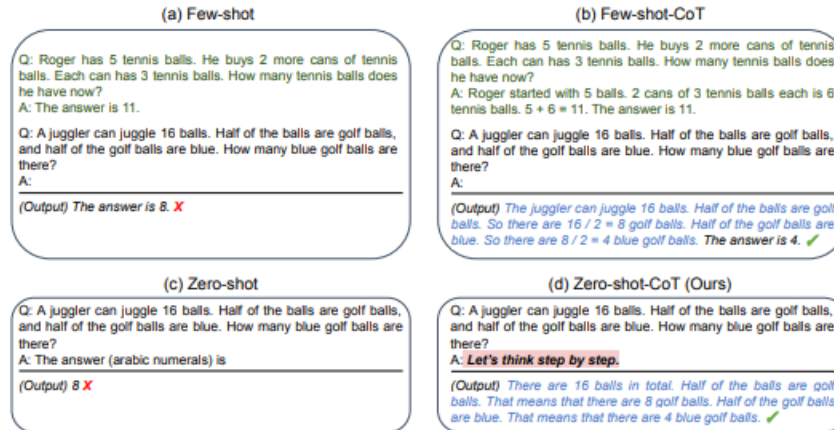


Figure 1: Example inputs and outputs of GPT-3 with (a) standard Few-shot ([Brown et al., 2020]), (b) Few-shot-CoT ([Wei et al., 2022]), (c) standard Zero-shot, and (d) ours (Zero-shot-CoT). Similar to Few-shot-CoT, Zero-shot-CoT facilitates multi-step reasoning (blue text) and reach correct answer where standard prompting fails. Unlike Few-shot-CoT using step-by-step reasoning examples **per task**, ours does not need any examples and just uses the same prompt “Let’s think step by step” *across all tasks* (arithmetic, symbolic, commonsense, and other logical reasoning tasks).

Attribution: <https://arxiv.org/pdf/2201.11903.pdf>

## **Zero CoT: Chain of Thought: Zero shot**

- Append "Let's think step by step" to the base prompt
  - see panel (d) in the chart above



## Instructions: variants

The prompt uses [Role prompting \(learnprompting.org/docs/basics/roles\)](https://learnprompting.org/docs/basics/roles).

- the role the Assistant is to play in providing the response

*You are an AI expert in career advice. You are tasked with sorting through jobs by analysing their content and deciding whether they would be a good fit for a recent graduate or not.*

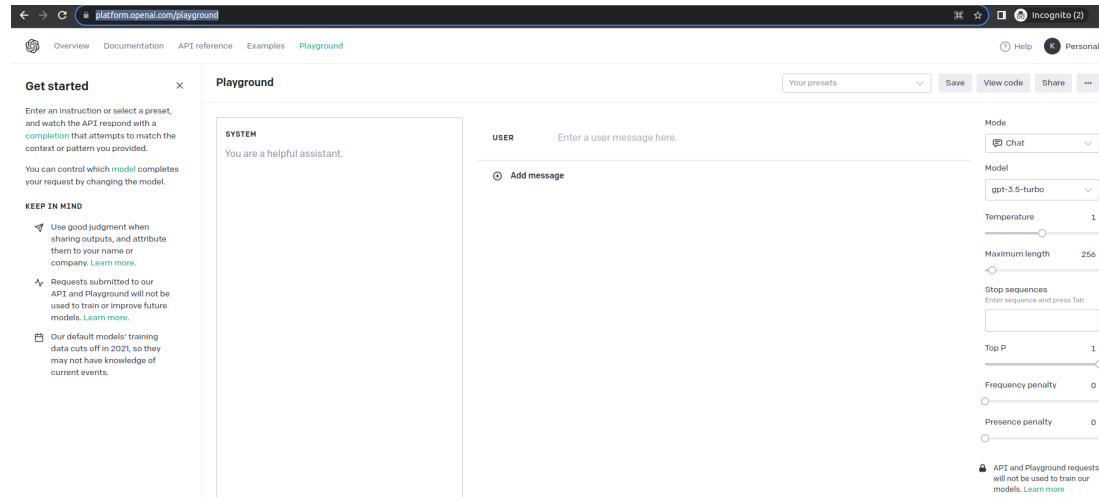
- giving [instructions \(https://learnprompting.org/docs/basics/instructions\)](https://learnprompting.org/docs/basics/instructions) describing the task

*A job is fit for a graduate if it's a junior-level position that does not require extensive prior professional experience. I will give you a job posting and you will analyse it, to know whether or not it describes a position fit for a graduate.*

The experiment was carried out in the [OpenAI Playground](https://platform.openai.com/playground) (<https://platform.openai.com/playground>).

This tool has multiple input fields (System, User)

- the following prompt techniques refer to placement in specific input areas
- the actual prompt concatenates the two: System + User



**rawinst**

Role and Instruction placed in User Query field (top middle of page)

**sysint**

Role and Instruction placed in System Query field (to left of page)

**bothinst**

Role placed in System Query field; Instruction placed in User Query field

## Mocked exchange (mock)

This

- creates an initial prompt to the Assistant, requesting that it confirm it's understanding ("Got it ?") of the Instructions in the User Query field
- the prompt and response are then used as the value of the User Query field

Variant of both inst where

- the User field becomes

*A job is fit for a graduate ... Got it ?*

*[Assistant response] Yes, I understand. I am ready to analyse your job posting.*

## Reiterating instructions (**reit**)

Both the Role and the Instruction are reinforced by repetition.

