



دانشگاه صنعتی شریف

دانشکده مهندسی کامپیوتر

درس پردازش زبان های طبیعی

**عنوان:**

**سیستم پرسش و پاسخ پزشکی**

اعضای گروه:

سید محمدرضا حسینی

فرحان سراوند

وحیدالدین مقیمی

استاد: احسان الدین عسگری

## فهرست مطالب

1) بازیابی متون مرتبط

2) تولید پاسخ

## ۱. بازیابی متون مرتبط

برای ساخت این سیستم در ابتدا نیاز به پیدا کردن دیتاست مناسب برای انجام این تمرین بود. Medmcqa دیتاست استفاده شده است که یک دیتاست برای پاسخ گویی به سوالات پزشکی میباشد. پس از دانلود دیتاست، برای اینکه بتوانیم جواب مورد نظرمان را بیابیم، میبایست بتوانیم از یک تکنیک مناسب برای بازیابی متون استفاده کنیم. تکنیک استفاده شده بدین صورت است که در ابتدا با استفاده از TF-IDF، کوئری و سوال فرد و جواب های موجود را تبدیل میکنیم سپس با استفاده از Cosine similarity میزان شباهت کوئری و جواب ها را میسنجیم و بر اساس بالاترین شباهت جواب ها را مرتب میکنیم

## 2. تولید پاسخ

هدف از این تسک پیاده سازی یک RAG(Retrieval Augment Generation) است که در آن با استفاده از داده

های پزشکی به توان به سوالات مطرح شده پاسخ مناسبی داد.

پس از انتخاب دیتاست و پیاده سازی تکنیک بازیابی متون به سراغ تسک اصلی میرویم. در ابتدا از مدل پایه بدون هیچ

فاین تیونینگ استفاده میکنیم.

```
query = "Which vitamin is supplied from only animal source?"

data = load_data(json_file)
filtered_data, texts = preprocess_texts(data)

tfidf_matrix = compute_tfidf(texts, query)
similarities = find_similar_texts(tfidf_matrix)
related_texts = get_related_texts(filtered_data, similarities)

model_name = "t5-small"
model = T5ForConditionalGeneration.from_pretrained(model_name)
tokenizer = T5Tokenizer.from_pretrained(model_name)

input_text = create_input(query, related_texts)
print(f"Input text for the model: {input_text}")

answer = generate_answer(
    model, tokenizer, input_text,
    max_length=150,
    num_beams=10,
    early_stopping=False,
    temperature=0.7,
    top_k=50,
    top_p=0.9,
    repetition_penalty=2.0,
    no_repeat_ngram_size=3,
    length_penalty=3.0,
    do_sample=True
)

print(f"Generated answer: {answer}")
```

پرسش و پاسخ به صورت زیر میباشد:

```
Input text for the model: پرسش: Which vitamin is supplied from only animal source? زمینه: Ans.
(c) Vitamin B12 Ref: Harrison's 19th ed. P 640* Vitamin B12 (Cobalamin) is synthesized solely
by microorganisms.* In humans, the only source for humans is food of animal origin, e.g.,
meat, fish, and dairy products.* Vegetables, fruits, and other foods of nonanimal origin
doesn't contain Vitamin B12 .* Daily requirements of vitamin Bp is about 1-3 pg. Body stores
are of the order of 2-3 mg, sufficient for 3-4 years if supplies are completely cut off.
Vitamin D is not strictly a vitamin since it can be synthesized in the skin, and under most
conditions, this is the major source of the Vitamin D. Ref : Biochemistry by U. Satyanarayana
3rd edition Pgno : 123 Ans. (a) Animal SourceRef: Harrison / 640Only source of vitamin B12 for
humans is food of animal origin, e.g., meat, fish, and dairy products.Vegetables, fruits, and
other foods of non-animal origin are free from cobalamin unless they are contaminated by
bacteria. Cobalamin is synthesized solely by microorganisms.
Generated answer: Biochemistry by U. Satyanarayana 3rd edition Pgno : 123 Ans. (a) Animal
SourceRef: Harrison / 640Only source of vitamin B12 for humans is food of animal origin, e.g.,
meat, fish, and dairy products.* Vegetables, fruits, and other foods of non-animal origin
doesn't contain Vitamin B12.
```

