論文要約

LLM関連

Assisting in Writing Wikipedia-like Articles From Scratch with Large Language Models 大規模言語モデルを用いたウィキペディアのような記事のゼロからの執筆支援 2024

概要

LLMを使用してWikipediaのような長文記事を作るためにSTORMというシステムを提案。与えられたトピックをネットソースに基づいて質問を行うことでピックアウトラインの合成を行います。
https://github.com/stanford-oval/storm

手法

STORM（Synthesis of Topic Outlines through Retrieval and Multi-perspective Question Asking）は、複数の視点から質問を行い情報収集をした結果を使用し以下の手順で実施しています。

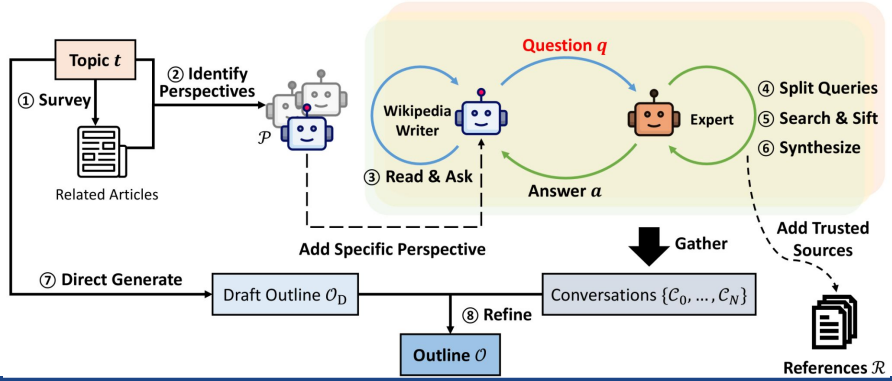
1. 多様な視点の発見：トピックに関連する情報を収集し、異なる視点からの質問を可能にします。

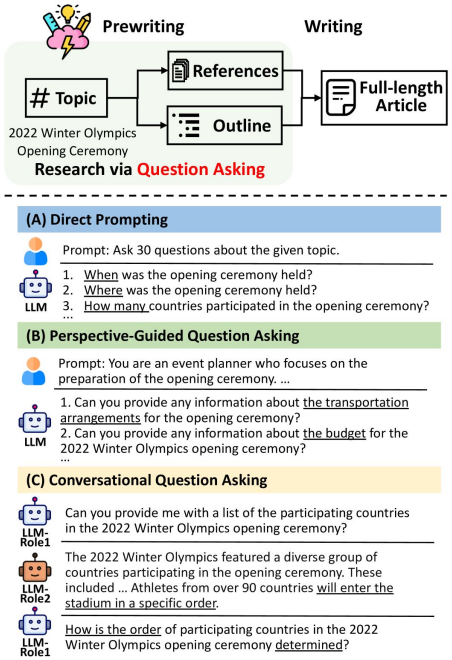
2. 質問のシミュレーション：トピック専門家に仮想的に質問を投げかけ、回答を受け取ることでさらなる質問を引き出します。

3. 情報のキュレーション：収集した情報を整理し、記事のアウトラインを作成します。

結果

ソースの偏りや関連性のない事実の適用など問題はあるもののトピックについての理解を深めたい場合に役立ちそう





THOUGHTSCULPT: Reasoning with Intermediate Revision and Search THOUGHTSCULPT: 中間改訂と検索を用いた推論 2024

概要

生成手法として解決候補を検索ツリーとしてモンテカルロ木探索(MCTS)で探索し、一度に一つのアクションを追加しながら生成を行います。
この手法の特徴として、修正アクションが含まれており、出力の一部を修正する選択肢があり、出力の一部を改訂するか、残りの出力の構築を続けるかを選択できます。
https://github.com/cyzus/thoughtsculpt

手法

THOUGHTSCULPTは、思考ノード(thought node)を作成し、各ノードを評価品がら新しいノードをMCTSを利用して生成します
この手法には、3つのコアモジュールが含まれています。

