به نام خدا

دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر

مدل‌های مولد عمیق

تمرین شماره سه

نام و نام خانوادگی	فاطمه نادی
شماره دانشجویی	۸۱۰۱۰۱۲۸۵
تاریخ ارسال گزارش	۱۴۰۲/۱۰/۳۰

فهرست گزارش سوالات

فهرست شکل‌ها

فهرست جداول

پرسش ۱ LLM

گام صفر I) Preprocessing and Load Dataset

دیتاست به کار رفته در این سوال TweetSumm است که ساختار کلی آن به فرم زیر است:

```
DatasetDict({
    train: Dataset({
        features: ['original dialog id', 'new dialog id', 'dialog index', 'original dialog info', 'log', 'prompt'],
        num_rows: 879
    })
    validation: Dataset({
        features: ['original dialog id', 'new dialog id', 'dialog index', 'original dialog info', 'log', 'prompt'],
        num_rows: 110
    })
    test: Dataset({
        features: ['original dialog id', 'new dialog id', 'dialog index', 'original dialog info', 'log', 'prompt'],
        num_rows: 110
    })
})
```

شکل 1 اجزا و ویژگی‌های تشکیل دهنده دیتاست **tweetsumm**

همان‌طور که مشاهده می‌شود در داده‌های آموزش ۸۷۹ مکالمه به همراه خلاصه‌ی آن آورده شده است.
به منظور انجام مراحل بعدی سوال نیاز است در ابتدا پیش‌پردازشی روی داده‌های ورودی اعمال شود تا بتوان به طور موثر در وظایف بعدی از آن استفاده نمود.
همچنین این دیتاست شامل ۱۱۰ داده ارزیابی و ۱۱۰ داده تست است که تمامی پیش‌پردازش‌ها روی هر سه این‌ها که در ادامه توضیح داده خواهد شد اعمال خواهد شد.
تابع `json.loads()` در پایتون برای تجزیه و تحلیل یک رشته JSON به یک شیء پایتونی (مثل دیکشنری یا لیست) استفاده می‌شود. این تابع رشته JSON را به دیکشنری‌ها و لیست‌ها تبدیل می‌کند و امکان استفاده آسان از داده‌های JSON در برنامه‌های پایتونی را فراهم می‌کند. در فرآیند پیش‌پردازش از این تابع استفاده کرده تا خلاصه مکالمات را از طریق این آدرس استخراج می‌کنیم:

```
json.loads(Dataset[index]["original dialog info"])["summaries"]["abstractive_summaries"]
```

همچنین برای استخراج مکالمات از بخش `Dataset[index]["log"]` ابتدا مکالمات user utterance و system response را جدا کرده داده را متن هر بخش از مکالمه را تمیزکرده و در نهایت با اضافه کردن نقش کدام به شکل :user ویا :agent سعی می‌کنیم بخش‌های مختلف مکالمه را تفکیک کرده تا برای مدل فهم آن آسان‌تر باشد.
در نهایت همین بخش‌های استخراج و پیش‌پردازش شده یعنی خلاصه و مکالمات را در دیتاست نگه‌داشته مابقی اجزا را حذف می‌کنیم.
به منظور تمیز کردن متن مکالمات نیز پیش‌پردازش‌های ساده همچون حذف لینک‌ها، حذف فاصله‌های اضافی، حذف نام‌کاربری افراد که با @ شروع شده باشند و همچنین الگوهای ثابت اصلاح شده است.

یک نمونه از متن موجود در داده آموزش را در ادامه مشاهده خواهید کرد، مواردی که با ### و یا # آغاز شده به صورتی دستی اضافه شده تا به فهم بهتر متن کمک کند و جز دیتاست نیست:

```
### Instruction: # Below is a dialogue between a human and an AI agent. Succinctly summarize the following customer-agent dialogue exchange in two sentences. Concisely encapsulate the core meaning from the customer and agent in separate sentences.

### Input:
user: So neither my iPhone nor my Apple Watch are recording my stepsactivity, and Health doesnt recognise either source anymore for some reason. Any ideas? please read the above.
agent: Lets investigate this together. To start, can you tell us the software versions your iPhone and Apple Watch are running currently?
user: My iPhone is on 11.1.2, and my watch is on 4.1.
agent: Thank you. Have you tried restarting both devices since this started happening?
user: Ive restarted both, also unpaired then repaired the watch.
agent: Got it. When did you first notice that the two devices were not talking to each other. Do the two devices communicate through other apps such as Messages?
user: Yes, everything seems fine, its just Health and activity.
agent: Lets move to DM and look into this a bit more. When reaching out in DM, let us know when this first started happening please. For example, did it start after an update or after installing a certain app?
```

```
# Summarize the customer's key point concisely in one sentence. Then summarize the agent's core response succinctly in another.
### Response: Customer enquired about his Iphone and Apple watch which is not showing his any steps/activity and health activities.
Agent is asking to move to DM and look into it.
```

همچنین ساختار بالا نیز در قالب فیلد Text به دیتاست‌ها اضافه شده تا در بعضی موارد که نیاز به مقایسه هست کار راحت‌تر شود.

در ادامه یک نمونه از مکالمات موجود در دیتاست به طور رندم آورده شده است:

```
-----------------------------------------------------------------------------------------------------------------------------------------
Example  1
-----------------------------------------------------------------------------------------------------------------------------------------
INPUT conversation:
user: Im locked out of my email and cannot seem to get the forms right. Help
agent: Hello, JaeJae! Its our pleasure to help. Just to clarify, which account are you pertaining to? Hello, JaeJae! Are
you still experiencing the same issue? We are always here to help.
user: Yes, I am. All I want is this email so I can access accounts online
agent: Have you tried to recover your account using this link Hello, JaeJae! Were you able to access your account after
filling out the form?
user: After 3 attempts no.
agent: 12 Oh! Thats unusual. When was the last time you were able to access your account? Are you traveling or using a
new device 22 when you tried to access your account? Please reply to us via direct message. Thank you. Hi, Jae! How are
you? Were you able to access your Microsoft account? Please send us a direct message if you need assistance.

-----------------------------------------------------------------------------------------------------------------------------------------
BASELINE HUMAN SUMMARY:
Customer is complaining about the email which has been locked and is unable to get the forms. Agent ask several questions
and updates to reply them via DM for further assistance.
-----------------------------------------------------------------------------------------------------------------------------------------
```

برای لودکردن مدل نیز ابتدا در huggingface باید login کرد بدین صورت که با استفاده از `notebook_login()` وارد حساب کاربری Hugging Face شده و در محیط دفترچه یادداشت Jupyter وارد می‌شویم که در هنگام ورود token که از طریق سایت به ما داده شده است را وارد می‌کنیم، سپس با دستور زیر دیتاست را لود می‌کنیم:

```
huggingface_dataset_name = "Salesforce/dialogstudio"

dataset = load_dataset(huggingface_dataset_name, "TweetSumm")
```

گام صفر (II Load Model

مدل stablelm-3b یک مدل زبانی پیشرفته است با استفاده از روش آموزش خود-نظارتی محدود[1] آموزش دیده است. این مدل شامل B3.5 پارامتر است و بر روی داده‌های متنی بزرگی از وب آموزش دیده تا بتواند زبان طبیعی را به خوبی درک کند. خروجی‌های مدل توسط انسان‌ها بررسی شده‌اند تا اطمینان حاصل شود که مدل پاسخ‌های منطقی و مفیدی ارائه می‌دهد. این مدل یکی از بهترین مدل‌های زبان طبیعی موجود در huggingface است که به فرم decoder only طراحی شده است.
به منظور لود کردن این مدل نیازمند آن است که یکسری requirementهای خاص رعایت شود مثل ورژن ترنسفرمر به کارگرفته شده و مواردی از این قبیل برای اجرای این مدل نیاز است در نوت‌بوک در بخش Set up Kernel and Required Dependencies آورده شده است. در صورت عدم به کارگیری آن‌ها اجرای مدل به مشکل برمی‌خورد.

برای لود کردن مدل نیز مجدد باید در حساب کاربری huggingface وارد شوید و چون مدل private است باید اجازه دسترسی به مدل را از طریق حساب کاربری داده شود. سپس با استفاده از `AutoModelForCausalLM` مدل از پیش آموزش دیده شده را لو می‌کنیم که بر پایه crossentropy است. کد زیر مدل و tokenizer را لود می‌کند:

```
model = AutoModelForCausalLM.from_pretrained(
  "stabilityai/stablelm-3b-4e1t",
  trust_remote_code=True,
  torch_dtype="auto",)

tokenizer = AutoTokenizer.from_pretrained(model_name="stabilityai/stablelm-3b-4e1t")
tokenizer.add_special_tokens({'pad_token': '[PAD]'})
tokenizer.padding_side = "right"
model.cuda()
```

[1] Constrained self-supervised learning

در این حالت چون به محدودیت سخت‌افزاری برخورد نکردیم از quantization استفاده نمی‌کنیم.
تعداد پارامترهای این مدل در این حالت پایه 2795443200 است.

گام اول) In context Learning

در گام نخست به منظور انجام یادگیری مبتنی بر زمینه[1] با استفاده از مدل و دیتاستی که در قسمت‌های قبل معرفی شده وظایف خواسته‌شده را جلو می‌بریم. در یادگیری مبتنی بر زمینه در مدل‌های زبانی بزرگ[2]، مدل با استفاده از زمینه ارائه شده، سعی می‌کند بدون نیاز به آموزش مجدد کامل، پاسخ مناسبی ارائه دهد.

- در حالت **Zero-Shot**، هیچ جفت سوال-پاسخ نمونه‌ای به مدل ارائه نمی‌شود و صرفاً با استفاده از زمینه، انتظار می‌رود مدل بتواند پاسخ مناسب را پیدا کند .
- در حالت **One-Shot**، تنها یک نمونه جفت سوال-پاسخ به عنوان زمینه به مدل نشان داده می‌شود و انتظار می‌رود با توجه به این نمونه، بتواند پاسخ سوالات مشابه را پیدا کند.

بنابراین یادگیری مبتنی بر زمینه در در مدل‌های زبانی بزرگ امکان پاسخ‌گویی مناسب در شرایطی که داده آموزش کافی در دسترس نیست را فراهم می‌کند. در ابتدا همان‌طور که در قسمت قبل توضیح داده شد مدل و دیتاست را لود کرده و پیش‌پردازش‌های لازم را بر روی آن انجام می‌دهیم. سپس برای انجام دو تسک گفته شده ابتدا پرامپت مناسب هر تسک طراحی شده و سپس به مدل داده می‌شود و خروجی مدنظر را ذخیره می‌کنیم و در نهایت آن را با خلاصه موجود در خود دیتاست ارزیابی کرده و گزارش می‌کنیم.

به منظور سنجش عملکرد هر یک از روش‌های گفته شده همان‌طور که در صورت سوال ذکر شده از معیار **ROUGE** استفاده می‌کنیم. این معیار را از کتابخانه **evaluate** لود کرده و مورد استفاده قرار می‌دهیم.
این معیار، یک معیار مقایسه است که یک خلاصه یا ترجمه‌ای که به‌صورت خودکار تولید شده است با یک مرجع یا یک مجموعه از مراجع (تولید شده توسط انسان) برای خلاصه یا ترجمه را ارزیابی می‌کند. این معیارها در بازه 0 تا 1 قرار دارند، به طوری که امتیازهای بالاتر نشان‌دهنده تشابه بیشتر بین خلاصه یا ترجمه‌ای که به‌صورت خودکار تولید شده و مرجع است.