سپس به سراغ فاین تیون کردن مدل T5 میرویم. برای اینکار تعداد داده های مختلفی آزمایش شد و در نهایت ۲۰۰۰ پرسش و پاسخ اول موجود در دیتاست برای فاین تیون کردن مدل استفاده شد تا بتوان اینکار را در یک زمان مناسب انجام داد. جواب ها در یک لیست و سوال ها در یک لیست دیگر ذخیره میشود. سوال ها به عنوان ورودی و جواب به عنوان خروجی مطلوب که مدل باید به آن دست پیدا کند فرض میشود. برای توکنایز کردن داده ها نیز از توکنایزر مدل T5 استفاده شده تا با آن همخوانی داشته باشد. این مدل ۶ ایپاک فاین تیون شده و Loss های دریافت شده از آموزش و ولیدیشن نشان دهنده موفق بودن فاین تیونینگ میباشد.

```
json_file = 'train.json'
finetune_data = load_finetune_data(json_file, limit=2000)
inputs, targets = preprocess_finetune_data(finetune_data)

train_inputs, eval_inputs, train_targets, eval_targets = train_test_split(inputs, targets, test_size=0.1)

model_name = "t5-small"
tokenizer = T5Tokenizer.from_pretrained(model_name)
model = T5ForConditionalGeneration.from_pretrained(model_name)

train_encodings = tokenize_data(tokenizer, train_inputs, train_targets)
eval_encodings = tokenize_data(tokenizer, eval_inputs, eval_targets)

train_dataset = Dataset.from_dict(train_encodings)
eval_dataset = Dataset.from_dict(eval_encodings)

finetune_model(model, tokenizer, train_dataset, eval_dataset)

model.save_pretrained("./finetuned_model_6e")
tokenizer.save_pretrained("./finetuned_model_6e")
```

Epoch	Training Loss	Validation Loss
1	4.701400	4.509769
2	4.704900	4.277831
3	4.388300	4.167476
4	4.424300	4.105363
5	4.394600	4.077777
6	4.090000	4.070305

پس از فاین تیون کردن مدل از این مدل برای تولید پاسخ با توجه به دیتا استفاده میکنیم. در ابتدا مطابق روش گفته شده پرسش به همراه جواب ها با استفاده از TF\_IDF تبدیل میشوند سپس با استفاده از cosine\_similarity مرتبط ترین جواب ها پیدا میشوند. با دادن مرتبط ترین جواب ها و کوئری به مدل فاین تیون شده، جواب به آن سوال تولید میشود.

برای تولید پاسخ تعدادی پارامتر ست شده است که به صورت زیر میباشد:

`max_length = 150` که حداکثر میزان توکن تولیدی را نشان میدهد

`num_beams = 5` که تعداد پرتوهای استفاده شده در beam search را نمایش میدهد. ست کردن درست این پارامتر سبب میشود که تعداد جواب های تولیدی بیشتر شود و بهترین آن ها برای خروجی استفاده شود.

`early_stopping` : ست کردن این پارامتر سبب میشود که در صورتی که به اندازه تعداد پرتو جواب ها را تولید کردیم، عملیات متوقف شود

`temprature=0.7` با ست کردن این پارامتر میزان قطعی بودن یا تخیلی بودن جواب را تعیین میکنیم. هر چه این عدد کم تر و نزدیک به صفر باشد جواب با توجه به منابع موجود در پایگاه دانش تولید میشود و هرچه بیشتر باشد مدل نیز سعی میکند جواب را طبق دانسته ها تغییر دهد.

`top_k=50` تعداد کاندیداهای برتر که باید از آن نمونه برداری شود را نشان میدهد.

