

# دوره آموزش تولید محتوای متنی با استفاده از هوش مصنوعی مولد

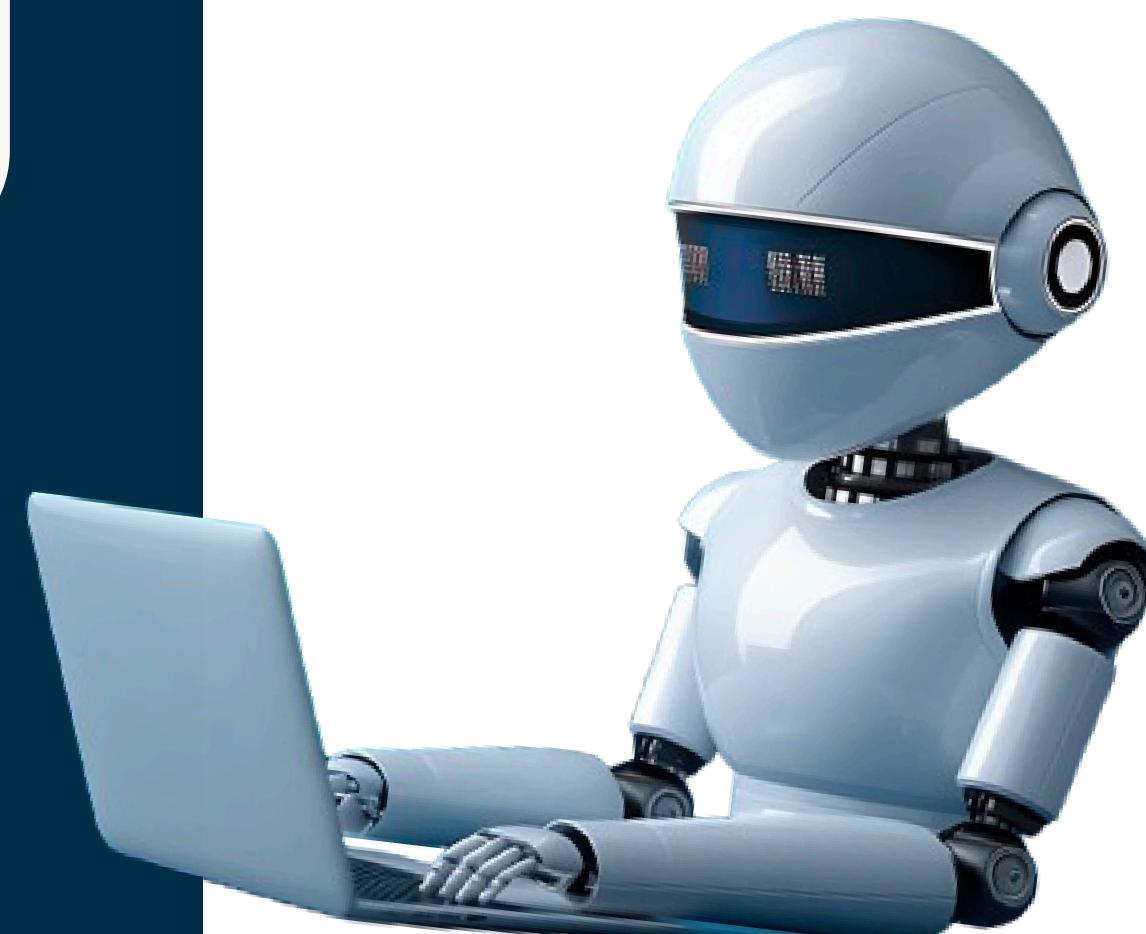
## فصل ششم

### مباحث پیشرفته ی مهندسی پرامپت

مرکز آموزش فناوری اطلاعات شهرستان گرگان  
مهندس مصطفی صادقی



سازمان آموزش فنی و حرفه ای کشور



# Iterative Refinement



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# چيست Iterative Refinement؟

- روشی برای بهبود خروجی مدل از طریق اصلاح تدریجی در همان مکالمه.
- هر پیام جدید، همراه با پیام‌های قبلی به صورت یک پرامپت واحد پردازش می‌شود.
- مدل علاوه بر دستور جدید، تاریخچه گفتگو + نمونه‌های تولیدشده را نیز در نظر می‌گیرد.
- این روش کیفیت، دقت و سازگاری خروجی را نسبت به بازنویسی پرامپت افزایش می‌دهد.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# چرا مکالمه بهتر از بازنویسی پرامپت است؟

- کل گفت‌وگو تبدیل می‌شود به Context ترکیبی شامل: دستورها، اصلاحات، و خروجی‌های مدل.
- مدل می‌تواند منظور شما را دقیق‌تر بفهمد زیرا مثال‌های واقعی از سبک موردنظر در مکالمه وجود دارد.
- دستوره‌های جدید در Context خروجی قبلی تفسیر می‌شوند و ابهام کاهش می‌یابد.
- خروجی نهایی به صورت طبیعی‌تر به سمت سبک، لحن و ساختار مطلوب هدایت می‌شود.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