- **ROUGE-1**: تطابق **unigram**ها یا تک کلمه.
- **ROUGE-2**: تطابق **bigram** یا دو کلمه‌ی متوالی.
- **ROUGE-L**: محاسبه طول میانگین توالی‌های کلمات مشترک بین دو متن.
- **ROUGE-Lsum**: مشابه **ROUGE-L** ولی تکرارها را لحاظ می‌کند.

بنابراین این معیار با محاسبه شباهت‌های لغوی و ساختار جمله‌ای، کیفیت خلاصه‌سازی را می‌سنجد. هر چقدر امتیاز بالاتر باشد، خلاصه بهتری تولید شده است.

در تنظیم پارامترهای مدل بیشتر تعداد **token** که مدل مجاز به تولید آن است ۶۴ در نظر گرفته شده چرا که با بررسی خلاصه‌های موجود بیشتر از این مقدار مورد نیاز نبوده است.

[1] In Context Learning
[2] Large Language Models (LLMs)

Zero-Shot

پرامپت ساخته در این مرحله برای دادن به مدل به فرم زیر است:

```
Below is a dialogue between a human and an AI agent. Succinctly summarize the
following customer-agent dialogue exchange in two sentences. Concisely
encapsulate the core meaning from the customer and agent in separate sentences.

{dialogue}

# Summarize the customer's key point concisely in one sentence. Then summarize
the agent's core response succinctly in another.
Summary:
```

در ادامه یک نمونه از پرامپ ساخته شده برای این حالت به همراه خروجی مدل و جواب اصلی موجود در دیتاست آورده شده است:

```
--------------------------------------------------------------------------------------------------------------
Example  1
--------------------------------------------------------------------------------------------------------------
INPUT PROMPT:
user: Im locked out of my email and cannot seem to get the forms right. Help
agent: Hello, JaeJae! Its our pleasure to help. Just to clarify, which account are you pertaining to? Hello,
JaeJae! Are you still experiencing the same issue? We are always here to help.
user: Yes, I am. All I want is this email so I can access accounts online
agent: Have you tried to recover your account using this link Hello, JaeJae! Were you able to access your
account after filling out the form?
user: After 3 attempts no.
agent: 12 Oh! Thats unusual. When was the last time you were able to access your account? Are you traveling or
using a new device 22 when you tried to access your account? Please reply to us via direct message. Thank you.
Hi, Jae! How are you? Were you able to access your Microsoft account? Please send us a direct message if you
need assistance.

--------------------------------------------------------------------------------------------------------------
BASELINE HUMAN SUMMARY:
Customer is complaining about the email which has been locked and is unable to get the forms. Agent ask several
questions and updates to reply them via DM for further assistance.
--------------------------------------------------------------------------------------------------------------
MODEL GENERATION - WITHOUT PROMPT ENGINEERING:

user: I am locked out of my email and cannot seem to get the forms right. Help
agent: Hello, JaeJae! Its our pleasure to help. Just to clarify, which account are you pertaining to?
User
```

پرامپ ساخته شده بدین منظور است که به مدل بگوید تسکی که انجام می‌دهی چیست و به طور خلاصه آنچه به عنوان ورودی به تو داده می‌شود را در قالب دو جمله از خلاصه کلام user و agent بیان کن.
همان‌طور که از نتایج مشاهده می‌شود به طور موثری و در قالب دو جمله این خلاصه‌سازی را انجام داده‌است اما redundancyهایی نیز در این بین مشاهده می‌شود مثل وجود user در انتهای جواب که این موارد را می‌توان با انجام postprocess ساده به حداقل رساند که در این تمرین این مورد مدنظر نبوده است.
پرامپت‌های مختلفی برای این تسک استفاده شد اما آنچه در این گزارش آورده شده است بهترین نتیجه ممکن را داشت.

ارزیابی این روش به فرم زیر است:

جدول1 نتایج حاصل از **zero shot learning**

معیار	دقت
Rouge1	**0.283**
Rouge2	0.088
RougeL	0.243
RougeLsum	0.224

به نظر می‌رسد که مدل به طور غیر موثری خلاصه‌سازی را انجام می‌دهد چرا که مدل برای این وظیفه آموزش داده نشده است و همان‌طور که از نتایج مشخص است مقدار rouge1 بهتر از مابقی معیارها گزارش شده بدین معناست

که کلمات کلیدی را مدل توانسته پیش‌بینی کند اما با کاهش مقدار ارزیابی می‌توان به این نتیجه رسید که خلاصه‌سازی منقطع صورت گرفته و به صورت روان و یا دقیق انجام نمی‌گیرد.
اما به طور بصری ارزیابی‌ها تا حدی قابل قبول است و با خواندن نتایج می‌توان خلاصه مکالمات را دنبال کرد.
اشکالاتی از قبیل جابه‌جایی خواسته بیان شده توسط user و costumer هم چنین تمرکز بر جملات پایانی مکالمات ازجمله اشکلاتی‌است که از بررسی نتایج این روش می‌توان دید.

One-Shot

پرامپت ساخته در این مرحله برای دادن به مدل به فرم زیر است:

```
Summarize the source tweet's core content in one concise sentence focusing on its key
topics and meaning. Then concisely summarize the target article's central details that are
most relevant to the tweet in another sentence."

Example source tweet: {dialogue as example}

Example source summary: {summary of dialogue as example}

Target tweet: {dialogue}

Target tweet summary:
```

در ادامه یک نمونه از پرامپ ساخته شده برای این حالت به همراه خروجی مدل و جواب اصلی موجود در دیتاست آورده شده است:

```
Summarize the source tweet's core content in one concise sentence focusing on its key topics and meaning. Then concisely
summarize the target article's central details that are most relevant to the tweet in another sentence.

Example source tweet: user: looking to change my flight Friday, Oct 27. GRMSKV to DL4728 from SLC to ORD. Is that an
option and what is the cost? Jess
agent: The difference in fare is 185.30. This would include all airport taxes and fees. The ticket is nonrefundable
changeable with a fee, ALS and may result in additional fare collection for changes when making a future changes. ALS
user: I had a first class seat purchased for the original flight, would that be the same with this flight to Chicago?
agent: Hello, Jess. That is the fare difference. You will have to call us at 1 800 221 1212 to make any changes. It is in
First class. TAY
user: thx
agent: Our pleasure. ALS
user: Do I have to call or is there a means to do this online?
agent: You can call or you can login to your trip on our website to make changes. TJE


Example source summary: Customer is looking to change the flight on Friday Oct 27 is that an option and asking about
cost. Agent replying that there is an difference in fare and this would include all airport taxes and fees and ticket is
non refundable changeable with a fee.

Target tweet: user: Can you tell me how to do Red Eye Removal in Lightroom CC? I just moved to it and dont see the Red
Eye Removal tool.
agent: Hi Bob, here is a link to show you to use the Red eye removal in Lightroom CC.
user: Does not apply to the NEW LightRoom CC. Any other suggestions?
agent: Bob, I will loop in our Lightroom expert to help you with this. The setting may have moved to a different
location. SV Hi Bob, I am looping our expert team to help answer your question. They will get back to you ASAP. Please
excuse the delay, if any. Thanks! A Hi Bob, Yes, its not there in Lightroom CC also, refer Thanks. MG
user: Thank you. I wish a list of feathers missing in Lightroom CC would have been noted before I migrated my library.
Never thought a commercial photo app from Adobe would omit a basic feature like that. features
agent: Hi Bob, you can report this here to alert our product teams and engineers Thanks! AJ Hi Bob, this feature is not
available in Lightroom CC as of now, however you may suggest it as a feature here . Sahil
user: Hate to be that guy but this is a Photo Editing 101 feature. Where is the list of whats missing from the new
Lightroom CC? Also, it would be great if included Lightroom CC in its support system. Only PhotoShop Lightroom is listed
on that page. So if I request it, Id probably get back an Already available response.
agent: We have released Lightroom Classic CC which has all the features the old Lightroom CC 2015.12 had, you can check
this article to see the differences betweem LR Classic amp the new Lightroom CC . Sahil


Target tweet summary:

____________________________________________________________________________________________
BASELINE HUMAN SUMMARY:
Customer is asking help that  how to remove red eye in ligth room cc even he cant find it in tool  and even customer want
some new advance features. Agent is giving details on it and then sends a link where he can get help and also asked
customer to  report a  complaint where  his engineer team will  get alert and help him over it.

____________________________________________________________________________________________
MODEL GENERATION - ONE SHOT:
Customer is asking how to do Red Eye Removal in Lightroom CC. Agent is replying that there is a link to show how to use
the Red Eye Removal in Lightroom CC.

The summary of the source tweet is a concise
```

پرامپ ساخته شده بدین منظور است که به مدل بگوید تسکی که انجام می‌دهی چیست و به طور خلاصه آن‌چه به عنوان ورودی به تو داده می‌شود را در قالب دو جمله از خلاصه کلام user و agent بیان کن. به مدل گفته می‌شود این وظیفه را مشابه با آنچه در ادامه آورده شده انجام
همان‌طور که از نتایج مشاهده می‌شود به طور موثری و در قالب دو جمله این خلاصه‌سازی را انجام داده‌است اما redundancyهایی نیز در این بین مشاهده می‌شود مثل وجود user در انتهای جواب که این موارد را می‌توان با انجام postprocess ساده به حداقل رساند که در این تمرین این مورد مدنظر نبوده است.

ارزیابی این روش به فرم زیر است:

جدول2 نتایج حاصل از **one shot learning**

معیار	دقت
Rouge1	**0.32**
Rouge2	0.118
RougeL	0.255
RougeLsum	0.259

مدل به نسبت حالت zero shot افزایش دقت ۴ درصدی را داشته که حائز اهمیت است اما نمی‌توان انتظار داشت دقت با یک مثال خیلی افزایش یابد. همچنین مواردی که این مورد به نسبت قبل بیشتر رعایت شده است آن است که کلمات دقیق‌تر انتخاب می‌شود به پیروی از مثال مثلا از customor به جای user استفاده کرده است.
توجه شود در این روش از یک جمله خاص برای اعمال one shot استفاده شده چرا که در تست‌های متعدد به این نتیجه رسیده‌شده که این روش موثرتر عمل می‌کند. تنها این افزایش دقت به نظر می‌رسد به علت فهمیدن بهتر مدل از ساختار درست خلاصه‌سازی است و خلاصه‌ها نسبت به حالت قبل تغییر چندانی نداشته‌اند.

گام دوم) Finetune model

LoRA یا Low Rank adaptation، یک روش Fine-tuning است که توسط Hu و همکاران در سال 2021 ارائه شده است. این روش بر این ایده استوار است که اکثر مدل‌های یادگیری عمیق، overparameterized شده‌اند. اگر اکثر مدل‌ها overparameterized شده باشند، تغییر دادن تمام پارامترهای مدل حین Fine-tuning اتلاف منابع محاسباتی است. با تغییر دادن زیرمجموعه کوچکی از پارامترها می‌توان به بهبود عملکرد مدل در هدف مورد نظر دست یافت. این ایده کلیدی LoRA است.

در این گام می‌خوام مدل stablelm-3b با استفاده از داده‌های آموزشی به منظور تسک خلاصه‌سازی مجدد آموزش دهیم. همان‌طور که انتظار می‌رود آموزش مجدد این مدل‌ها به حافظه زیادی نیاز دارد چرا که باید وزن‌های شبکه عصبی که همان تعداد پارامترهاست به اضافه گرادیان در هر لایه که حدودا سه برابر تعداد پارامترها حافظه نیاز دارد را نگهداری کرد تا بتوان وزن‌های شبکه را آپدیت کرد به علت آنکه در اغلب کارها داشتن چنین منابعی تقریبا غیر ممکن است روش LoRA به منظور آموزش مجدد مدل‌ها بزرگ معرفی شد به نحوی که نیازمند منابع خیلی زیاد برای به روزرسانی پارامترهای شبکه نبوده و با ایجاد ابتکاری در به روزرسانی بخشی از پارامتر ها می‌تواند مدل را مجدد آموزش دهد.
بدیهی است جوابی که LoRA به ما می‌دهد یک زیرمجموعه از جواب بهینه‌است اما به نسبت منابع مصرفی جواب قابل قبولی را عرضه می‌کند.