`top_p = 0.9` نمونه گیری هسته ای که مجموع احتمالات را تا یک آستانه خاص شامل میشود.

repetition\_penalty=0.2 این پارامتر مقدار جریمه ای که به ازای تکرار کلمات اتفاق میفتد را نشان میدهد. این

جریمه از تکرار کردن کلمات توسط مدل جلوگیری میکند.

سپس پس از فاین تیون با استفاده از روش گفته شده و استفاده از TF\_IDF و Cosine Similiarity جواب های پیدا

شده به مدل داده میشود و مدل جواب نهایی را تولید میکند. جواب RAG پیاده سازی شده با مدل فاین تیون شده به

صورت زیر است:

```
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.

Input text for the model: پرسش: Which vitamin is supplied from only animal source? زمینه: Ans. (c) Vitamin B12 Ref: Harrison's 19th ed. P 640* Vitamin B12 (Cobalamin) is synthesized solely by microorganisms.* In humans, the only source for humans is food of animal origin, e.g., meat, fish, and dairy products.* Vegetables, fruits, and other foods of nonanimal origin doesn't contain Vitamin B12 .* Daily requirements of vitamin Bp is about 1-3 pg. Body stores are of the order of 2-3 mg, sufficient for 3-4 years if supplies are completely cut off. Vitamin D is not strictly a vitamin since it can be synthesized in the skin, and under most conditions, this is the major source of the Vitamin D. Ref : Biochemistry by U. Satyanarayana 3rd edition Pgno : 123 Ans. (a) Animal SourceRef: Harrison / 640Only source of vitamin B12 for humans is food of animal origin, e.g., meat, fish, and dairy products.Vegetables, fruits, and other foods of non-animal origin are free from cobalamin unless they are contaminated by bacteria. Cobalamin is synthesized solely by microorganisms.
Generated answer: Biochemistry by U. Satyanarayana 3rd edition Pgno : 123 Ans. (a) Animal SourceRef: Harrison / 640Only source of vitamin B12 for humans is food of animal origin, e.g., meat, fish, and dairy products.* Vegetables, fruits, and other foods of non-animal origin doesn't contain Vitamin B12.
```

برای ارزیابی مدل از بخشی از دیتاست استفاده شده که برای فاین تیون مدل استفاده نشده است تا مدل آن را ندیده باشد