چرا مکالمه بهتر از بازنویسی پرامپت است؟

Prompt = Conversation

# چرا مکالمه بهتر از بازنویسی پرامپت است؟

در روش Iterative Refinement مهم‌ترین نکته این است که:  
خروجی‌های مدل فقط پاسخ نیستند؛

بلکه خودشان به بخشی از «نمونه‌ها» و «الگوهای مرجع» تبدیل می‌شوند که مدل در مراحل بعدی از آن‌ها برای تفسیر دستورهای جدید استفاده می‌کند.

مدل، متن‌هایی را که خودش در پیام‌های قبلی نوشته، به عنوان مثال‌های معتبر در نظر می‌گیرد.  
این مثال‌ها همانند داده آموزشی کوچک عمل می‌کنند.  
دستور جدید شما در چارچوب همین نمونه‌ها درک می‌شود.  
هرچه نمونه‌ها دقیق‌تر شوند، مدل رفتار خود را با همان نمونه‌ها هم‌تراز می‌کند.  
پس خروجی مدل فقط پاسخ نیست،  
بلکه ورودی مرحله بعد و بخشی از پرامپت کل است.

# In Context Learning

# چيست In Context Learning؟

- In-Context Learning روشی برای یادگیری در مدل‌های زبانی بزرگ است.
- این یادگیری بدون تغییر وزن‌ها و پارامترهای مدل انجام می‌شود.
- تمام یادگیری از طریق محتوای پرامپت صورت می‌گیرد.

در این روش، مدل آموزش جدیدی نمی‌بیند و Fine-Tuning هم انجام نمی‌شود. مدل فقط بر اساس آنچه در متن ورودی می‌بیند، رفتار خود را تنظیم می‌کند.





Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# چیست In Context Learning؟

- مدل از روی ورودی‌ها الگو استخراج می‌کند.
- این الگوها فقط در همان مکالمه معتبرند.
- یادگیری موقتی و وابسته به Context است.

به محض تغییر پرامپت یا پایان مکالمه، این یادگیری از بین می‌رود.  
بنابراین In-Context Learning یادگیری پایدار نیست، بلکه یادگیری لحظه‌ای است.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# مدل در In-Context Learning چه چیزی یاد می‌گیرد؟

- ساختار پاسخ
- الگوی ورودی و خروجی
- سبک، لحن و قالب

مدل از روی مثال‌ها می‌فهمد «چه کاری باید انجام دهد» و «چگونه باید انجام دهد»، حتی اگر این موارد به صورت مستقیم گفته نشده باشند.

# چيست In Context Learning؟

وقتی از یک مدل زبانی می‌خواهیم متنی بنویسد که دقیقاً شبیه انتظار ما باشد، معمولاً با مشکل مواجه می‌شویم. هرچه دستور را دقیق‌تر و طولانی‌تر می‌کنیم، باز هم خروجی کاملاً مطابق خواسته‌ی ما نیست.

دلیل این موضوع این است که نوشتن صرفاً اجرای یک دستور نیست؛ نوشتن ترکیبی از سبک، لحن، احساس، انتخاب واژه و نحوه‌ی واکنش است که به‌سختی می‌توان همه‌ی آن را به دستورات عمل تبدیل کرد.

# چيست In Context Learning؟

در یادگیری انسانی، وقتی می‌خواهیم مثل یک فرد یا نویسنده‌ی خاص بنویسیم، ابتدا کتاب‌ها و نوشته‌های او را می‌خوانیم. با دیدن مثال‌ها، به مرور الگوهای زبانی، سبک بیان و طرز فکر او را درک می‌کنیم. این نوع یادگیری مبتنی بر مشاهده، بسیار مؤثرتر از خواندن یک لیست دستور است.

مدل‌های زبانی هم دقیقاً به همین شکل عمل می‌کنند.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# چیست In Context Learning؟

In-Context Learning یعنی مدل از روی محتوا و مثال هایی که داخل پرامپت می بیند یاد می گیرد که چه کاری باید انجام دهد. در این روش:

- ما خروجی مطلوب را نشان می دهیم.
- مدل از روی الگوها، نقش ها و ساختار متن، رفتار مناسب را استنتاج می کند.