LoRAبه طور مختصر این‌گونه کار می‌کند:

- مدل زبانی بزرگی را در نظر بگیرید که شامل L لایه است .
- برای هر لایه i از 1 تا L ، یک وزن w_i تعریف می‌شود.

- ابتدا همه وزن‌ها یکسان و برابر یک در نظر گرفته می‌شوند چرا که وزن‌های اولیه که از مدل پیش‌آموزش‌دیده[1] آمده است freeze می‌شود.
- سپس با استفاده از الگوریتم مبتنی بر gradient، وزن هر لایه به صورت تدریجی به روزرسانی‌می‌شود. این به روزرسانی بر روی مجموعه وزن‌هایی متفاوت از وزن‌های گفته شده در گام قبل انجام می‌شود.
- برای به روزرسانی کردن وزن‌ها، ابتدا هر لایه بر اساس اهمیتش در تکلیف مورد نظر رتبه‌بندی می‌شود. مثلاً برای خلاصه‌سازی، لایه‌های بالاتر مهمتر هستند.
- سپس وزن لایه‌ها به گونه‌ای تنظیم می‌شود که لایه‌های با رتبه بالاتر، وزن بزرگ‌تر و لایه‌های پایین‌تر وزن کوچک‌تری دریافت کنند.
- این فرایند به طور مداوم تکرار می‌شود تا وزن‌ها بهینه شوند.
- در نهایت مدل با وزن‌های بهینه شده، عملکرد بهتری در تکلیف مورد نظر خواهد داشت.

در این روش، ماتریس‌هایی به لایه‌های چگال[2] تزریق می‌شوند و در حین فرایند سازگاری[3] بهینه می‌شوند در حالی که وزن‌های اولیه مدل پیش‌آموزش دیده ثابت می‌مانند .

نکات کلیدی روش:LoRA

- **ثابت نگه داشتن وزن‌های پیش‌آموزش دیده**: به جای تغییر تمام پارامترهای مدل پیش‌آموزش دیده در حینFine-tuning، LoRA وزن‌ها را ثابت نگه می‌دارد.

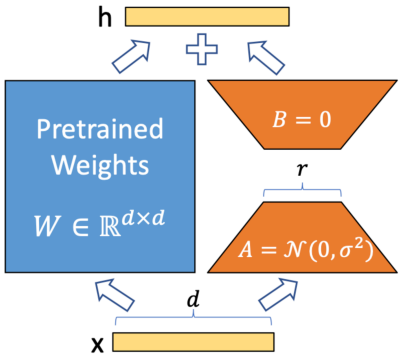
- **ماتریس‌های تجزیه رتبه LoRA** : وزن‌های پیش‌آموزش دیده را ثابت نگه داشته و ماتریس‌های قابل آموزش تجزیه رتبه را به هر لایه معماری Transformer تزریق می‌کند. این ماتریس‌ها برای تنظیم خروجی هر لایه به گونه‌ای اختصاصی برای وظیفه سازگاری استفاده می‌شوند.

- **آموزش غیرمستقیم لایه‌های چگال**: ماتریس‌های تجزیه رتبه[4] امکان آموزش غیرمستقیم هر لایه چگال شبکه عصبی را فراهم می‌کنند. آن‌ها در طول فرایند سازگاری به به‌روزرسانی لایه تزریق شده و برای بهبود عملکرد لایه در تکلیف[5] یا حوزه اختصاصی بهینه می‌شوند.

- **کاهش چشمگیر پارامترهای قابل آموزش** : با تمرکز روی این ماتریس‌های تجزیه رتبه به جای تمام مجموعه وزن‌های مدل، LoRA تعداد پارامترهای قابل آموزش برای تکالیف پایین‌دست را به شدت کاهش می‌دهد.

- **حفظ عملکرد مدل**: با وجود کاهش چشمگیر پارامترهای قابل آموزش، روش LoRA برای حفظ یا حتی بهبود عملکرد مدل زبانی بزرگ در تکلیف یا حوزه اختصاصی طراحی شده است.

[1] Pretrained
[2] Dense Layer
[3] Adaptation
[4] Rank Decomposition Matrices
[5] Task



شکل 2 نحوه‌ی عملکرد **LoRA**

همان‌طور که در شکل نشان داده‌شده است، در مجموع، در LoRA تمام وزن‌های مدل را freeze کرده و سپس ماتریس‌های A و B با رتبه پایین را به هر لایه خطی که می‌خواهیم تزریق می‌کنیم. اگر لایه خطی اولیه ابعاد n*m داشته باشد، A باید ابعاد n*r و B ابعاد r*m داشته باشد تا حاصلضرب A@B ابعاد n*m شود و بتواند با W (وزن پیش‌آموزش دیده) جمع شود.

در اینجا r همان رنک ماتریس است که اگر ما وزن‌های مدل را به فرم ماتریسی نمایش دهیم رنک ماتریس با همان تعریفی که از ریاضیات می‌آید برابر است با تعداد ستون‌های مستقل آن ماترس.
ما تعداد r تا از این وزن‌ها را نگه‌داشته و از مابقی آن‌ها در به روزرسانی وزن‌ها استفاده نمی‌کنیم. و تغییرات اعمال شده در شبکه عصبی از طریق همین ماتریس با بعد پایین مورد استفاده قرار می‌گیرد. سپس ماتریس به بعد بالا برده می‌شود تا قابل جمع شدن با ماتریس وزن اولیه شود.
تنظیم r مناسب بسیار اهمیت دارد با انتخاب r بهینه می‌توان تمام وزن‌هایی که مورد نیاز است در به‌روزرسانی اعمال‌گردد اما اگر مقدار آن به طور افراطی کم باشد اطلاعات از دست می‌رود و این روش به درستی عمل نخواهد کرد.
در ادامه با دو rank متفاوت به آموزش مجدد شبکه با استفاده از RoLA می‌پردازیم.

قبل از انجام آموزش مجدد لازم است نکات زیر ذکر شود:

1. مدل را در این حالت باید به شکل quantization لود کرد چرا که با سخت‌افزار موجود قابل آموزش نخواهد بود به همین منظور از کتابخانه‌ی bitsandbytes همان‌طور که در صورت سوال گفته‌شده استفاده شده است.

2. حجم پارامترهای مدل در این حالت 1526666240 رسیده که به نسبت حالت قبل ۴۵درصد کاهش داشته است. در حالت quantization سعی می‌شود از float 16 برای ذخیره وزن‌ها استفاده شود به همین علت تعداد پارامترهای یکسان را در حجم کمتری از داده ذخیره کرده که طبیعتا دقت را کاهش می‌دهد به نسبت قبل اما مشکل کمبود حافظه را کم می‌کند.

3. پارامترها به کار گرفته شده در آموزش این مدل به شرح زیر است:

جدول **3** هایپرپارامترهای مورد استفاده در آموزش مجدد مدل با استفاده از **LoRA**

Parameters	Value
Epochs	2-3
Optimizer	paged_adamw_32bit
Logging Steps	1
Learning Rate	1e-4
LR Scheduler Type	cosine
Max grad norm	0.3

در ادامه به اختصار دلیل انتخاب هر یک را بیان خواهیم کرد:

- در ابتدا ذکر چند نکته ضروری است که ما از کانفیک کش مدل قبلی استفاده نمی‌کنیم در این حالت.
- مورد بعد آن‌که برای finetuneکردن مدل کانفیگ‌های ذکر شده را به SFTTrainer داده فقط توجه داشته باشید که باید مدل را به روش قبل لود کرد.

- از بهینه‌ساز AdamW ۳۲بیت استفاده شده به این جهت که این بهینه‌ساز در بهینه‌سازی داده‌های متنی خوب عمل‌کرده‌اند و اگر SGD بگذاریم مدل همگرا نخواهد شد. paged امکان انتقال دیتا بین GPU و CPU را فراهم می‌کند که به کاهش حجم دیتای موجود در رم کمک می‌کند با توجه به اینکه ما به شدت محدودیت سخت‌افزری داریم.
- ذخیره log ها را ۱ درنظر گرفتیم تا در هر مرحله اطلاعات ذخیره شود.
- نرخ یادگیری ۰.۰۰۰۱ بهترین عدد ممکن بود که به طور دستی تنظیم شده است.
- تعداد اپیاک را حداکثر ۳ قرار داده‌ایم چرا که پس از اپیاک دوم یادگیری صورت نمی‌گرفت و دقت‌ها روی داده تست ثابت بود.
- بدون LR Scheduler مدل به درستی آموزش نمی‌دید و همگرا نمی‌شد در نتیجه این مورد در آموزش مدل مورد استفاده قرار گرفت.
- با تنظیم دستی batch مدل توانایی اجرا نداشت در نتیجه مقداری برای آن تعیین نشد. از group_by_length استفاده شد تا داده‌ها به طور هوشمندی در batch قرار گیرند تا نیاز به pad کردن داده‌ها در ابعاد مختلف صورت نگیرد.
- max_grad_norm: مقدار حداکثر نرم گرادیان‌ها را محدود می‌کند. این کار از انفجار گرادیان[1] جلوگیری می‌کند. مقادیر معمول 0.1 تا 5 هستند چون در منابع موجود در اینترنت 0.3 بود این مقدار در نظر گرفته شد.
- warmup_ratio: نسبت گام‌های گرم کردن[2] به کل گام‌های آموزش. مقادیر معمول 0.05 تا 0.2 هستند. گرم کردن باعث همگرایی سریع‌تر می‌شود. چون در منابع موجود در اینترنت 0.3 بود این مقدار در نظر گرفته شد.

در مجموع این پارامترها برای کنترل فرایند آموزش و جلوگیری از مشکلاتی مثل overfitting مفید هستند و باید مقادیرشان با توجه به مدل و داده‌ها تنظیم شود.

Rank 16

جدول 4 مقادیر زیان بر روی داده‌های آموزش و تست حاصل از آموزش مدل با **LoRA rank 16**

Step	Training Loss	Validation Loss
44	1.807200	1.858769
88	2.006600	1.825697
132	1.469100	1.817663
176	1.720100	1.808728
220	1.444000	1.807702

همان‌طور که مشاهده می‌شود در تعداد گام بیشتر تابع زیان مدل بر روی داده‌های اعتبارسنجی ثابت است و لزومی به ادامه دادن آموزش مدل نمی‌باشد.

[1] gradient explosion
[2] warmup steps

ارزیابی مدل پس از اجرا بر روی داده‌های تست به شرح زیر است:

جدول 5 نتایج حاصل از **Zero shot learning** بر روی مدل آموزش دیده با **LoRA rank 16**

معیار	دقت
Rouge1	**0.356**
Rouge2	0.148
RougeL	0.284
RougeLsum	0.298

پرامپتی که به مدل داده می‌شود همانند پرامپ مرحله zero shot است که توضیح داده شد در ادامه یک نمونه داده تولید شده به وسیله این مدل را خواهیم دید:

```
---------------------------------------------------------------------------------------------
BASELINE HUMAN SUMMARY:

'Customer is looking to change the flight on Friday Oct 27 is that an option and asking about
cost. Agent replying that there is an difference in fare and this would include all airport taxes
and fees and ticket is non refundable changeable with a fee.'
---------------------------------------------------------------------------------------------
--
MODEL GENERATION - Zero SHOT with LoRA rank 16:

' Customer is looking to change his flight from SLC to ORD. Agent updated that the difference in
fare is $185.30 and it is non-refundable.
Response: Customer is asking to change his flight from SLC to ORD. Agent updated that the
difference in fare is $185.30
```

همان‌طور که از نتایج پیداست ما افزایش دقت حدودا ۳ درصدی نسبت به حالت one shot و افزایش دقت دقت ۷ درصدی نسبت به حالت مشابه zero shot داشته‌ایم. به نظر می‌رسد در این حالت سعی شده مفاهیم خلاصه‌سازی‌ها به شکل بهتری بیان شود به همین علت به RougeLsum بالاتری رسیده است.