و بر اساس آن تست ها انجام شود.

```
# Calculate BLEU, ROUGE, and METEOR scores
base_bleu_score = bleu.compute(predictions=base_predictions, references=[[ref] for ref in base_references])
base_rouge_score = rouge.compute(predictions=base_predictions, references=base_references)
base_meteor_score = meteor.compute(predictions=base_predictions, references=base_references)

finetuned_bleu_score = bleu.compute(predictions=finetuned_predictions, references=[[ref] for ref in finetuned_references])
finetuned_rouge_score = rouge.compute(predictions=finetuned_predictions, references=finetuned_references)
finetuned_meteor_score = meteor.compute(predictions=finetuned_predictions, references=finetuned_references)

print("Base Model BLEU Score:", base_bleu_score)
print("Base Model ROUGE Score:", base_rouge_score)
print("Base Model METEOR Score:", base_meteor_score)

print("Fine-tuned Model BLEU Score:", finetuned_bleu_score)
print("Fine-tuned Model ROUGE Score:", finetuned_rouge_score)
print("Fine-tuned Model METEOR Score:", finetuned_meteor_score)

# Calculate precision, recall, and accuracy
def calculate_precision_recall_accuracy(predictions, references):
    precision = np.mean([1 if pred in ref else 0 for pred, ref in zip(predictions, references)])
    recall = np.mean([1 if ref in pred else 0 for pred, ref in zip(predictions, references)])
    accuracy = np.mean([1 if pred == ref else 0 for pred, ref in zip(predictions, references)])
    return precision, recall, accuracy

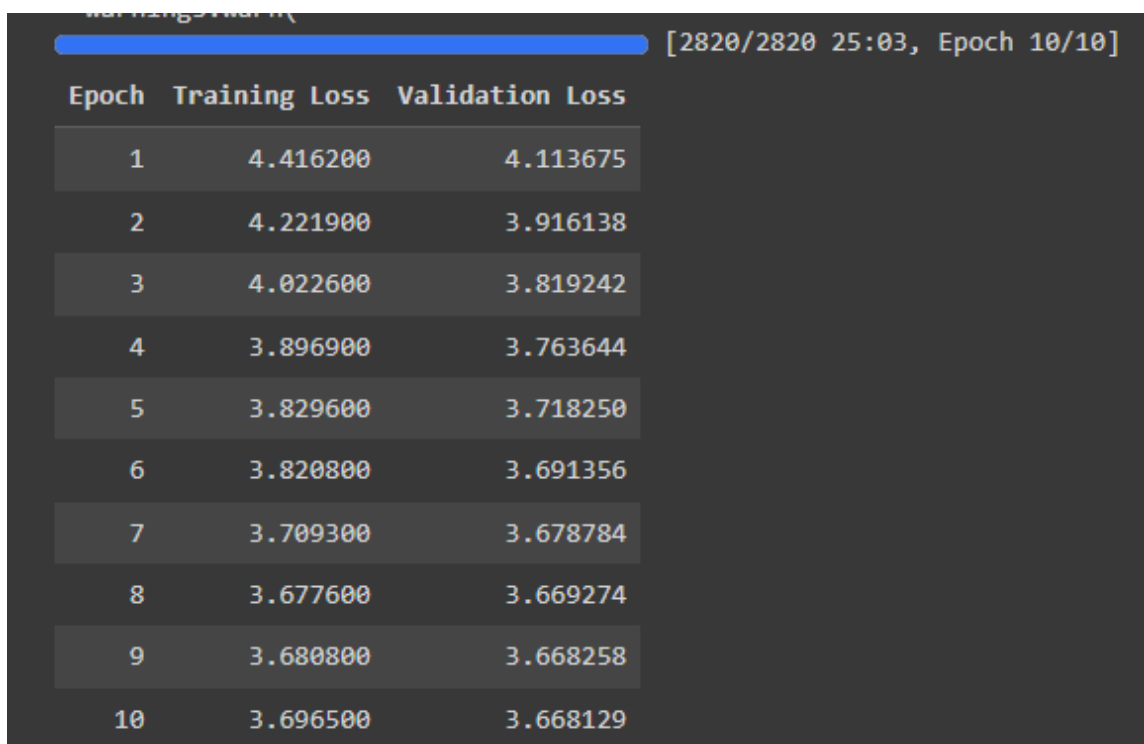
# Base model precision, recall, accuracy
base_precision, base_recall, base_accuracy = calculate_precision_recall_accuracy(base_predictions, base_references)
print("Base Model Precision:", base_precision)
print("Base Model Recall:", base_recall)
print("Base Model Accuracy:", base_accuracy)

# Fine-tuned model precision, recall, accuracy
finetuned_precision, finetuned_recall, finetuned_accuracy = calculate_precision_recall_accuracy(finetuned_predictions, finetuned_references)
print("Fine-tuned Model Precision:", finetuned_precision)
print("Fine-tuned Model Recall:", finetuned_recall)
print("Fine-tuned Model Accuracy:", finetuned_accuracy)
```