هیچ آموزش جدیدی انجام نمی شود؛ یادگیری صرفاً از «Context پرامپت» اتفاق می افتد.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# چرا مثال‌ها قدرتمند هستند؟

مثال‌ها حجم زیادی از اطلاعات را به صورت فشرده منتقل می‌کنند. یک جمله‌ی نمونه می‌تواند هم‌زمان نشان دهد:

- چه واژه‌هایی استفاده می‌شود
- لحن رسمی است یا احساسی
- پاسخ کوتاه است یا مفصل
- واکنش مثبت است یا خنثی

اگر بخواهیم همین اطلاعات را با دستور بیان کنیم، باید چندین برابر متن بنویسیم. به همین دلیل، مثال‌ها نسبت به دستورها بسیار کاراتر هستند.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# ارتباط با اصلاح تدریجی (Iterative Refinement)

در مکالمه‌های چند مرحله‌ای، هر پاسخ قبلی مدل به عنوان یک مثال جدید عمل می‌کند. وقتی خروجی را اصلاح می‌کنیم، در واقع داریم نمونه‌های خوب و بد را به مدل نشان می‌دهیم. مدل از روی همین نمونه‌ها، پاسخ‌های بعدی را بهبود می‌دهد.

پس اصلاح تدریجی در عمل، شکلی از In-Context Learning است.





Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# استفاده ی ترکیبی از مثال ها و دستورها

مدل همیشه همه ی قوانین را فقط از روی مثال متوجه نمی شود. بعضی الگوها برای انسان بدیهی هستند، اما برای مدل نه. در این موارد، اگر مثال ها کم باشند، مدل ممکن است قانون را نادیده بگیرد یا ناقص اجرا کند.

بعضی چیزها را بهتر است صریح بگوییم. قوانینی که:

- دقیق هستند
- استثنا ندارند
- به صورت مکانیکی اجرا می شوند
- با یک دستور کوتاه خیلی شفاف تر منتقل می شوند تا با ده ها مثال.
- به همین دلیل، در پرامپت های قوی معمولاً:
- مثال ها برای انتقال سبک و حس استفاده می شوند
- دستورها برای مشخص کردن قوانین خاص

وقتی تعداد مثال ها را افزایش می دهیم، مدل الگو را بهتر تشخیص می دهد. در این حالت حتی اگر دستور صریح حذف شود، مدل همچنان همان رفتار را ادامه می دهد. به این رویکرد Few-shot Learning می گوییم؛ یعنی یادگیری از چند نمونه ی محدود داخل پرامپت.



# Writing Persona



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# Writing Persona Pattern چیست؟

Writing Persona Pattern یعنی ساختن یک «پرامپت قابل استفاده مجدد» که به مدل آموزش می‌دهد دقیقاً با سبک، لحن، قالب و ساختار شما بنویسد. ایده اصلی این است: به جای اینکه فقط دستور بدهید، نمونه‌ی واقعی از نوشته‌های خودتان را هم داخل پرامپت می‌گذارید تا مدل از آن‌ها «سبک» را یاد بگیرد.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# Writing Persona Pattern چیست؟

وقتی می‌گویید: «یک پاراگراف درباره‌ی موضوع X بنویس»، مدل:  
لحن مورد نظر شما را ندارد

سبک و قالب نوشتن شما را رعایت نمی‌کند

ایده‌ها را به مسیر اشتباه می‌برد (حتی ممکن است موضوع را بد بفهمد)

دلیلش این است که فقط به مدل گفته ایم «چه بنویسد»، نه «چطور مثل من بنویس».



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# فرمت پایه‌ی Writing Persona Pattern

دستورالعمل‌ها درباره اینکه چه چیزی / چطور نوشته شود...

موارد زیر نمونه‌هایی از سبک نوشتاری موردنظر برای استفاده هستند:

نمونه‌های نوشتاری اینجا قرار می‌گیرند...



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# Separator (جداکننده) چرا ضروری است؟

بین دستورها و نمونه‌ها باید جداکننده بگذارید (مثل --- یا =====) چون:  
ممکن است خودِ نمونه‌ها داخلشان جمله‌های دستوری داشته باشند  
مدل باید بفهمد این قسمت «نمونه‌ی سبک» است، نه «دستور اجرای کار»  
این کار احتمال برداشت اشتباه را کم می‌کند.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# چرا از این الگو استفاده کنیم؟

با اضافه کردن نمونه‌ها، خروجی معمولاً:  
از حالت عمومی و ضعیف → به یک پیش‌نویس نزدیک به سبک شما تبدیل می‌شود  
حتی اگر همه‌ی جزئیات ساختار را نگوئید، مدل از نمونه‌ها الگو می‌گیرد  
پرامپت بزرگ‌تر می‌شود، اما کنترل شما بیشتر می‌شود  
نکته مهم: خروجی را به عنوان نسخه «نهایی» در نظر نگیرید؛ آن را یک پیش‌نویس بدانید که شما ویرایشش می‌کنید.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# آیا این الگو جایگزین نویسنده می شود؟

هدف این نیست که AI به جای شما فکر کند.  
هدف این است که یک پیش نویس نزدیک به سبک نوشتن شما بنویسد  
بعد شما:

ایده های جافتاده را اضافه می کنید  
جمله ها را اصلاح می کنید  
و مسئولیت متن نهایی را می پذیرید  
AI ابزار کمکی برای نوشتن است، نه نویسنده ی مستقل.