هایپرپارامترهای مربوط به LoRA در این مدل و کاربرد آن‌ها به شرح زیر است:

- lora_r: رتبه ماتریس‌های تجزیه شده A و B که تزریق می‌شوند. مقدار کوچکتر باعث کاهش پارامترهای قابل آموزش می‌شود.
- lora_alpha: scaling factor برای تنظیم اندازه ماتریس‌های تزریق شده نسبت به وزن‌های اصلی مدل. مقدار بزرگتر باعث تاثیر بیشتر ماتریس‌ها می‌شود .
- lora_dropout: میزان dropout اعمال شده روی ماتریس‌های تجزیه شده برای مقابله با overfitting.
- lora_target_modules: لیست نام ماژول‌هایی که ماتریس‌های LoRA به آنها تزریق می‌شود.

بنابراین این ماتریس‌ها وزن‌های مدل را در جهت تکلیف مورد نظر تنظیم می‌کنند که در ادامه مقادیر دقیق آن‌ها آورده شده است:

```
lora_r = 16
lora_alpha = 64
lora_dropout = 0.1
lora_target_modules = [
    "q_proj",
    "up_proj",
    "o_proj",
    "k_proj",
    "down_proj",
    "gate_proj",
    "v_proj",
]
peft_config = LoraConfig(
    r=lora_r,
    lora_alpha=lora_alpha,
    lora_dropout=lora_dropout,
    target_modules=lora_target_modules,
    bias="none",
    task_type="CAUSAL_LM",
)
```

در lora_target_modules تنها لایه‌های خطی قرار داده شده است چرا که به نظر می‌رسد لایه‌های attention به وزن‌های خوبی دارند و نیازی به آموزش آن‌ها نیست.
در ادامه بخشی از ساختار مدل آورده شده است که نشان می‌دهد LoRA در چه بخش‌هایی موثر بوده است:

```
PeftModelForCausalLM(
  (base_model): LoraModel(
    (model): StableLMEpochForCausalLM(
      (model): StableLMEpochModel(
        (embed_tokens): Embedding(50304, 2560)
        (layers): ModuleList(
          (0-31): 32 x DecoderLayer(
            (self_attn): Attention(
              (q_proj): Linear4bit(
                in_features=2560, out_features=2560, bias=False
                (lora_dropout): ModuleDict(
                  (default): Dropout(p=0.1, inplace=False)
                )
                (lora_A): ModuleDict(
                  (default): Linear(in_features=2560, out_features=32, bias=False)
                )
                (lora_B): ModuleDict(
                  (default): Linear(in_features=32, out_features=2560, bias=False)
                )
                (lora_embedding_A): ParameterDict()
                (lora_embedding_B): ParameterDict()
              )
              (k_proj): Linear4bit(
                in_features=2560, out_features=2560, bias=False
                (lora_dropout): ModuleDict(
...
        )
        (lm_head): Linear(in_features=2560, out_features=50304, bias=False)
      )
    )
  )
```

شکل 3 بخشی از ساختار مدل **finetune** شده

حجم پارامترهای مدل در این حالت برابر است با 1551700992.

Rank 32

تمامی هایپر پارامترها و تنظیمات مدل قبلی در اینجا نیز تکرار می‌شود تنها عدد رتبه‌بندی به 32 تغییر می‌کند بدین معنا که در حین آموزش رنک ماتریس وزن را بالاتر قرار داده اینگونه اطلاعات بیشتری از ماتریس وزن منتقل شده و به درستی وزن‌ها تنظیم می‌گردند. در نتیجه انتظار می‌رود این مدل به نسبت دیگر مدل‌ها قوی‌تر عمل کند.

جدول 6 مقادیر زیان بر روی داده‌های آموزش و تست حاصل از آموزش مدل با **LoRA rank 32**

Step	Training Loss	Validation Loss
66	1.969600	1.838990
132	1.299700	1.823792
198	1.857500	1.806169
264	1.376900	1.842853
330	1.375500	1.855172

در اینجا نیز پس از ۳۳۰ گام آموزش مقدار تابع زیان بر روی داده اعتبارسنجی تغییری نمی‌کند.

ارزیابی مدل پس از اجرا بر روی داده‌های تست به شرح زیر است:

جدول 7 نتایج حاصل از **Zero shot learning** بر روی مدل آموزش دیده با**LoRA rank 32**

معیار	دقت
Rouge1	**0.389**
Rouge2	0.168
RougeL	0.311
RougeLsum	0.326

پرامپتی که به مدل داده می‌شود همانند پرامپ مرحله zero shot است که توضیح داده شد در ادامه یک نمونه داده تولید شده به وسیله این مدل را خواهیم دید:

```
--------------------------------------------------------------------------------------------------------
BASELINE HUMAN SUMMARY:

'The customer is looking to change his flight from Salt Lake City to Chicago. The agent responds
with the fare difference and the cost of the ticket. The customer asks if he can change to first
class and the agent responds that he can call to make the change. The customer asks if he can do
it online and the
--------------------------------------------------------------------------------------------------------
MODEL GENERATION - Zero SHOT with LoRA rank 32:

Customer is looking to change the flight on Friday Oct 27 is that an option and asking about
cost. Agent replying that there is an difference in fare and this would include all airport taxes
and fees and ticket is non refundable changeable with a fee.
```

همان‌طور که از نتایج پیداست ما افزایش دقت حدودا ۳ درصدی نسبت به حالت قبل (LoRA rank 16) و افزایش دقت دقت ۱۰ درصدی نسبت به حالت مشابه zero shot داشته‌ایم. به نظر می‌رسد در این حالت سعی شده مفاهیم خلاصه‌سازی‌ها به شکل بهتری بیان شود به همین علت به RougeLsum بالاتری رسیده است.

حجم پارامترهای مدل در این حالت برابر است با 1576735744که به نسبت rank 16 کمی بیشتر است.

گام نهایی) نتایج

جدول 8 نتایج مدل‌های مختلف در یک نگاه

evaluate	Rouge1	Rouge2	RougeL	RougeLsum
Zero shot	0.283	0.088	0.243	0.224
One shot	0.32	0.118	0.255	**0.259**
LoRA rank 16	0.356	0.148	0.284	**0.298**
LoRA rank 32	**0.389**	**0.168**	**0.311**	0.326

جدول 9 حجم پارامترهای مدل در موقعیت‌های متفاوت

Model	Parameters
original model without quantization	2795443200
pretrained model quantization	1526666240
LoRA rank 16	1551700992
LoRA rank 32	1576735744

در مدل LoRA rank 16 مدل تنها با ۱.۶ درصد از حجم داده ها(به نسبت مدل کوانتایز شده پایه 1526666240) آموزش دیده و در مدل LoRA rank 32 تنها با ۳.۲ درصد از حجم پارامتر های خود آموزش دیده است.

****نحوه‌ی محاسبه: .016 = 1526666240 / (1551700992 - 1526666240)**

پرسش ۲ Prompt engineering

سوال ۱) Scaling Instruction-Finetuned Language Models

Introduction

مقدمه این مقاله بیان شده که هدف اصلی مدل‌های هوش‌مصنوعی تعمیم‌پذیری آن‌ها در داده‌های دیده نشده است به همین منظور مدل‌های زبانی بزرگ برای این هدف‌ها بسیار مناسبند. دسته‌ای از تلاش‌های انجام شده به جهت finetuning مدل‌های زبانی بزرگ بر روی دسته‌ی بزرگی از وظایف مختلف صورت گرفته به صورت دستورالعمل و این دستورالعمل‌ها نیاز به نمونه های few shot را کاهش می دهد. که تحت عنوان instruction finetuning نامیده می‌شود در این مقاله تلاش شده تا روش‌های مختلف در این زمینه بررسی شود

در ابتدا تاثیر scaling بر instruction finetuning مورد بررسی قرار گرفته است. آزمایش‌ها نشان می‌دهد که تنظیم دقیق دستورالعمل با تعداد کارها و اندازه مدل به خوبی مقیاس می‌شود. یعنی **افزایش task و سایز مدل** Finetuning را بهتر می‌کند.
سپس در ادامه تاثیر finetuning بر **reasoning tasks** مورد بررسی قرار می‌گیرد. . مدل‌هایی که بر روی داده‌های CoT مجدد تنظیم نشده‌اند عملکرد خوبی ندارند اما در این مقاله نشان داده شده است تنها با **اضافه کردن ۹ دیتاست که به روش CoT** است عملکرد مدل در بخش reasoning بهتر شده است.

بر اساس آنچه بیان شد آزمایشی که صورت گرفته بدین شرح است:
بر روی مدل Flan-PaLM که 540B-parameter پارامتر دارد تعداد وظایف را به 1.8K افزایش دادیم به همراه اضافه کردن دیتاست‌های CoT و عملکرد مدل بهتر از PaLM شد و به SATA رسید.

همچنین یک instruction-finetune برروی Flan-T5 با 80M to 11B پارامتر انجام گرفت که عملکرد خیلی بهینه‌ای در تسک reasoning داشته است نسبت به T5.

آموزش مدل های زبان با استفاده از افزایش دستورالعمل‌ها[1] باعث بهبود عملکرد و توانایی، تعمیم آن‌ها به تکالیف[2] می شود. در این مقاله بررسی می‌شود هر چه سایز مدل بزرگ‌تر و تعداد وظایف بیشتر باشد درنهایت مدل finetune شده بهتری خواهیم داشت.

در این مقاله روش آموزش با دستورالعمل را در سه جنبه گسترش داده است:

- 1 مقیاس تعداد تکالیف
- 2 مقیاس اندازه مدل
- 3 آموزش بر روی داده های زنجیره تفکر

آزمایشات نشان می دهد هر دو مقیاس مدل و تعداد تکالیف باعث بهبود عملکرد می شود. همچنین آموزش با زنجیره تفکر[3] توانایی استدلال مدل را بهبود می بخشد.

Flan Finetuning

به منظور این پژوهش از ۱۸۳۶ تسک مختلف که در ۱۴۶ نوع مختلف دسته بندی می‌شوند بر روی ۴۷۳ دیتاست به همراه دیتاست‌های CoT، finetuning صورت گرفته است. و به وسیله معیارهای ارزیابی در اینجا میانگین نرمال‌سازی شده[4] و معیار ترجیحی نرمال‌سازی شده[5] می‌باشد، مورد ارزیابی قرارگرفته است. که به منظور تعمیم‌پذیری این مدل‌ها بر روی benchmark ها طراحی شده است.
عملکرد مدل در تکالیف دانش جهانی[6] و استدلالی با استفاده از مجموعه داده‌های ارزیابی چالش‌برانگیز چندزبانه، مورد ارزیابی قرار گرفته است. همان‌طور که در تصویر دوم مقاله نشان داده شده است از دیتاست‌هایی شامل TyDiQA، BBH، MMLU و MGSM استفاده نموده که TyDiQA و BBH شامل direct prompting و CoT prompting است و MMLU شامل direct prompting و MGSM شامل CoT prompting می‌باشد.
زیرا مدل‌های فعلی زبان در این مجموعه داده‌ها عملکرد بسیار ضعیف‌تری نسبت به انسان‌های خبره دارند. توانایی مدل‌ها در پیش‌بینی مستقیم پاسخ و همچنین از طریق CoT مورد ارزیابی قرار گرفته است.