در هنگام فاین تیونینگ پارامترهای مختلفی امتحان شد تا عملکرد مدل بهبود یابد ولی تنها حالتی که بهترین جواب را از فاین تیون کردن مدل دریافت کردیم در حالت فاین تیون توضیح داده شده است. که همانطور که مشخص است در معیار BLUE نسبت به حالت پایه بهبود داشته ایم. دلیل اینکه در سایر معیارها دچار افت شده ایم به دلیل وسیع بودن داده استفاده شده و حجم اطلاعات زیادی است که سعی در آموزش مدل داریم.

```
Base Model BLEU Score: {'score': 17.629505379361227, 'counts': [1444, 1194, 1845, 922], 'totals': [1924, 1882, 1840, 1798], 'precisions': [75.05197505197505, 63.443145589798085, 56.793478260869565, 48.484848484848484], 'recalls': [63.443145589798085, 75.05197505197505, 48.484848484848484, 56.793478260869565], 'fmeasure': 0.4953541960164611}, mid-Score(precision=0.7389642138212178, recall=0.4953541960164611), fmeasure=0.4953541960164611)
Base Model ROUGE Score: {'rouge1': AggregateScore(low=Score(precision=0.6584495807825766, recall=0.4644801900567144, fmeasure=0.4953541960164611), mid=Score(precision=0.7389642138212178, recall=0.4953541960164611), fmeasure=0.4953541960164611), high=Score(precision=0.7389642138212178, recall=0.4953541960164611), fmeasure=0.4953541960164611)}
Base Model METEOR Score: {'meteor': 0.4999402446590186}
Fine-tuned Model BLEU Score: {'score': 18.91291897774872, 'counts': [1478, 962, 723, 568], 'totals': [2894, 2852, 2810, 2768], 'precisions': [51.07118175535591, 33.73071528751753, 25.72953736641229, 18.91291897774872], 'recalls': [33.73071528751753, 25.72953736641229, 18.91291897774872, 14.780000000000001], 'fmeasure': 0.3854562880634302}, mid-Score(precision=0.5566060495765959, recall=0.3854562880634302), fmeasure=0.3854562880634302)
Fine-tuned Model ROUGE Score: {'rouge1': AggregateScore(low=Score(precision=0.4594775631015364, recall=0.4378280755172668, fmeasure=0.3854562880634302), mid=Score(precision=0.5566060495765959, recall=0.3854562880634302), fmeasure=0.3854562880634302), high=Score(precision=0.5566060495765959, recall=0.3854562880634302), fmeasure=0.3854562880634302)}
Fine-tuned Model METEOR Score: {'meteor': 0.38879519825876346}
Base Model Precision: 0.07142857142857142
Base Model Recall: 0.023809523809523808
Base Model Accuracy: 0.0
Fine-tuned Model Precision: 0.047619047619047616
Fine-tuned Model Recall: 0.023809523809523808
Fine-tuned Model Accuracy: 0.0
```

به عنوان مثال حالت زیر حالتی بوده که مقدار Training loss , evaluation Loss بیشتر کاهش پیدا کرده و به نظر در فاین تیون کردن مدل موفق تر بوده ایم.



Epoch	Training Loss	Validation Loss
1	4.416200	4.113675
2	4.221900	3.916138
3	4.022600	3.819242
4	3.896900	3.763644
5	3.829600	3.718250
6	3.820800	3.691356
7	3.709300	3.678784
8	3.677600	3.669274
9	3.680800	3.668258
10	3.696500	3.668129