# محدودیت مهم Writing Persona

نمونه‌هایی که انتخاب می‌کنید، پرسونا را به یک «فضای مسئله» محدود می‌کند.  
اگر نمونه‌ها از متن آکادمیک باشند، شما دارید به مدل یاد می‌دهید:  
مثل من برای مقاله بنویس  
نه اینکه:

مثل من برای ایمیل، شبکه اجتماعی، پیام شخصی، یا متن تبلیغاتی بنویس  
پس اگر همان پرسونا را برای ایمیل به همکار یا پیام کوتاه استفاده کنید:

- لحن اشتباه می‌شود
- قالب اشتباه می‌شود
- محتوا نامتناسب می‌شود





Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# محدودیت مهم Writing Persona

برای هر نوع خروجی، مثال همان نوع را بدهید:

- برای مقاله → نمونه‌ی مقاله‌های خودتان

- برای ایمیل → نمونه‌ی ایمیل‌های خودتان

- برای شبکه اجتماعی → نمونه‌ی پست‌های خودتان

- برای پیام رسمی سازمانی → نمونه‌ی پیام‌های رسمی خودتان

اگر مثال‌ها درست انتخاب نشوند، مدل کانتکست کافی برای In-Context Learning ندارد و خروجی نامطلوب می‌شود.

# Preference-Driven Refinement of Prompts

# Preference-Driven Refinement of Prompts

یک روش مرحله‌ای برای ساختن پرامپت‌های قابل استفاده مجدد (Reusable) به جای اینکه فقط «دستورها» را عوض کنیم، «ترجیحات» خود را به شکل مثال از خروجی که مدل به ما داده است، وارد پرامپت می‌کنیم. مدل با دیدن نمونه‌های خوب/بد، سریع‌تر به خروجی دلخواه نزدیک می‌شود.

چرا مهم است؟

- نمونه متن‌های آماده با لحن دلخواه نداریم.
- نوشتن مثال از ابتدا وقت گیر است.

# گام‌های اصلی این فرآیند چیست؟

این الگو از خود LLM کمک می‌گیرد تا نمونه‌هایی (Examples) برای پرامپت شما تولید کند، و شما با انتخاب و برچسب گذاری این نمونه‌ها، ترجیحات خود را به مدل تزریق می‌کنید.

• شروع با پرامپت خام (Naive Prompt):

1. یک دستور ساده بنویسید (مثلاً: "یک ایمیل به همکارانم درباره دوره جدید پرامپت‌نویسی بنویس").

2. خروجی را از مدل بگیرید.

• شناسایی ترجیحات:

3. خروجی تولید شده توسط مدل را بررسی کنید.

4. بخش‌هایی را که از نظرتان مطلوب است کپی کنید (مثال‌هایی از آنچه مدل باید انجام دهد).

5. بخش‌هایی را که بنظرتان نامطلوب است کپی کنید (مثال‌هایی از آنچه مدل نباید انجام دهد).

• تزریق بازخورد و اصلاح پرامپت:

6. نمونه‌های "مطلوب" و "نامطلوب" را به پرامپت اصلی برگردانید و آن‌ها را برچسب گذاری کنید (مثلاً: Examples of what I like: و Examples of what I don't like:).

7. دستورالعمل‌های اضافی را برای راهنمایی بیشتر اضافه کنید (مثلاً: "بر اهمیت پرامپت‌نویسی برای دانشجویان فنی و حرفه‌ای تأکید کن").

8. پرامپت جدید و اصلاح شده را دوباره اجرا کنید.

9. خروجی‌های جدید را مجدداً بررسی کنید، بخش‌های مورد علاقه و غیرعلاقه را استخراج کرده و به پرامپت برگردانید.

10. یک یا دو دستورالعمل جدید دیگر اضافه کنید (مثلاً: "از زبان ساده و مستقیم و بدون صفت‌های زیاد استفاده کن").

# چرا این الگو مؤثر است؟

- استفاده از یادگیری درون متنی (In-Context Learning): به جای حدس زدن و نوشتن دستی همه دستورالعمل‌ها، شما به مدل مثال‌هایی از عملکرد خوب و بد خودش می‌دهید. این مثال‌ها قدرت مدل را در درک سبک، لحن و ساختار مورد نظر شما به شدت افزایش می‌دهند.
- افزایش تدریجی کیفیت: پرامپت به جای اینکه در یک مرحله نوشته شود، در طی چند گام تکراری، با غنی شدن دستورالعمل‌ها و مثال‌ها قوی‌تر و قوی‌تر می‌شود.
- صرفه‌جویی در زمان: به جای نوشتن مثال‌ها از صفر، از خروجی خود مدل برای ایجاد مثال‌ها استفاده می‌کنید.