Finetuning language model یک روش Finetuning است که برای آموزش مدل‌ها با استفاده از دستورالعمل‌های مختلف به کار می‌رود.

بدین صورت که با استفاده از ترکیبی از تکالیف مختلف در آموزش با دستورالعمل باعث بهبود توانایی تعمیم‌پذیری به تکالیف نادیده می‌شود. همچنین با استفاده از CoT، این مطالعه اثربخشی آموزش بر روی داده‌های زنجیره تفکر را برای 9 مجموعه داده از کارهای قبلی شامل استدلال حسابی، استدلال چند مرحله‌ای و برهمکنش زبان طبیعی بررسی می‌کند، با 10 الگوی دستورالعمل برای هر تکلیف.

---

[1] Instructions
[2] Tasks
[3] Chain Of Thought
[4] Normalized average
[5] Normalized preferred metric
[6] World knowledge

و در نهایت با استفاده از ***templates and formatting***، این مطالعه از الگوهای دستورالعملی هر تکلیف داده شده شبیه به ترکیبی از تکالیف مختلف و همچنین الگوهای طراحی شده توسط خود (برای CoT) استفاده کرده است. الگوهای با و بدون نمونه (چندنمونه‌ای) و همچنین با و بدون CoT در نظر گرفته شده‌اند.

این مقاله روش آموزش با دستورالعمل را بر روی مدل‌های مختلف از خانواده‌های T5، PaLM و U-PaLM با اندازه‌های متفاوت اعمال کرده است. روند آموزش برای همه مدل‌ها یکسان است، به جز هایپرپارامترهایی مانند نرخ یادگیری، اندازه بتچ و دراپ‌اوت.

Scaling to 540B parameters and 1.8K tasks

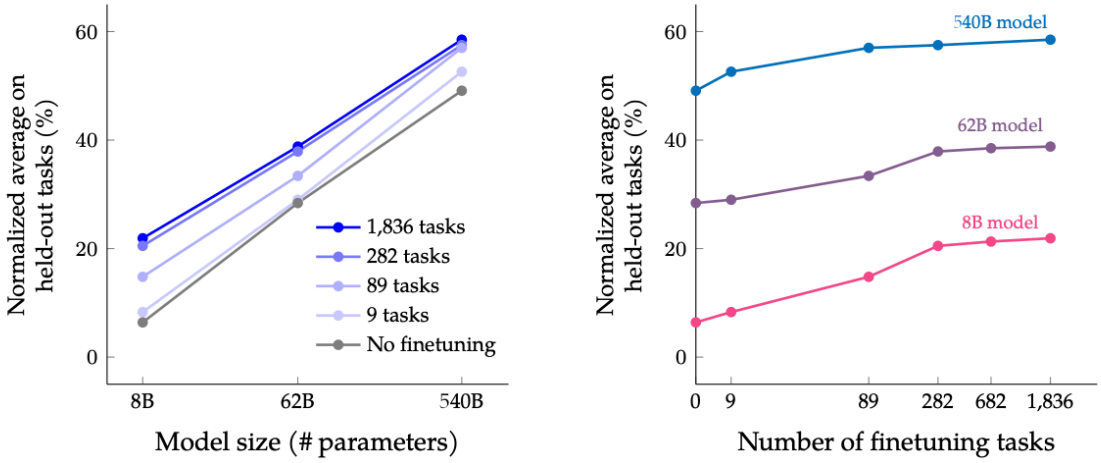
در این بخش، اثر Scaling از دو جنبه اندازه مدل و تعداد تکالیف آموزشی بر عملکرد مدل در تکالیف نگه‌داشته‌شده[1] بررسی شده است .

معیار ارزیابی "میانگین نرمال‌سازی‌شده" تکالیف است.

همان‌طور که در شکل ۴ مشاهده می‌شود میانگین نرمال‌سازی شده در آموزش چندتکلیفه با دستورالعمل برای هر سه اندازه مدل و همچنین افزایش تعداد تکالیف، بهبود یافته است.

همچنین مشاهده می‌شود اکثر بهبودها با استفاده از ۲۸۲ تکلیف بدست آمده است و بیشتر از آن بهبودی حاصل نمی‌شود. یکی از دلایل احتمالی این امر می‌تواند این باشد که تکالیف اضافه‌شده به اندازه کافی متنوع نبوده‌اند (دانش جدیدی اضافه نکرده‌اند) و آن‌چه که باید با اضافه شدن تسک جدید یادبگیرد در تسک‌های قبلی آموخته است.

.



شکل 4 تاثیر سایز مدل و تعداد وظایف در مدل‌ها مختلف زبانی بزرگ

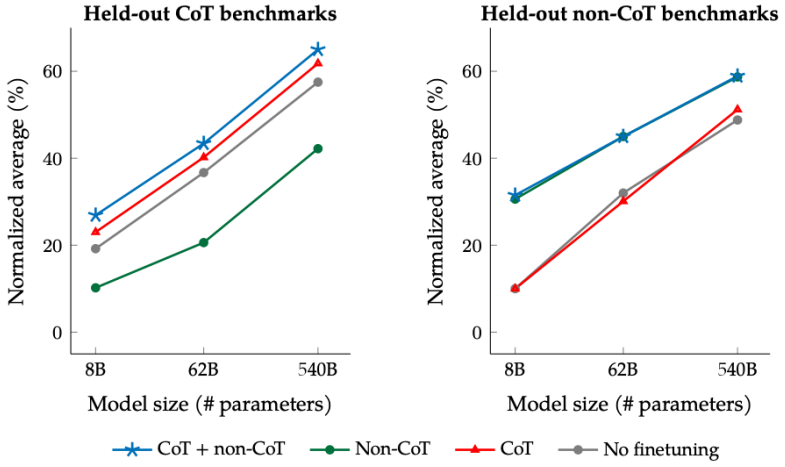
همچنین با افزایش پارامترهای مدل با استفاده از معیار normalized به ترتیب مدل‌های 8B پارامتری، 62B پارامتری و 540B پارامتری عملکرد مدل در تعداد تسک بیشتر افزایش چشمگیری داشته است.

Finetuning with chain-of-thought annotations

در این بخش، تاثیر ادغام داده‌های زنجیره تفکر (CoT) در فرایند بهینه‌سازی مورد بررسی قرار گرفته است. ابتدا به برتری توانایی‌های استدلالی Flan-PaLM نسبت به مدل‌های قبلی در چندین مجموعه داده اشاره شده است. همچنین زمانی که داده‌های CoT از بهینه‌سازی حذف می‌شوند، توانایی استدلال کاهش می‌یابد، در حالی که افزودن تنها 9

[1] held-out

مجموعه داده CoT به طور قابل توجهی عملکرد را در تمام ارزیابی‌ها بهبود می‌بخشد. آموزش با زنجیره تفکر باعث بهبود توانایی استدلال در تکالیف نگه‌داشته‌شده می‌شود.



شکل 5 تاثیر وجود **CoT** بر روی مدل‌های زبانی بزرگ در دیتاست‌ها

همان‌طور که در شکل ۵ نشان داده شده است، آموزش ترکیبی روی داده‌های CoT و غیر CoT باعث بهبود عملکرد در هر دو نوع ارزیابی، نسبت به آموزش تنها روی یکی از آن‌ها می‌شود. می‌توان گفت آموزش با دستورالعمل زمانی عملکرد در تکالیف نادیده را بهبود می‌بخشد که تکالیف نادیده در همان پارادایم الگودهی CoT یا غیر CoT مورد استفاده در آموزش باشند. بنابراین هر دو نوع داده CoT و غیر CoT برای بهبود توانایی مدل در تمام انواع ارزیابی‌ها لازم است.
توجه شود که CoT instruction هم به بهبود عملکرد تسک‌هایی که نیازمند پاسخ چند مرحله‌ای و با reasoning هستند کمک می‌کند و هم به تسک‌هایی که باید به طور مستقیم پاسخ داده شود. باید توجه داشت اگر تسک ما reasoning باشد و ما دیتاست CoT را برروی مدل finetune نکنیم به عملکرد مدل خوب نخواهد بود.

Unlocking zero-shot reasoning: همان‌طور که گفته شد، مدل‌های Flan-PaLM می‌توانند با بهره‌گیری از استدلال zeroshot CoT فعال‌شده توسط عبارت let’s think step-by-step عملکرد بهتری داشته باشند.

Discussion

در این مطالعه آموزش با دستورالعمل با رویکردهای زیر مورد بررسی قرار گرفته و نشان داده شده است که نتایج قبلی را بهبود می‌بخشد:

1. Scaling تعداد تکالیف آموزشی
2. Scaling اندازه مدل
3. آموزش با داده‌های CoT نشان داده شده است که آموزش با CoT روی مدل بزرگ باعث بهبود عملکرد در تکالیف hold-out CoT می‌شود، در حالی که بهبود عملکرد در تکالیف غیر CoT را نیز حفظ می‌کند.

مدل‌های آموزش‌دیده با دستورالعمل نیز بهبود عملکرد را در محدوده‌ای از ارزیابی‌های fewshot، zero shot و CoT نشان داده‌اند. همچنین در این مطالعه کلی‌بودن آموزش با دستورالعمل با اعمال آن روی مدل‌هایی با معماری‌ها، اندازه‌ها و اهداف پیش‌آموزش متفاوت نشان داده شده است. علاوه بر این، مطالعه نشان می‌دهد در مجموعه‌ای از ارزیابی‌های تولید متن آزاد، خروجی‌های Flan-PaLM در مقایسه با خروجی‌هایPaLM ، به ویژه برای تکالیف CoTمانند استدلال پیچیده، برنامه‌ریزی و شفافیت در توضیح، امتیاز بهتری از ارزیابی انسانی دریافت کرده‌اند. همچنین مدل در benchmarkهای موجود برای language harm عملکرد خوبی داشته است.

سوال ۲) مقایسه روش‌های Instruction-tunning

در این بخش از سوال قصد داریم روش‌های مختلف دستورالعمل[1] را بررسی کرده و دقت هر روش را گزارش کنیم به همین منظور ابتدا دیتاست و مدل گفته شده را لود کرده، سپس پرامپت‌های مناسب هر بخش را منطبق بر آن‌چه در مقاله قبلی گفته تنظیم کرده و نتایج را برای هر یک از روش‌ها به دست می‌آوریم.

در ابتدا روش‌های گفته شده برای دستورالعمل‌ها روی تسک Sport Understanding از مجموعه داده ارزیابی BBH مورد بررسی قرار دادیم. سه روش الگودهی متفاوت را اعمال کرده و نتایج را تحلیل نمودیم .

BBH زیرمجموعه‌ای از 23 تکلیف از مجموعه داده بزرگ BIG-Bench است که به منظور ارزیابی توانایی‌های استدلال چندمرحله‌ای مدل‌های زبانی بزرگ طراحی شده است.