در معیار BLUE این مدل به شدت نسبت به مدل گفته شده در بالا عمل میکند ولی توانسته ROUGE و METEOR بالاتری نسبت به مدل بالا داشته باشد ولی عملکرد آن نسبت به مدل پایه ضعیف تر است.



```
Base Model BLEU Score: {'score': 19.67316506107492, 'counts': [1536, 1307, 1159, 1031], 'totals': [1953, 1911, 1869, 1827], 'precisions': [78.64823348694317, 68.39351125065411, 62.011771000535, 57.44444444444444], 'recalls': [78.64823348694317, 68.39351125065411, 62.011771000535, 57.44444444444444], 'fmeasure': 68.39351125065411, 'mid-Score': 62.011771000535, 'agg-Score': 19.67316506107492}
Base Model ROUGE Score: {'rouge1': AggregateScore(low=Score(precision=0.6884174621302949, recall=0.4832937496976874, fmeasure=0.5209093334907381), mid=Score(precision=0.769314563502743, recall=0.5209093334907381, fmeasure=0.5209093334907381), high=Score(precision=0.84823348694317, recall=0.6839351125065411, fmeasure=0.769314563502743)), 'rouge2': AggregateScore(low=Score(precision=0.5957655865918976, recall=0.3719458162205621, fmeasure=0.4133342541575794), mid=Score(precision=0.6699583359523383, recall=0.5209093334907381, fmeasure=0.5209093334907381), high=Score(precision=0.769314563502743, recall=0.6839351125065411, fmeasure=0.769314563502743)), 'rougeL': AggregateScore(low=Score(precision=0.5957655865918976, recall=0.3719458162205621, fmeasure=0.4133342541575794), mid=Score(precision=0.6699583359523383, recall=0.5209093334907381, fmeasure=0.5209093334907381), high=Score(precision=0.769314563502743, recall=0.6839351125065411, fmeasure=0.769314563502743))}
Base Model METEOR Score: {'meteor': 0.5207160868888759}
Base Model Precision: 0.0
Base Model Recall: 0.047619047619047616
Base Model Accuracy: 0.0
Fine-tuned Model Precision: 0.0
Fine-tuned Model Recall: 0.0
Fine-tuned Model Accuracy: 0.0
```

سعی کردیم تعداد داده ها و ایپاک ها را نیز تغییر داده و پس از لرنینگ ریت و وارمینگ استپ را نیز تغییر دهیم ولی باز نتایج از مدل پایه ضعیف تر عمل کرده اند :

[5630/5630 27:01, Epoch 10/10]		
Epoch	Training Loss	Validation Loss
1	4.393800	4.140195
2	4.302400	4.024040
3	4.160300	3.965619
4	4.292700	3.933632
5	4.034300	3.904641
6	4.253400	3.886363
7	4.084500	3.879431
8	3.979000	3.873139
9	4.072300	3.871222
10	4.150400	3.870848

```
Base Model BLEU Score: {'score': 18.007909925233037, 'counts': [1454, 1154, 974, 844], 'totals': [1996, 1953, 1910, 1867], 'precisions': [72.84569138276554, 59.08858166922683, 50.994764397901, 44.44444444444444], 'recalls': [72.84569138276554, 59.08858166922683, 50.994764397901, 44.44444444444444], 'fmeasure': 59.08858166922683, 'mid-Score': 50.994764397901, 'agg-Score': 18.007909925233037}
Base Model ROUGE Score: {'rouge1': AggregateScore(low=Score(precision=0.5909026843816844, recall=0.5088833903235308, fmeasure=0.4818410873371489), mid=Score(precision=0.6665892417414951, recall=0.5909026843816844, fmeasure=0.5909026843816844), high=Score(precision=0.769314563502743, recall=0.6839351125065411, fmeasure=0.769314563502743)), 'rouge2': AggregateScore(low=Score(precision=0.35990642002699, recall=0.44034935367307093, fmeasure=0.30950314124485423), mid=Score(precision=0.443641636450647, recall=0.5209093334907381, fmeasure=0.5209093334907381), high=Score(precision=0.769314563502743, recall=0.6839351125065411, fmeasure=0.769314563502743)), 'rougeL': AggregateScore(low=Score(precision=0.35990642002699, recall=0.44034935367307093, fmeasure=0.30950314124485423), mid=Score(precision=0.443641636450647, recall=0.5209093334907381, fmeasure=0.5209093334907381), high=Score(precision=0.769314563502743, recall=0.6839351125065411, fmeasure=0.769314563502743))}
Base Model METEOR Score: {'meteor': 0.3465162769077006}
Base Model Precision: 0.0
Base Model Recall: 0.18604651162790697
Base Model Accuracy: 0.0
Fine-tuned Model Precision: 0.0
Fine-tuned Model Recall: 0.09302325581395349
Fine-tuned Model Accuracy: 0.0
```