# Prompting For Options



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# هوش مصنوعی ابزار تقویت تفکر، نه جایگزین آن

هوش مصنوعی مولد زمانی بیشترین ارزش را ایجاد می‌کند که به‌عنوان ابزاری برای گسترش تفکر استفاده شود، نه ماشینی برای حذف آن. استفاده‌ای که صرفاً با هدف سرعت انجام شود، به‌مرور باعث کاهش دقت، خلاقیت و قضاوت می‌شود. در مقابل، زمانی که این ابزار برای دیدن گزینه‌های بیشتر، بررسی راه‌حل‌های متنوع و تعمیق فهم مسئله به کار گرفته شود، کیفیت خروجی به‌طور چشمگیری افزایش پیدا می‌کند. محور این فصل دقیقاً روی همین نوع استفاده متمرکز است.

مواردی که این رویکرد را از استفاده‌ی سطحی جدا می‌کند:

- تمرکز بر کیفیت تصمیم به جای سرعت انجام
- فعال نگه داشتن نقش قضاوت انسانی
- استفاده از AI برای گسترش فضای فکر، نه بستن آن





Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# نگاه به خروجی مدل‌های زبانی به‌عنوان یک دیدگاه

خروجی مدل زبانی نباید به‌عنوان پاسخ نهایی تلقی شود. هر خروجی تنها یکی از بی‌نهایت دیدگاه‌های ممکن نسبت به مسئله است. مدل‌ها بر اساس احتمال و الگو تولید می‌کنند و نه بر اساس حقیقت مطلق یا زمینه‌ی دقیق مسئله. این موضوع به‌ویژه زمانی اهمیت پیدا می‌کند که تصمیم‌ها مهم یا اثرگذار هستند.

ویژگی‌های خروجی به‌عنوان دیدگاه:

- ممکن است ناقص باشد.
- ممکن است جانبدارانه باشد.
- ممکن است تنها بخشی از فضای مسئله را پوشش دهد.

رفتار صحیح در مواجهه با خروجی:

- تولید چند دیدگاه
- مقایسه‌ی آن‌ها
- انتخاب یا ترکیب آگاهانه

مثال:

- اجرای یک پرامپت یکسان در دو گفتگوی مستقل
- مشاهده‌ی تفاوت لحن، ساختار و پیشنهادها
- استفاده از تفاوت‌ها برای بهبود نتیجه‌ی نهایی





Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# بهره‌برداری آگاهانه از قابلیت تولید سریع

مدل‌های زبانی می‌توانند در زمان بسیار کوتاه، تعداد زیادی خروجی تولید کنند. این ویژگی زمانی ارزشمند می‌شود که به صورت آگاهانه از آن استفاده شود. به جای اینکه تولید سریع باعث پایان زودهنگام فکر شود، باید به نقطه‌ی شروع فکر تبدیل شود.

تولید یک خروجی = از دست دادن فرصت  
تولید چند خروجی = ایجاد فضای انتخاب

- مزایای تولید چندرویکرد مختلف:
- افزایش احتمال دیدن راه حل بهتر
  - کاهش وابستگی به اولین پاسخ
  - ایجاد امکان مقایسه و تحلیل
  -

نمونه‌ی پرامپت ضعیف:

«یک ایمیل برای معرفی محصول X بنویس»

نمونه‌ی پرامپت قوی:

«سه ایمیل با سه رویکرد متفاوت برای معرفی محصول X بنویس و تفاوت آن‌ها را توضیح بده»



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# فعال شدن تفکر انتقادی با وجود رویکردهای مختلف

تفکر انتقادی زمانی فعال می‌شود که انتخاب واقعی وجود داشته باشد. وجود تنها یک گزینه، ذهن را به پذیرش یا رد محدود می‌کند. اما چند گزینه ذهن را وارد فرآیند تحلیل می‌کند: چرا این بهتر است؟ چه چیزی در این یکی ضعیف‌تر است؟ کدام با هدف هم‌راست‌تر است؟

مزایای وجود گزینه‌های متعدد:

- فعال شدن قوه‌ی مقایسه
- شکل‌گیری سلیقه و سبک شخصی
- امکان ترکیب بهترین بخش‌ها

مثال:

- از یک متن تیتراژ انتخاب می‌شود
- از متن دیگر لحن حفظ می‌شود
- از متن سوم ساختار یا دعوت به اقدام برداشته می‌شود

نتیجه:

خروجی نهایی بهتر از هر نسخه‌ی منفرد



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

بزرگ‌ترین بازدهی هوش مصنوعی  
مولد، کمک به تصمیم‌گیری بهتر از  
طریق دیدن رویکردها و گزینه‌های  
بیشتر است، نه انجام سریع‌تر کارها.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# الگوی Ask for Options Pattern

این الگو بر یک اصل ساده اما قدرتمند استوار است: هرگز فقط یک راه حل نخواهید، همیشه چندین گزینه درخواست کنید.