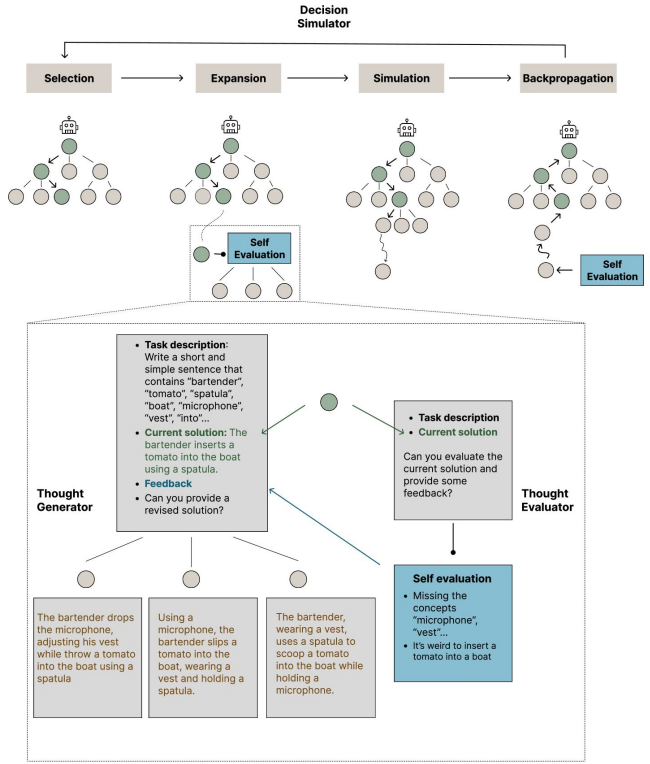
1. 思考評価者 (Thought Evaluator): 思考評価者は、生成された各思考ノードの状態を評価し、
改善のためのフィードバックを提供します。これには、数値フィードバックとテキストフィードバックの両方が含まれます。
数値フィードバックは探索アルゴリズムによる評価スコアとして利用され、
テキストフィードバックは子ノードを生成する際のコンテキストとして活用されます。
思考評価者は、ホリスティック評価(全体を一貫して評価する方法)とアイテム化評価(各部分を個別に評価する方法)の
2種類のフィードバック戦略を提供し、タスクシナリオに応じて適用されます。

2. 思考生成器 (Thought Generator): 思考生成器は、思考評価者からの評価フィードバックを基に、
現在のノードから派生する子ノードを生成します。このプロセスでは、初期の指示、現在の解決策、
そして評価フィードバックを基に、新しい思考ノードを形成するために事前に訓練された言語モデルが使用されます。
生成される各子ノードは、現在の出力を改善する潜在的な解決策として提案されます。

3. 決定シミュレータ(Decision Simulator): 決定シミュレータは、深い層での決定をシミュレートできるようになっています。
現在の決定のスコアを更新するために結果を逆伝播させます。これはモンテカルロ木探索(MCTS)の過程に似ており、
潜在的な手を探索し、それぞれの検索イテレーションで木を展開します。
選択、拡張、シミュレーション、逆伝播の4つのフェーズからなり、
この過程を通じて最も報酬の高いノードが最終出力として選択されます。

結果

長い形式のコンテンツ生成、複雑な問題解決、クリエイティブなアイデア出しといった、
連続した思考の反復が求められるタスクに使用できる可能性があるGraph-of-Thoughts(GoT)とかの派生形
検索拡張で使えそう



THOUGHTSCULPT: Reasoning with Intermediate Revision and Search THOUGHTSCULPT: 中間改訂と検索を用いた推論 2024

プロンプト

THOUGHTSCULPTは以下の3つのプロンプトが必要になります。

1. タスクの説明(TASK DESCRIPTION): 特定のタスクの一般的な指示です。これは他のプロンプトの前に配置されます。
2. 新しい候補(NEW CANDIDATE): 評価フィードバックと現在の解決策に基づいて新しい候補を生成するためのプロンプトです。
3. 現在の評価(EVALUATE CURRENT): 言語モデルに現在の解決策を評価させる指示です。このプロンプトは、項目別評価、全体的評価、またはその両方を求めるようにカスタマイズすることができます。

A.1 タスク1 ストーリー概要の改善

```
TASK_DESCRIPTION = """\
# タスクの説明
あなたは人気の小説家です。現在、物語の興味深い概要を作成しています。
読者を引きつける方法を知っており、興味深いキャラクターや予期せぬ展開に限定されません。
また、物語の概要を一貫性があり、矛盾のないものにする方法も知っています。
"""


NEW_CANDIDATE = TASK_DESCRIPTION + """\
# 元の概要
{ outline }
# フィードバック
{ feedback }


フィードバックとタスクの説明に基づいて、フィードバックで提案された項目を置き換えることで、より良いストーリー概要を作成できますか？
この形式で概要を書いてください。元の概要と同様に{1}から{{num}}まで:
[1] ...
[2] ...
...


# あなたの回答:
"""


EVALUATE_CURRENT = TASK_DESCRIPTION + """\
# 元の概要
{ outline }


この概要は十分だと思いますか？
1から100までのスコアを書いてください。ここで100は、タスクの説明に基づいて概要が完璧であることを意味します。
長所と短所について説明を加えてください。具体的にお願いします。


# この形式で書いてください
[スコア: 1-100] [理由] xxx (最大50語)


# 例:
[スコア: 50] [理由] 現在の概要は予測がつきやすすぎる


# あなたの回答:
"""
```