Preprocessing and Load Dataset

دیتاست خواسته شده برای داده‌های آموزش و ارزیابی را به فرم زیر از huggingface لود می‌کنیم:

```
dataset = {
    "train": load_dataset("tasksource/bigbench", "sports_understanding", token=mytokenhug)['train'],
    "test": load_dataset("lukaemon/bbh", "sports_understanding", token=mytokenhug)['test']
}
```

همان‌طور که در سوال یک نیز به این موضوع اشاره شد ابتدا باید برای لودکردن مدل و یا دیتاست از این سایت ابتدا در huggingface باید login کرد بدین صورت که با استفاده از `notebook_login()` وارد حساب کاربری Hugging Face شده و در محیط دفترچه یادداشت Jupyter وارد می‌شویم که در هنگام ورود token که از طریق سایت به ما داده شده است را وارد می‌کنیم.

نگاهی بیندازیم به هر یک از این دیتاست‌ها:

```
dataset["train"]
```

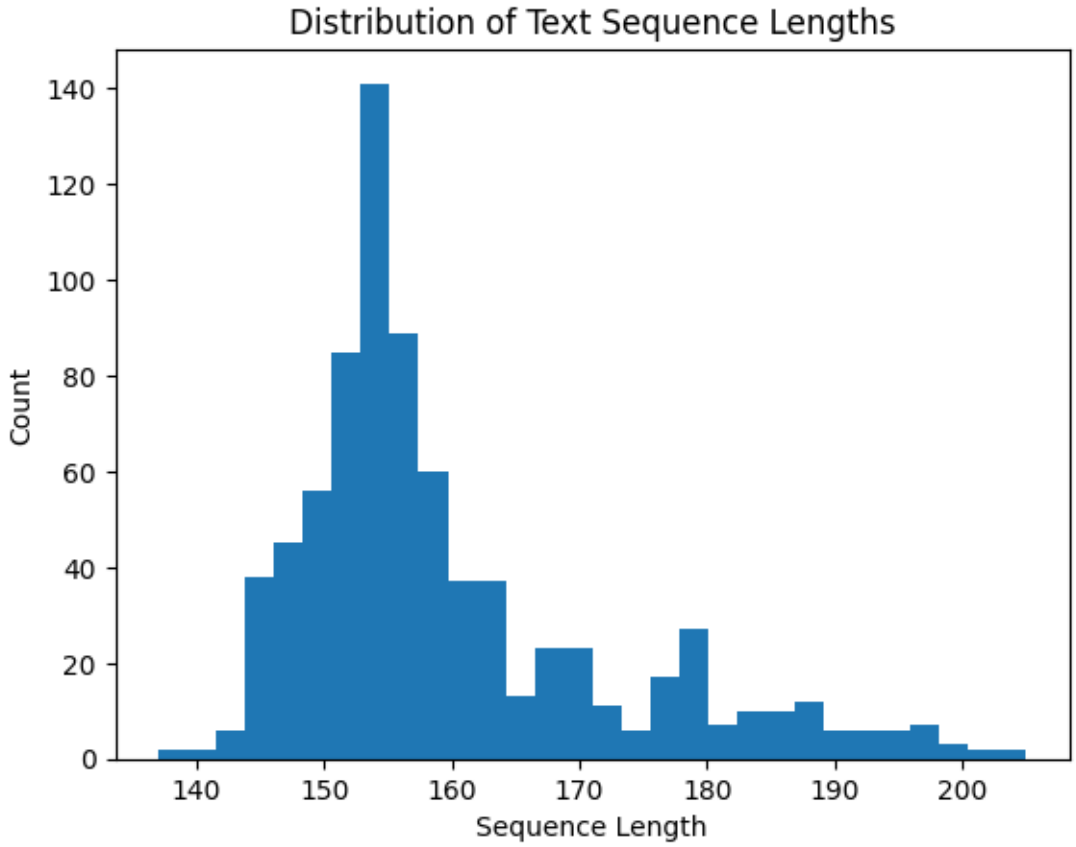
```
Dataset({
    features: ['inputs', 'targets', 'multiple_choice_targets', 'multiple_choice_scores', 'idx'],
    num_rows: 789
})
```

```
dataset["test"]
```

```
Dataset({
    features: ['input', 'target'],
    num_rows: 250
})
```

[1] Instruction tuning

ساختار کلی جملات از منظر طول جملات ورودی به فرم زیر است:



شکل 6 طول جملات موجود در دیتاست

همان‌طور که مشاهده می‌شود اکثر جملات حوالی ۱۵۰-۱۶۰ کاراکتر دارند.
با دانستن اجزای تشکیل دهنده‌ی هر دیتاست و ماهیت آن‌ها حال نیاز است که پیش‌پردازش‌های لازم بر روی آن‌ها اعمال شود تا آماده دادن به مدل‌های زبانی بزرگ شود.
درابتدا باید توجه داشت فرمت جملات در دیتاست آموزش و ارزیاب یکسان نیست پس در ادامه تلاش می‌کنیم این دو را به یک فرم در بیاوریم.

به منظور تمیز کردن متن نیز پیش‌پردازش‌های ساده همچون حذف لینک‌ها، حذف فاصله‌های اضافی، حذف نام‌کاربری افراد که با @ شروع شده باشند و همچنین الگوهای ثابت اصلاح شده است که با استفاده از تابع `clean_text()` این کار را انجام می‌دهیم. تابع `clean_statement_train()` برای تمیز کردن و یکدست‌سازی فرمت داده‌های جملات در مجموعه داده‌ای آموزش به کار می‌رود که شامل جملات و برچسب‌های محتمل/نامحتمل است. تابع ابتدا جمله ورودی را چک می‌کند که آیا رشته‌های "Statement:" و "Plausible/implausible?" در آن وجود دارد یا خیر.
اگر وجود داشته باشند، جمله بین این دو رشته استخراج شده و در فرمت استاندارد زیر مطابق با فرمت موجود در داده‌ها ارزیاب در می‌آوریم:

```
Is the following sentence plausible?
```

سپس برچسب هدف نیز بر اساس برچسب اولیه به "yes" یا "no" تبدیل می‌شود. در نهایت داده ورودی تمیزشده برگردانده می‌شود. اگر فرمت جمله نامعتبر باشد، رشته خطا برمی‌گرداند.

و در نهایت ستون‌های اضافی از دیتاست حذف شده و فقط input و target در آن باقی می‌ماند.

برای پیش‌پردازش بر روی داده‌های ارزیاب فقط متن inputs با استفاده از تابع `clean_text()` تمیز و یک دست می‌شود.

داده‌های آموزش قبل از پیش‌پردازش به فرم زیر هستند که برای مثال یک نمونه درزیر آورده شده است:

```
Determine whether the following statement or statements are plausible or implausible:
Statement: Trevor Bauer swung to protect the runner in the World Series
Plausible/implausible?
['plausible']
```

و پس از اعمال پردازش به فرم زیر درمی‌آیند

```
Is the following sentence plausible? Trevor Bauer swung to protect the runner in
the World Series.
yes
```

Load Model

مدل و tokenizer را همانند سوال یک از huggingface لود می‌کنیم به فرم زیر:

```
tokenizer = T5Tokenizer.from_pretrained("google/flan-t5-large")
tokenizer.add_special_tokens({'pad_token': '[PAD]'})
tokenizer.padding_side = "right"

model = T5ForConditionalGeneration.from_pretrained("google/flan-t5-large",
device_map="auto", use_auth_token = mytokenhug)
```

اگر به عنوان مثال به مدل ورودی زیر را بدهیم.

```
input_text = "translate English to French: How old are you?"
input_ids = tokenizer(input_text, return_tensors="pt").input_ids.to("cuda")

outputs = model.generate(input_ids)
print(tokenizer.decode(outputs[0]))
```

خروجی مدل به فرم زیر است:

```
<unk> quelle âge avez-vous?</s>
```

حال که مدل و دیتاست آماده شد می‌توانیم Instruction tuning را انجام دهیم.

Answer Only

پرامپت ساخته در این مرحله برای دادن به مدل به فرم زیر است:

```
prompt = f"""
# Answer the following yes/no question with only yes or no.
{input}
"""
```

در ادامه یک نمونه از پرامپ ساخته شده برای این حالت به همراه خروجی مدل و جواب اصلی موجود در دیتاست آورده شده است:

Example 2

INPUT PROMPT:

Answer the following yes/no question with only yes or no.
Is the following sentence plausible? Aaron Nola converted the first down.

BASELINE HUMAN SUMMARY:
no

MODEL GENERATION – WITHOUT PROMPT ENGINEERING:
no

پرامپ ساخته شده دقیقا از متن مقاله آورده شده است بدین منظور است که به مدل بگوید تسکی که انجام می‌دهی چیست و به صورت yes/no پاسخ بده
مدل در این حالت تنها به yes/no اکتفا نمی‌کند و مقادیر متفاوتی را تولید می‌کند به منظور کنترل این مسئله راهکارهای مختلفی وجود دارد، به مدل گفته شد تنها یک token تولید کن و یا پس از آن‌که مدل خروجی داد یک مرحله پس‌پردازش نیاز است که مثلا اگه جمله‌ای گفته که در آن yes/no وجود دارد تنها این نتیجه ثبت شود.
تابعی که پس‌پردازش را انجام می‌دهد به فرم زیر است:

```
def find_last_yes_or_no(s):
    matches = re.findall(r'\b(?:yes|no)\b', s, flags=re.IGNORECASE)
    if matches:
        return matches[-1]
    else:
        return random.choice(["yes", "no"])
res = []
for sgen in generated_text:
  res.append(find_last_yes_or_no(sgen))
```

این تابع سعی می‌کند تمام yes و noهای موجود در خروجی را استخراج کرده و آخرین yes یا no را به عنوان خروجی برگرداند.
همچنین اگر مدل yes/no تولید نکرده بود به طور تصادفی در خروجی yes/no قرار می‌گیرد.
در فایل zeroshot.csv که ضمیمه شده است هم خروجی مدل و هم نتیجه پس‌پردازش آورده شده است.
توجه شود که برای ارزیابی مدل باید خروجی categorical به فرم عددی در بیاید به همین منظور یک mapping نیز به همین علت ایجاد شده است.

ارزیابی این روش به فرم زیر است:

جدول10 نتایج **answer only**

	precision	recall	f1-score	support
0	0.57	0.87	0.68	135
1	0.58	0.22	0.32	115
accuracy			0.57	250
macro avg	0.57	0.54	0.50	250
weighted avg	0.57	0.57	0.52	250

معیار مهمی که در طبقه‌بندی مورد استفاده قرار می‌گیرد f1-score است که در اینجا مقدار ۵۷٪ به دست آمده است. با توجه به اینکه مدل آموزش ندیده است و صرفا از مدل خواسته شده که به سوالات به این فرم پاسخ دهد به نظر نتایج بهتر از نتایج تصادفی است و تا حدی قابل قبول است.

Three Shot

در این حالت پرامپتی که به مدل داده می‌شود بدین صورت است که سه نمونه جمله و پاسخ به مدل نشان داده می‌شود و سپس از مدل خواسته می‌شود که پاسخ دهد. انتظار می‌رود آموزش اندکی در این رابطه اتفاق بیفتد و نتایج بهتر شود.
پرامپت ساخته در این مرحله برای دادن به مدل به فرم زیر است:

```
# Q: Answer the following yes/no question.
Is the following sentence plausible? Jarvis Landry gained five yards.
# A: yes

# Q: Answer the following yes/no question.
Is the following sentence plausible? Sam Darnold struck out.
# A: no

# Q: Answer the following yes/no question.
Is the following sentence plausible? Delon Wright took a left footed shot.
# A: no

# Q: Answer the following yes/no question.
Is the following sentence plausible? Mike Trout hit a walkoff homer.
# A:
```

نتایج مدل:

```
--------------------------------------------------------------------------------
BASELINE HUMAN TARGET:
yes

--------------------------------------------------------------------------------
MODEL GENERATION - 3 SHOT:
yes
```

فرمت این پرامپت نیز همانند آنچه در مقاله گفته شده است قرار دادەشد.
بدین صورت که سوال یا Q و جواب به صورت A نشانه‌گذاری می‌شود و در ادامه سوال خواسته شده آورده می‌شود.
در این مرحله نیز ما پس‌پردازش بر روی خروجی مدل را داریم هر چند که در این حالت مدل با دیدن نمونه‌ها یادمی‌گیرد که فقط yes/no تولید کند و نیازی به پس‌پردازش نیست. مابقی مراحل نیز همانند قسمت قبل صورت گرفت.