## مرحله ۱: درخواست اولیه

- مسئله را شفاف شرح دهید
- تعداد گزینه‌های مورد نظر را مشخص کنید (معمولاً ۳ تا ۵)
- درخواست مقایسه واضح داشته باشید

## مرحله ۲: تحلیل خروجی

- تمام گزینه‌ها را مرور کنید - حتی آن‌هایی که به نظر غیرعملی می‌رسند
- مزایای هر گزینه را بررسی کنید - چه نکات جدیدی یاد گرفتید؟
- معایب و ریسک‌ها را ببینید - هوش مصنوعی چه خطراتی را شناسایی کرده که شما ندیده‌اید؟

## مرحله ۳: تصمیم‌گیری بهبودیافته

- حالا شما چندین گزینه ساختاریافته دارید
- تحلیل مقایسه‌ای آماده است
- می‌توانید ترکیبی از چند گزینه ایجاد کنید
- یا گزینه کاملاً جدیدی ابداع کنید که نقاط قوت چند گزینه را دارد



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# روش‌های تولید گزینه‌ها و رویکردهای مختلف

## ۱. بازتولید (Regenerate) با تغییرات جزئی

**توضیح:** پس از دریافت اولین پاسخ، پرامپت اصلی را با یک تغییر دوباره ارسال کنید یا از دکمه Regenerate استفاده کنید.

**چگونه:** در همان گفتگو، پرامپت را کمی تغییر دهید و دوباره اجرا کنید.

**مزیت:** سریع و ساده است.

**نکته:** تغییرات ممکن است قابل توجه نباشد و Context مکالمات قبلی رو پاسخ جدید تاثیر بگذارد.

## ۲. گفتگوهای مستقل (New Conversations)

**توضیح:** پرامپت‌های مشابه را در چند گفتگوی جداگانه کپی و اجرا کنید.

**چگونه:** یک تب جدید در ChatGPT باز کنید، پرامپت را paste کرده و اجرا کنید. این کار را چند بار تکرار کنید.

**مزیت:** هر پاسخ کاملاً مستقل است، زیرا مدل تاریخچه پاسخ‌های قبلی را نمی‌بیند. این می‌تواند منجر به تنوع بیشتر و غیرقابل پیش‌بینی‌تری شود.

**نکته:** راه حل خوبی برای شکستن یک خط فکری تکراری در مدل.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# روش‌های تولید گزینه‌ها و رویکردهای مختلف

۳. درخواست مستقیم چند گزینه در یک پرامپت  
توضیح: به طور صریح از مدل بخواهید چندین گزینه را در یک پاسخ ایجاد کند.  
چگونه: از پرامپت‌هایی مانند این استفاده کنید: "سه نسخه مختلف از ایمیل بنویس...". یا "پنج ایده برای این مشکل ارائه کن."  
مزیت: همه گزینه‌ها را به سرعت و در یکجا دریافت می‌کنید.  
نکته: از آنجا که مدل همه گزینه‌ها را با هم می‌سازد، ممکن است تنوع عمیقی نداشته باشند و همه از یک "حالت" یا "خط فکری" مشابه پیروی کنند.

۴. استفاده از مدل‌ها یا ابزارهای مختلف  
توضیح: همان درخواست را در مدل‌های زبانی مختلف امتحان کنید.  
چگونه: پرامپت خود را در Claude، ChatGPT، (آنتروپیک)، Gemini (گوگل)، Copilot (مایکروسافت) و دیگر مدل‌ها اجرا کنید.  
مزیت: هر مدل شخصیت، نقاط قوت، آموزش‌های اولیه و جهت‌گیری‌های متفاوتی دارد که منجر به دیدگاه‌های منحصربه‌فرد می‌شود.  
نکته: این روش به‌ویژه برای مسائل پیچیده، خلاقانه یا چندوجهی بسیار ارزشمند است، زیرا هر مدل ممکن است بر جنبه متفاوتی از مسئله تمرکز کند یا سبک نوشتاری کاملاً متفاوتی ارائه دهد.



# تعیین "سطح" و "منبع" تفاوت: راهبردی برای تنوع کیفی بین گزینه ها

این روش به معنای تعیین آگاهانه معیار یا بعدی است که شما می خواهید گزینه های تولید شده در آن بعد با یکدیگر تفاوت داشته باشند. به جای گفتن "متفاوت باش"، شما دقیقاً مشخص می کنید که در چه زمینه ای و تا چه سطحی متفاوت باشد.