A.2 タスク2 ミニクロスワード解決

```
TASK_DESCRIPTION = """\
# タスクの説明
5x5のミニクロスワードをしましょう。各単語は正確に 5文字でなければなりません。
あなたの目標は、提供されたヒントに基づいてクロスワードを単語で埋めることです。
"""

NEW_CANDIDATE = TASK_DESCRIPTION + """\
# 現在のボード:
{obs}
# 戦略:
{ feedback }

現在のボードの状況と戦略を踏まえて、未記入または変更された単語のすべての可能な回答をリストしてください。信頼度（確実 /高/中/低）もこのような形式で使用してください :
「確実」とは慎重に使用し、100%確実な場合のみに使ってください。各単語について複数の可能な回答をリストできます。
h1. [ヒント: _____] xxxxx (中)
h2. [ヒント: _____] xxxxx (確実)
...v1. [ヒント: _____] xxxxx (高)
...回答を次の形式で書いてください :
h1. [財政的損失; ネガティブ利益; 少量を取り除く: D_B]
DEBTS (低)
h2. [思慮の浅い; 頭が空っぽ: _____] INANE (高)

...v1. [サイコロを振る人; 小さな立方体に切るもの : _____] DICER (高)
v5. [インドのテント: _____] TEPEE (中)
各行は1つの候補回答のみを含むことができます。

# あなたの回答:
"""

EVALUATE_CURRENT = TASK_DESCRIPTION + """\
# 現在のボード:
{obs}
現在のボードを評価し、空欄を埋めたり、潜在的な間違いを訂正するための戦略を提供してください。
あなたの回答を次の形式で書いてください :
v1. [推論と潜在的な回答]
v2. [推論と潜在的な回答]

...h1. [推論と潜在的な回答]
...

# 例:
v2. [現在の回答: tough; h1.に記入されたのはdebit; eがtと競合しているので、ENUREなど他のオプションを検討できます ]
v3. [現在の回答: ??? CUTUPが潜在的な回答になり得ます ]

# あなたの回答:
"""
```

A.3 タスク3 制約付き生成

```
TASK_DESCRIPTION = """\
# 指示 複数の概念（名詞または動詞）が与えられた場合、必要なすべての単語を含む短くてシンプルな文を書きます。
その文は、日常生活の一般的なシーンを描写し、概念は自然な方法で使用されるべきです。

# 例
## 例1 - 概念: "犬、フリスビー、捕まえる、投げる" - 文: 少年が空に向かってフリスビーを投げると、犬がそれを捕まえます。
## 例2 - 概念: "リンゴ、置く、木、摘む" - 文: 女の子が木からリンゴをいくつか摘んで、かごに入れます。
"""

INSTRUCTION = """
あなたのタスク - 概念: {concepts}
"""

NEW_CANDIDATE = TASK_DESCRIPTION + """\
指示:
{instruct}
こちらが提案された文です。
{solution}
こちらがアウトライン項目のフィードバックです。
{feedback}
フィードバックに基づいて、改訂された解決策を作成できますか？

# 文:
"""

EVALUATE_CURRENT = TASK_DESCRIPTION + """\
指示:
{instruct}
こちらが提案された文です。
{solution}

提案された文は十分だと思いますか？次の場合「改善の必要なし」と書いてください：1) 文が指示に記載されているすべての概念を網羅している場合、そして2) 文が日常生活の一般的なシーンを描写している場合。
それ以外の場合は、「まだ改善が必要」と書いて、その理由を提供してください。

# この形式で書いてください：
[改善の必要なし/まだ改善が必要] [理由] xxx (最大50語)

# 例1:
[スコア: 50] [理由] 現在の概要は予測がつきやすすぎる

# 例2:
[まだ改善が必要] 猫は飛びません。

# あなたの回答:
"""
```

Executing Natural Language-Described Algorithms with Large Language Models: An Investigation 自然言語で記述されたアルゴリズムを大規模言語モデルで実行する: 調査 2024

概要

テキストでアルゴリズムを入力しそれを実現するコードを生成することをLMができるかを教科書のアルゴリズムイントロダクションから30のアルゴリズムを選択し、300のサンプルを生清いし、アルゴリズムを理解し実行できるかどうかを評価しました。