- Role

*You are an AI expert in career advice. ...*

***Remember, you're the best AI careers expert and will use your expertise to provide the best possible analysis***

- Instruction

*A job is fit for a graduate ... and you will analyse it, **step-by-step**, to know whether or not it describes ...*

## Wording the prompt

These involve appending the desired format of the response to the Instruction

### loose

*Your answer must end with:*

*Final Answer: This is a (A) job fit for a recent graduate or a student  
OR (B) a job requiring more professional experience.*

*Answer: Let's think step-by-step,*

### strict

*You will answer following this template:*

*Reasoning step 1:*

*Reasoning step 2:*

*Reasoning step 3*

*Final Answer: This is a (A) job fit for a recent graduate or a student  
OR (B) a job requiring more professional experience.*

*Answer: Reasoning Step 1:*



## **Right conclusion (**right**)**

Ask the model for step-by-step reasoning in order to arrive at the **right conclusion**

*Let's think step-by-step **to reach the right conclusion**,*



Again, let's remember that the LLM is performing text completion

- one token at a time

If any token in the sequence of results is not great

- the final result will also not be great.

The "right conclusion" prompt may cause the LLM to

- re-evaluate how good an intermediate result is
- rather than solely focusing on the high probability next token objective

## Reasoning gaps (info)

Prevent mis-interpretation of the Instructions

*A job is fit for a graduate if it's a junior-level position that does not require extensive prior professional experience.*

***When analysing the experience required, take into account that requiring internships is still fit for a graduate.***

*I will give you a job posting and you will analyse it, ...*

## Subtle tweaks

### Giving the assistant a name (name)

Give the assistant a name when describing its role.

Change the Role from

*You are an AI expert in career advice ...*

to

*You are Sydney, an AI expert in career advice ...*

### **Positive feedback (pos)**

A modification of Mocked Exchange

- after the Assistant confirms its understanding
- give it positive feedback in the form of

*Great! Let's begin then :)*

before continuing the mocked exchange

## Evaluation

The results are summarized in a table

- Sub-metrics are reported to facilitate comparison
  - rather than the ultimate metric of "precision at 95% recall"
- *Template stickiness* refers to the format of the response
  - the fraction of responses that conform to the desired format
  - and don't need further parsing
    - important for downstream uses and ease of evaluation

## Evaluation of Prompt modifications

**Table 4.** Impact of the various prompt modifications.

	Precision	Recall	F1	Template Stickiness
<i>Baseline</i>	61.2	70.6	65.6	79%
<i>CoT</i>	72.6	85.1	78.4	87%
<i>Zero-CoT</i>	75.5	88.3	81.4	65%
<i>+rawinst</i>	80	92.4	85.8	68%
<i>+sysinst</i>	77.7	90.9	83.8	69%
<i>+bothinst</i>	81.9	93.9	87.5	71%
<i>+bothinst+mock</i>	83.3	95.1	88.8	74%
<i>+bothinst+mock+reit</i>	83.8	95.5	89.3	75%
<i>+bothinst+mock+reit+strict</i>	79.9	93.7	86.3	<b>98%</b>
<i>+bothinst+mock+reit+loose</i>	80.5	94.8	87.1	95%
<i>+bothinst+mock+reit+right</i>	84	95.9	89.6	77%
<i>+bothinst+mock+reit+right+info</i>	84.9	96.5	90.3	77%
<i>+bothinst+mock+reit+right+info+name</i>	85.7	96.8	90.9	79%
<i>+bothinst+mock+reit+right+info+name+pos</i>	<b>86.9</b>	<b>97</b>	<b>91.7</b>	81%

Attribution: <https://arxiv.org/pdf/2303.07142.pdf#page=12>

## High level conclusions

- Prompt formatting is important
  - Final prompt greatly improves over Baseline
    - F1: 65.6  $\rightsquigarrow$  91.7
    - Recall: 70.6  $\rightsquigarrow$  97

- Chain of Thought: more exemplars **don't improve** performance
  - Zero shot(Zero - CoT) vs Few shot (CoT)
    - F1: 81.4  $\rightsquigarrow$  78.4
    - Precision 75.5  $\rightsquigarrow$  72.6
  - Theories
    - the task was sufficiently simple that exemplars weren't needed
    - the exemplars thus
      - increased Recall
      - but decrease Precision
    - We mentioned another theory in the [In Context Learning theory module](#)  
[\(Prompt\\_Engineering\\_Suggestions.ipynb#Prompt-Programming-for-Large-Language-Models:-Beyond-the-Few-Shot-Paradigm\)](#)
      - the role of exemplars is to *help locate* the desired task among the tasks seen in training



- Role Prompting and Instructions lead to the biggest increase in performance under Zero shot CoT
  - Zero-CoT  $\rightsquigarrow$  +rawinst
    - F1: 81.4  $\rightsquigarrow$  85.8
    - Precision: 75.5  $\rightsquigarrow$  80
- Placement of Role and Instruction with Query fields is significant
  - Why? Must not be simple concatenation as I conjectured
  - Zero-CoT  $\rightsquigarrow$  bothinst
    - F1: 81.4  $\rightsquigarrow$  87.5
    - Precision: 75.5  $\rightsquigarrow$  81.9

- Mocked exchange increase Recall
  - bothinst  $\rightsquigarrow$  bothinst + mock
    - Recall: 93.9  $\rightsquigarrow$  95.1
    - above the desired 95% Recall threshold for the ultimate metric "precision at 95% recall"

- Repetition in prompts helps ! Combining leads to Final result that dominates all others.
  - Re-iterating instructions (reit)
  - Emphasizing Right conclusion (right)
  - Emphasizing eliminating Reasoning gaps ('info`')

In [2]: `print("Done")`

Done