ارزیابی این روش به فرم زیر است:

جدول 11 نتایج 3shot

	precision	recall	f1-score	support
0	0.57	0.79	0.66	135
1	0.56	0.30	0.39	115
accuracy			0.57	250
macro avg	0.56	0.55	0.53	250
weighted avg	0.56	0.57	0.54	250

می‌بینیم که در این حالت افزایش دقتی نداشته‌ایم و مانند حالت zeroshot عمل شده است. تنها عملکردی که با اضافه کردن این سه نمونه ایجاد شد آن بود که مدل تنها yes/no پاسخ می‌داد اما در فهم درمورد خروجی نقشی نداشته این نمونه‌ها.
نتایج خروجی مدل و جملات در فایل threeshot.csv ضمیمه شده است.
توجه داشته باشید که پرامپت‌های نمونه ثابت است و با رندم قرار دادن آن نتایج بدتر می‌شد در نتیجه برای این حالت از سه جمله ثابت استفاده می‌شود.
مورد دیگری که در این حالت مشاهده می‌شود چون پرامپت ورودی دوتا نتیجه no داشته و یک مورد yes در این حالت مدل بیشتر تمایل دارد تا no پاسخ بدهد اما حالت zeroshot را مشاهده کنید مقدار recall برای yes برابر با ۸۷ درصد است اما در این حالت به ۷۹ درصد رسیده که نشان می‌دهد مدل در این حالت تمایل بیشتری به پاسخ no پیدا کرده است.

Chain of Thought (CoT)

در این حالت نیز همانند آن‌چه در مقاله گفته شده پرامپت را به فرم زیر تعریف می‌کنیم:

```
prompt = f"""
# Answer the following yes/no question by reasoning step-by-step.
{input}
"""
```

یک نمونه از خروجی مدل در زیر آورده شده است:

```
--------------------------------------------------------------------------------------------------------
Example  1
--------------------------------------------------------------------------------------------------------
INPUT PROMPT:

# Answer the following yes/no question by reasoning step-by-step.
Is the following sentence plausible? Ryan Tannehill hit a triple.

--------------------------------------------------------------------------------------------------------
BASELINE HUMAN SUMMARY:
no
--------------------------------------------------------------------------------------------------------
MODEL GENERATION - WITHOUT PROMPT ENGINEERING:
Ryan Tannehill is an American professional football linebacker for the Baltimore Ravens.
Therefore, the final answer is no.
--------------------------------------------------------------------------------------------------------
```

همان‌طور که مشاهده می‌شود خروجی مدل به فرم reasining در آمد که در مقاله ادعا می کند این فرمت از پاسخگویی سبب افزایش دقت مدل می‌شود، اما باید توجه داشت که خروجی به فرم yes/no نیست در نتیجه باید پس‌پردازش‌های لازم روی آن صورت گیر که همانند آن‌چه در قسمت zeroshot توضیح داده شد این پس‌پردازش صورت می‌گیرد.
نتایج خروجی مدل و پس‌پردازش در فایل zeroshotCOT.csv ضمیمه شده است.

نتایج حاصل از اعمال CoT:

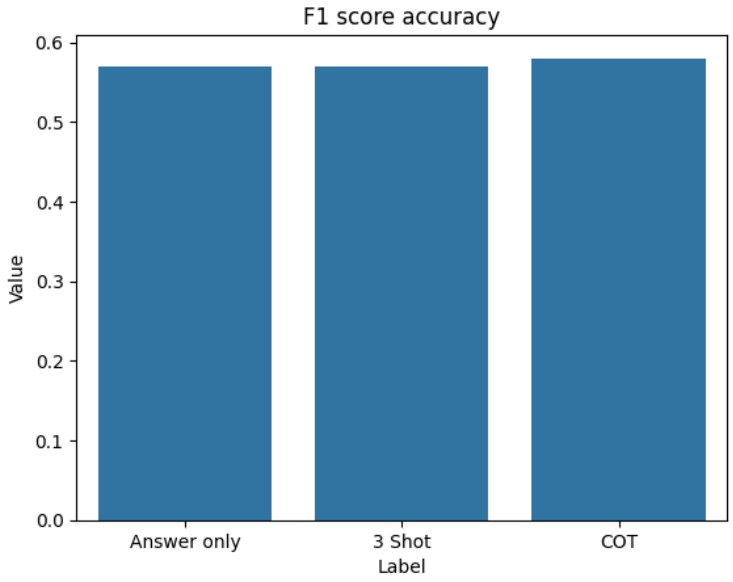
جدول 12 نتایج CoT prompting

	precision	recall	f1-score	support
0	0.61	0.59	0.60	135
1	0.54	0.57	0.55	115
accuracy			0.58	250
macro avg	0.57	0.58	0.57	250
weighted avg	0.58	0.58	0.58	250

همان‌طور که مشاهده می‌شود در این حالت نسبت به دو روش قبلی ما افزایش دقت داشته‌ایم هر چند خیلی محسوس نیست اما ادعای مقاله را ثابت می‌کند.
در این حالت بالانس بین جواب‌های yes/no خیلی بهتر شده و آن اختلاف ۳۰ درصدی که در دو مورد قبلی موجود بود در اینجا مشاهده نمی‌شود.
اضافه نمودن step-by-step reasoning by به انتهای پرامپت ورودی آن‌چنان که باید دقت مدل را افزایش نمی‌دهد چیزی که در مقاله ادعا کرده خیلی بیشتر از یک درصد افزایش دقت است. اما باید توجه داشت در مقاله ذکر کرده مدل را با دیتاست‌های CoT، finrtune کرده و افزایش دقت داشته نه به شکلی که ما استفاده می‌کنیم.
علت این امر را می‌توان چنین بیان کرد که مدل از پیش آموخته شده دانش حداکثری برای پاسخ به این سوالات را داشته و اضافه نمودن عبارت step-by-step دانشی را اضافه نمی‌کند. چرا که در گذار از zeroshot به 3shot نیز ما تغییری در افزایش قدرت مدل مشاهده نکردیم.

Results

نتایج به صورت نمودار به شرح زیر است:



شکل 7 نتایج هر یک از روش‌ها به تفکیک بر روی معیار F1

سوال ۳) APCot

این مقاله یک روش به نام Active Prompting with Chain-of-Thought را برای انتخاب مثال‌های آموزشی[1] مناسب برای الگودهی مدل‌های زبانی بزرگ ارائه می‌کند. در این پژوهش نشان داده می‌شود که اگر مراحل رسیدن به پاسخ نهایی به شکل زنجیره‌ای از تفکرات به عنوان چند نمونه سخت به مدل‌ها داده شود در عملکرد مدل موثر بوده و به نتایج بهتری منجر می‌شود.
تمرکز این پژوهش بر آن است که ابتدا هر داده آموزشی k بار به مدل داده می‌شود تا به ازای هر ورودی ما k پاسخ مختلف داشته باشیم سپس با معیارهایی که در ادامه توضیح داده خواهد شد میزان نایقینی مدل را برای هر پاسخ بررسی کرده n تا از داده‌های آموزشی که بیشترین نایقینی را دارند انتخاب می‌شوند، سپس در مرحله نهایی این n جمله منتخب که انگار از پیچیدگی بیشتری برخوردارند به روش زنجیره استدلال (CoT) توسط افراد متخصص الگودهی شده و این الگودهی‌ها به روش‌های استانداری که در دیگر پژوهش‌ها معرفی شده اعمال می‌گردد.
سپس از این مثال‌هایی که به روش CoT توضیح داده شده است برای نمونه‌های few-shot استفاده شده تا قدرت مدل در پاسخ دهی به سوالات را بالا ببرد. (از این نمونه‌ها برای الگودهی مدل زبانی بزرگ مثلاً (GPT-3) استفاده می‌شود)
نتایج این فرآیند نشان می‌دهد که APCot منجر به بهبود قابلیت استدلال مدل در حالت‌های صفرنمونه[2] و چندنمونه[3] در مجموعه داده‌های مختلف شده است.

در ادامه معیاری‌هایی که در این پژوهش استفاده شده تا میزان نایقینی پاسخ‌های مدل را به سوالات نشان می‌دهد را توضیح خواهیم داد.

Uncertainty Estimation

1. Disagreement
2. Entropy
3. Variance
4. Self-Confidence

ابتدا، ما از معیار عدم توافق[4] بین k پاسخ تولید شده $A = \{a_1, a_2, a_{3,\dots}, a_k\}$ برای اندازه‌گیری اطمینان استفاده می‌کنیم. عدم توافق به وسیله محاسبه تعداد پاسخ‌های یکتا در پیش‌بینی‌ها انجام می‌شود. اجرای این معیار ساده است. ابتدا، تعداد پاسخ‌های یکتا را با استفاده از یک عملیات مجموعه محاسبه می‌کنیم تا موارد تکراری حذف شوند، به دست آوردن موارد یکتا $A = \{a_1, a_2, a_{3,\dots}, a_h\}$. سپس عدم توافق با $u = \frac{h}{k}$ محاسبه می‌شود، که تعداد پاسخ‌های یکتا را نمایانگر می‌کند، یا به عبارتی دیگر برای هر سوال، تعداد پاسخ‌های یکتا محاسبه شده و نسبت آن به تعداد کل پاسخ‌ها بدست می‌آید. این روش به ما اطلاعاتی در مورد عدم اتفاق نظر در پاسخ‌های مدل فراهم می‌کند که درصورتی که مدل تعداد پاسخ‌های بیشتری را تولید کند در نتیجه ناهماهنگی بیشتری در آن و به طبع آن نایقینی بیشتری وجود دارد.

معیار entropy نیز می‌تواند به خوبی عدم قطعیت را نشان دهد، و در این‌جا بدین صورت عمل می‌کند که $p_\theta(a_j|q_i)$ احتمال تعداد تکرار جواب یکتا را در بین تمامی k جواب را نشان می‌دهد.

[1] Few-shot examples
[2] Zero shot
[3] Few shot
[4] Disagreement

اطمینان نیز می‌تواند توسط اندازه‌گیری آنتروپی مشخص شود که به صورت زیر محاسبه می‌شود:

$$u = \arg\max_i - \sum_{j=1}^{k} P_\theta(a_j|q_i) \ln P_\theta(a_j|q_i),$$

که $p_\theta(a_j|q_i)$ همان‌طور که پیش‌تر گفته شد، فراوانی یک پاسخ خاص پیش‌بینی‌شده در میان همه پیش‌بینی‌هاست. آنتروپی بزرگتر نشان‌دهنده عدم اطمینان بیشتر در سیستم است و آنتروپی کوچکتر نشان‌دهنده عدم اطمینان کمتر است. بنابراین، در استدلال‌های پیچیده، سوالاتی که دارای آنتروپی نسبتاً بزرگی هستند به عنوان کاندیداهای مورد نظر انتخاب خواهند شد.