چرا مهم است؟

- **جلوگیری از تنوع سطحی:** بدون این راهنما، مدل ممکن است فقط کلمات را عوض کند اما تمام گزینه ها در ساختار، لحن و استراتژی یکسان باشند (مثلاً سه ایمیل که همگی رسمی هستند).
- **پوشش طیف وسیع تر:** شما را مطمئن می سازد که گزینه ها واقعاً مکمل یکدیگر بوده و زوایای مختلفی از مسئله را پوشش می دهند.
- **ارتقای تفکر انتقادی:** شما را وادار می کند قبل از تولید، درباره مسئله و ابعاد آن فکر کنید.

# تعیین "سطح" و "منبع" تفاوت: راهبردی برای تنوع کیفی بین گزینه ها

۱. سطح سبک و لحن (Style & Tone)

منبع تغییر: احساس، رسمیت و شخصیت متن.

مثال پرامپت:

"برای اعلام این دوره، سه ایمیل بنویس:

۱. یک نسخه کاملاً رسمی و حرفه‌ای برای مدیران.

۲. یک نسخه دوستانه و مشتاقانه برای همکاران.

۳. یک نسخه هیجان‌انگیز و فوری که حس کوتاه بودن زمان را القا کند."

۲. سطح ساختار و قالب (Structure & Format)

منبع تغییر: سازماندهی اطلاعات و شکل ارائه.

مثال پرامپت:

"این ایده را در سه قالب مختلف ارائه کن:

۱. یک لیست بولت‌پوینت مختصر و مستقیم.

۲. یک پاراگراف روایی و داستان‌وار.

۳. یک Q&A (پرسش و پاسخ) برای رفع ابهامات رایج."





Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# تعیین "سطح" و "منبع" تفاوت: راهدردی برای تنوع کیفی بین گزینه ها

۳. سطح استراتژی یا رویکرد (Strategy or Approach)  
منبع تغییر: منطق پنهان پشت پیام یا چارچوب فکری.  
مثال پرامپت:

"برای بازاریابی این محصول، از سه چارچوب معروف استفاده کن و برای هر کدام یک پیام بنویس:  
۱. چارچوب AIDA (توجه، علاقه، تمایل، عمل).  
۲. چارچوب PAS (مشکل، تشدید، راه حل).  
۳. چارچوب Before-After-Bridge (قبل، بعد، پل)."

۴. سطح مخاطب یا دیدگاه (Audience or Perspective)  
منبع تغییر: فرد یا گروهی که پیام از نگاه او نوشته می شود.  
مثال پرامپت:

"این خبر را از سه دیدگاه کاملاً متفاوت بنویس:  
۱. از دید یک کارآفرین خوشبین که به دنبال فرصت است.  
۲. از دید یک مدیر محتاط که نگران ریسک است.  
۳. از دید یک مشتری کنجکاو که به دنبال حل مشکل شخصی خود است."



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# تعیین "سطح" و "منبع" تفاوت: راهبردی برای تنوع کیفی بین گزینه ها

۵. سطح محدودیت خلاق (Creative Constraint)  
منبع تغییر: اعمال یک قانون یا محدودیت خاص به عنوان محرک خلاقیت.  
مثال پرامپت:

- "این توضیح را در سه حالت بنویس:
۱. فقط با جملات حداکثر ۱۰ کلمه‌ای.
  ۲. به گونه‌ای که یک کودک ۱۰ ساله بفهمد.
  ۳. به صورت یک قافیه یا شعر کوتاه."



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# ارزیابی گزینه های ارائه شده توسط GenAI

پس از تولید گزینه های مختلف با استفاده از هوش مصنوعی مولید، باید این گزینه ها را ارزیابی کنیم. برای ارزیابی بی طرفانه، باید این دو مرحله را در گفتگوهای کاملاً جداگانه با هوش مصنوعی انجام دهیم. اگر در یک گفتگو هم ایده بسازیم هم معیار سنجش آن را، مدل دچار سوگیری تایید می شود و معیارهایی را پیشنهاد می دهد که ایده های از قبل تولید شده را تایید کنند.

مراحل ارزیابی:

۱. ساخت معیارهای ارزیابی (در یک گفتگوی مستقل)

کاری که می کنید: در یک چت جدید، از مدل می خواهید معیارهایی برای سنجش مسئله شما تولید کند، بدون آنکه راه حلی ارائه دهد. مثال پرامپت:

"برای این کار، پنج معیار برای مقایسه و ارزیابی راه حل های مختلف پیشنهاد کن. خود راه حل ها را لیست نکن. کار: [در اینجا توضیح مسئله را می گذارید]"

خروجی: لیستی از معیارهای کیفی (مثلاً: سهولت اجرا، زمان مورد نیاز، ثبات نتایج، نیاز به نظارت، ریسک شکست).

۲. تولید ایده ها (در گفتگوهای دیگر)

کاری که می کنید: در یک یا چند گفتگوی جداگانه، ایده ها، راه حل ها یا گزینه های ممکن را برای مسئله خود تولید می کنید.