معیار بعدی استفاده از واریانس است که به فرم زیر محاسبه می‌شود:

$$u = \arg\max_i \frac{\sum_{j=1}^{k}(a_j - \bar{a})^2}{k-1}\Big|_{q=q_i},$$

از انحراف معیار به عنوان نوعی از معیار عدم اطمینان را در نظر گرفته می‌شود که فرض می‌شود ممکن است برای پاسخ‌های عربی مناسب‌تر باشد. که $\bar{a}$ به فرم زیر تعریف می‌شود:

$$\bar{a} = \frac{1}{k}\sum_{j=1}^{k} a_j$$

مشاهده شده است که تغییرات زیادی در پاسخ‌های پیش‌بینی‌شده وجود دارد. برخی از پاسخ‌های پیش‌بینی‌شده اعداد کوچکی هستند (مثلاً ۱)، در حالی که برخی از آنها اعداد بزرگی هستند (مثلاً ۱۰۰۰۰).
برای کاهش مسأله اساسی اعداد بزرگ، پیشنهاد می‌شود که پیش‌بینی‌ها را با همهٔ اعداد مذکور در سوال نرمال‌سازی شود. به عنوان مثال، با داشتن یک سوال مانند "تعداد افراد x_1 نفر است. هر فرد x_2 سیب دارد. مجموع سیب‌ها چقدر است؟" و یک پاسخ پیش‌بینی شده $\hat{y}$ ، پس از نرمال‌سازی، به فرم $\frac{\hat{y}}{(|x_1|+|x_2|)}$ به‌دست می‌آید.

و در نهایت معیار آخر self-confidence است بدین صورت که می‌توان گفت استفاده از مدل‌های زبان بزرگ برای تخمین عدم اطمینان می‌تواند از طریق خود مدل‌های زبان انجام شود؛ به این معنا که از self-confidence استفاده شود. این معیار با استفاده از پرسش مستقلی که به صورت دستی ایجاد شده است به عنوان قالبT ، به‌دست می‌آید. به عنوان مثال، با پرسش q و پاسخ پیش‌بینی شده a ، یک جمله مانند "برای سوال q و پاسخ پیش‌بینی شده a، اطمینان خود را از پاسخ گزارش کنید. (الف) بسیار اطمینان دارم (ب) اطمینان دارم (ج) اطمینان ندارم (د) پاسخ اشتباه. سوالات با کمترین اطمینان را انتخاب می‌کنیم:

$$u = \arg\max_i (1 - \max_j P_\theta(a_j|q_i))$$
$$= \arg\min_i \max_j P_\theta(a_j|q_i),$$

که $p_\theta(a_j|q_i)$ یک متغیر گسسته از مجموعه {بسیار اطمینان دارم، اطمینان دارم، اطمینان ندارم، پاسخ اشتباه} است. مطالعه آزمایشی انجام شده و نشان داده شده است که معیارهای بر اساس disagreement، آنتروپی، و واریانس به ترتیب خوب عمل می‌کنند و به طور قابل‌توجهی بهتر از self-confidenc هستند. بنابراین، در این پژوهش، اصولاً از معیارهای disagreement و آنتروپی برای رویکرد خود استفاده شده که پیاده‌سازی آن‌ها نیز ساده‌تر است.

سوال ۴) self-consistency

این مقاله روشی به نام Self-Consistency را برای بهبود توانایی استدلال زنجیره‌ای[1] در مدل‌های زبانی بزرگ از پیش‌آموزش دیده پیشنهاد می‌کند.

استفاده اززنجیره استدلال در ترکیب با مدل‌های زبان بزرگ پیش‌آموزش دیده، نتایج امیدبخشی در وظایف استدلال پیچیده به دست آورده است. در این مقاله، یک راهبرد جدید برای تولید، به نام خودسازگاری[2]، جایگزینی برای تولید خروجی به روش حریصانه[3] در CoT ارائه می‌شود. این راهبرد ابتدا یک مجموعه متنوع از مسیرهای استدلال را نمونه‌برداری می‌کند به جای انتخاب تنها مسیر حریصانه، مطمئن‌ترین پاسخ را با حذف مسیرهای استدلالی که نمونه‌برداری شده‌اند، انتخاب می‌کند. و نشان می‌دهد که چگونه این ایده می‌تواند عملکرد استدلال زنجیره‌ای را بهبود بخشد.

به طور خلاصه ایده این مقاله آن است که می‌توان از روشی انسان‌ها برای استدلال استفاده می‌کنند الهام گرفت به گونه‌ای که ما زمانی یک پاسخ را با منطق خود قبول می‌کنیم که با روش‌های استدلالی مختلف به تعداد بیشتر به آن رسیده باشیم، این روش با به کار گیری این ایده سعی دارد تا به یک پاسخ درست با استفاده از شیوه‌های استدلالی متفاوت برسد.

در ابتدا مدل را وادار می‌کند تا زنجیره‌های استدلال متفاوتی را برای یک مسئله تولید کند. این کار را بدین شکل انجام می‌دهد که به جای یک پاسخ که از مدل به روش حریصانه به دست می‌آید از رمزگذار مدل نمونه‌گیری کند تا به چندین استدلال مختلف دست یابد. سپس سازگاری درونی این زنجیره‌ها را ارزیابی می‌کند. اگر زنجیره‌ها ناسازگار باشند، جریمه‌ای را اعمال می‌کند تا مدل یاد بگیرد زنجیره‌های سازگارتری تولید کند.

این روش نسبت به Chain of Thought معمولی بهتر عمل می‌کند چرا که مدل را مجبور می‌کند مسیرهای استدلالی متفاوتی را در نظر بگیرد و سازگاری درونی استدلال‌ها را تضمین می‌کند و در نهایت نیز از تولید زنجیره‌های تکراری یا نامربوط جلوگیری می‌کند.

[1] Chain of Thought
[2] self-consistency
[3] greedy

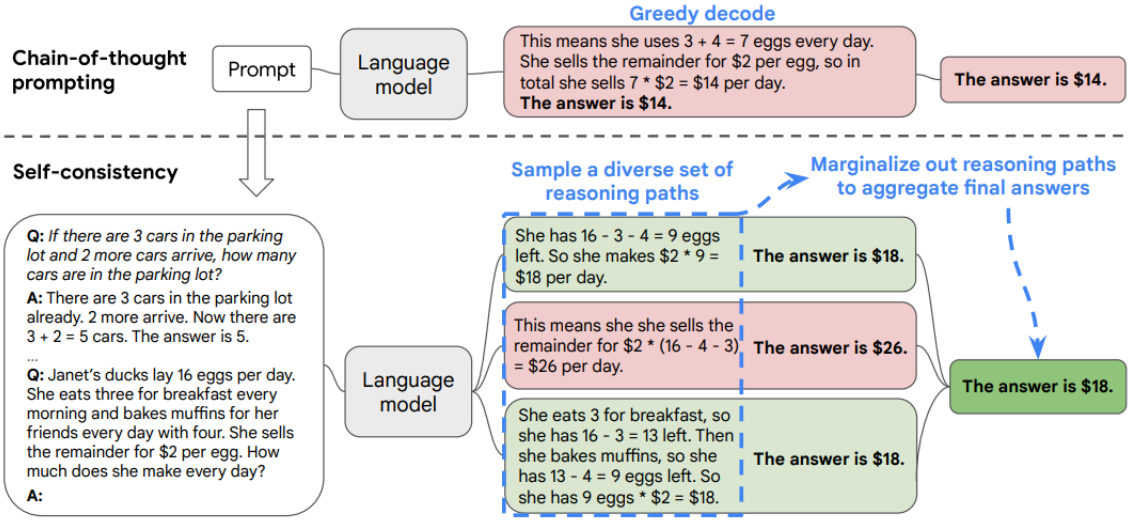


Figure 1: The self-consistency method contains three steps: (1) prompt a language model using chain-of-thought (CoT) prompting; (2) replace the "greedy decode" in CoT prompting by sampling from the language model's decoder to generate a diverse set of reasoning paths; and (3) marginalize out the reasoning paths and aggregate by choosing the most consistent answer in the final answer set.

شکل 8 **self consistency method**

همان‌طور که در شکل بالا نشان داده شده است ابتدا زنجیره‌های استدلال مختلف برای یک مسئله تولید می‌شود. سپس سازگاری درونی این زنجیره‌ها بررسی شده و امتیاز سازگاری محاسبه می‌شود. این امتیاز به عنوان بازخورد برای آموزش مدل استفاده می‌شود تا در تولید زنجیره‌های بعدی سازگارتر عمل کند. در روش‌های پیشین از روش حریصانه برای آوردن استدلال استفاده می‌کند اما در اینجا از یک روش که مخلوطی از دو روش optimal text generation with a fixed answer و non-ended text generation است به جهت رسیدن به جواب بهینه استفاده می‌کند.

این روش که الگوریتم زنجیره‌های استدلال متفاوتی را با تغییر دادن پارامتر تصادفی‌سازی مانند tempreture ایجاد می‌کند.همچنین در این پژوهش نشان داده شده است که این زنجیره‌های متفاوت اغلب ناسازگار هستند و خودسازگاری می‌تواند این مشکل را برطرف کند. همچنین بیان شده با افزایش تعداد زنجیره‌ها، تاثیر خودسازگاری بیشتر می‌شود.
در بخش SELF-CONSISTENCY OVER DIVERSE REASONING PATHS در مورد چگونگی تولید پاسخ‌ها توسط مدل توضیحاتی ارائه شده است. هر پاسخ تولید شده توسط مدل را با a_i نمایش می‌دهد که از یک مجموعه ثابت A ($A \in a_i$) استخراج می‌شود. و یک متغیر مخفی اضافی به نام r_i معرفی می‌شود که یک دنباله از توکن‌هاست و مسیر استدلال را در خروجی i ایجاد می‌کند. پس از تولید a_i ،r_i را که به معنای ایجاد مسیر استدلال r_i به پاسخ نهایی a_i است، انجام می‌دهد. مثالی از این موضوع را در Output 3 از شکل ۸ می‌توان ذکر کرد: جملات اولیه "او صبحانه 3 عدد می‌خورد ... بنابراین او 9 تخم‌مرغ دارد ضرب‌در 2 دلار = 18 دلار r_i " را تشکیل می‌دهد، در حالی که پاسخ 18 از جمله‌ی آخر "پاسخ 18 دلار است" به عنوان a_i معنی می‌شود.
سپس، بعد از نمونه‌برداری از چندین (a_i ،r_i) از decoder مدل، خود-سازگاری یک marginal روی r_i انجام می‌دهد. به عبارت دیگر، انتخاب پاسخ "سازگارتر" بین مجموعه پاسخ‌های نهایی به فرم زیر صورت می‌گیرد:

$$\arg\max_a \sum_{i=1}^{m} \mathbb{1}(\mathbf{a}_i = a)$$

وجود استدلال‌های مختلف و در نهایت انتخاب پاسخی که با تعداد بیشتری استدلال می‌توان به آن رسید سبب می‌شود روش CoT بهتر از روش‌های پیشین موجود عمل کند.

منابع

سوال یک

https://medium.com/@aditya.addy.bahl/democratizing-ai-fine-tuning-llama-2-a-step-by-step-instructional-guide-19d3dad84202
https://www.kaggle.com/code/mahimairaja/fine-tuning-llama-2-tweet-summarization
https://github.com/sarahaman/CIS6930_TweetSum_Summarization/blob/main/model_fine tuning/bart_model.ipynb

سوال دو

https://github.com/haotian-liu/LLaVA
https://medium.com/@aditya.addy.bahl/democratizing-ai-fine-tuning-llama-2-a-step-by-step-instructional-guide-19d3dad84202
https://www.kaggle.com/code/mahimairaja/fine-tuning-llama-2-tweet-summarization
https://github.com/sarahaman/CIS6930_TweetSum_Summarization/blob/main/model_fine tuning/bart_model.ipynb
https://huggingface.co/google/flan-t5-large