۳. ارزیابی نهایی با تلفیق دو خروجی

کاری که می کنید: تمام ایده های تولید شده (از مرحله ۲) و معیارهای ارزیابی (از مرحله ۱) را در یک گفتگوی جدید کنار هم قرار می دهید. مثال پرامپت:

"این [ایده ها/راه حل ها] را با استفاده از معیارهای زیر ارزیابی کن: [لیست معیارها]"

۴. درخواست فرمت: می توانید از مدل بخواهید نتیجه را در قالب جدول، ماتریس ارزیابی یا چارت مقایسه ای ارائه دهد.

# Controlling Output Formatting



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# مرور Template Pattern

- Template Pattern یک روش حرفه‌ای در پرامپت نویسی است که به شما امکان می‌دهد:
- یک قالب کاملاً مشخص تعریف کنید
  - بخش‌های قابل تغییر را به صورت جای خالی (Placeholder) تعیین کنید
  - مدل را موظف کنید فقط همان قالب را حفظ کند
  - و محتوا را دقیقاً در جای خالی‌ها قرار دهد
  - این دقیقاً همان کاری است که با یک دستیار حرفه‌ای انجام می‌دهید:
- فرم می‌دهید، جای خالی تعریف می‌کنید، و انتظار دارید فرم دقیقاً همین گونه کامل شود.

PlaceHolders = جای خالی

# مرور Template Pattern

برای اینکه مدل به صورت کامل از قالب تبعیت کند، چهار اصل وجود دارد:

1. اعلام وجود قالب

به مدل می‌گوییم:  
من یک قالب به تو می‌دهم.  
این جمله مسیر ذهنی مدل را روشن می‌کند.

2. تعریف جای خالی‌ها

کلماتی که با حروف بزرگ نوشته می‌شوند نقش Placeholder دارند:

PRODUCT\_NAME

PRICE

CTA\_TEXT

مدل می‌فهمد این بخش‌ها باید با خروجی واقعی جایگزین شوند.

3. دستور پر کردن بخش‌ها

با یک جمله شفاف:

جای خالی‌ها را با محتوای مناسب پر کن.

4. اجبار به حفظ قالب

برای جلوگیری از تغییر ناخواسته ساختار:

قالب را بدون هیچ گونه تغییر حفظ کن.

این مرحله قلب Template Pattern است.





Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# Markdown چیست؟

Markdown یک زبان متنی ساده برای قالب‌بندی متن است.  
در Word شما:

متن را انتخاب می‌کنید

روی دکمه Bold، Heading، Bullet و... کلیک می‌کنید

اما در مدل‌های زبانی:

GUI وجود ندارد

• فقط متن رد و بدل می‌شود

پس باید قالب‌بندی را در خود متن مشخص کنیم؛

و Markdown دقیقاً برای همین ساخته شده است.



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# چرا Markdown برای LLM ها مهم است؟

مدل های زبانی کاملاً Markdown را می فهمند.  
مدل هایی مثل ChatGPT و Claude هم Markdown را رندر می کنند.  
بنابراین:

- مدل متن Markdown تولید می کند.
- ابزار آن را به خروجی فرمت شده تبدیل می کند.
- بدون نیاز به ویرایش دستی.





Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# Footnotes (پاورقی ها)

Footnote ها:

- به مدل اجازه می دهند منبع یا دلیل پاسخ را مشخص کند.
- می توانید از مدل بخواهید:
- بگویند کدام بخش از ورودی را برای پاسخ استفاده کرده ، برای:
- کاهش Hallucination
- قانع کردن افراد بدبین به LLM
- خروجی قابل اعتماد



Iterative Refinement



In Context Learning



Writing Persona



PDR



Prompting for options



Output Formatting

# Escape Valves

وقتی از مدل زبانی می‌خواهید کاری را طبق قالبی خاص انجام دهد (مثلاً استخراج اطلاعات یا پر کردن جاهای خالی)، ممکن است بعضی داده‌ها با فرمت موردنظر شما سازگار نباشند. مثلاً از مدل می‌خواهید "اعداد" را از یک متن استخراج کند، اما برخی اطلاعات متن غیر عددی هستند (مثل نام مکان‌ها).

اگر به مدل نگوئید در چنین مواردی چه کار کند، مدل سعی می‌کند به زور آن اطلاعات را در قالب بگنجاند. (مثل قرار دادن یک میخ مربعی در سوراخ گرد!). این باعث اشتباه در خروجی می‌شود.

راه حل: تعیین Escape Valves

باید صراحتاً به مدل بگوئید اگر داده‌ای با قالب موردنظر مطابقت نداشت، چه کند.

- اگر داده عددی نیست، به جای آن "N/A" (غیرقابل اعمال) قرار بده.
- اگر اطلاعات کافی نیست، "اطلاعاتی موجود نیست" بنویس.
- اگر نمی‌توانی کاری را انجام دهی، "مشکل را توضیح بده" و ادامه نده.