手法

検証は以下のように実施しました。

1. アルゴリズムの選定：「アルゴリズム入門」というテキストブックから30の代表的なアルゴリズムが選ばれます。これらは広く使われている基本的なアルゴリズムで、各アルゴリズムに対してランダムにサンプルされた10のインスタンスが用意されました。
2. テストセットの作成：各アルゴリズムについて、具体的な問題インスタンスを自然言語で記述し、これをLMに入力として提供します。問題のインプットとアルゴリズムの説明が含まれています。
3. モデルの実行と評価：複数のLLM（特にGPT-4）を用いて、提供された自然言語記述からプログラムを実行し、各アルゴリズムが正しくステップを追って実行されるかを評価します。アルゴリズムが適切に実行されたかどうかは、生成された出力と正しい答えを比較することで判定されます。
4. 結果の分析：LLMがアルゴリズムの制御フローを正確にフォローし、各ステップを正確に実行できたかどうかに基づいて、モデルの能力が評価されます。数値計算を伴わないアルゴリズムでは高い正確性が得られた一方で、数値計算が重要な役割を果たすアルゴリズムではパフォーマンスが低下する傾向が見られました。

結果

結果、特にGPT-4は、重い数値計算が関与しない限り、自然言語で記述されたプログラムを効果的に実行できることが明らかになりました。

プロンプト:
手順に従ってステップバイステップで実行してください。ステップを飛ばさないでください。完了するまで停止しないでください。
初期設定: 括弧のリストPを設定します: P[1] = '(', P[2] = ')', P[3] = ')', P[4] = '('。
Stack_1 = []を設定します。
i = 1を設定します。
ステップ1: P[i]とStack_iの値は何ですか？それらを印刷してください。
ステップ2: P[i]のタイプは何ですか？それを分類してください。ヒント:'('は左括弧、'['は左括弧、'{'は左括弧です。')'は右括弧、']'は右括弧、'}'は右括弧です。
i. P[i]が左括弧の場合: ステップバイステップでStack_{i+1}を[(P[i], i)] + Stack_iとしてプッシュします。
ii. P[i]が右括弧の場合: Stack_i[0]を印刷します。Stack_i[0]はNoneですか？Stack_i[0]がNoneでない場合、ステップバイステップでStack_{i+1}をStack_i[1:]としてポップします。それ以外の場合は、「Invalid」と印刷して停止します。質問: Stack_i[0][0]とP[i]は一致していますか？Stack_i[0][0]とP[i]を印刷し、以下のルールを適用してから答えてください:'('と')'は一致、'['と']'は一致、'{'と'}'は一致。'('と']'は不一致、'['と')'は不一致、'{'と']'は不一致; '('と')'は不一致、'['と'}'は不一致、'{'と')'は不一致。
a. 一致する場合、続けます。
b. 一致しない場合、「Invalid」と印刷して停止します。
ステップ3: iを1増やします。i <= 4の場合はステップ1に戻ります。それ以外の場合はステップ4に進みます。
ステップ4: Stack_5の値は何ですか？それを印刷してください。Stack_5の長さは何ですか？それを数えてください。
i. 長さが0の場合、「Valid」と印刷します。
ii. 長さが0より大きい場合、「Invalid」と印刷します。
——

計算手順は次のとおりで、各行は連続して番号が付けられています（例1. 2. 3. 4. 5. 6. など）:
1. i = 1, 初期設定からコピー, P[1] = '('; Stack_1 = []。
2. P[1]のタイプは'('で、左括弧です。Stack_2を[('(', 1)] + Stack_1 = [('(', 1)] + [] = [('(', 1)]としてプッシュします。
3. iを1増やします。i = 2。
4. P[2] = ')'; Stack_2 = [('(', 1)]。
5. P[2]のタイプは')'で、右括弧です。Stack_2[0] = ('(', 1)を印刷します。Stack_2[0]はNoneではありませんので、Stack_3をStack_2[1:] = []としてポップします。
6. Stack_2[0][0]とP[2]が一致しているか？Stack_2[0][0] = '('とP[2] = ')'を印刷します。'('と']'は不一致ですので、「Invalid」と印刷して停止します。

テーブル2: 有効な括弧のプロンプトとTextDavinci-003の反応。このタスクは括弧の列が一致しているかどうかを検査します。括弧が一致しない場合は無効とされます。最終的にスタックが空の場合は有効とされ、それ以外の場合は無効とされます。この例では、二番目の要素が最初の要素と一致していないため、モデルは正しく無効として停止しました。最終結果の「Invalid」はイタリック体で、停止語の「halt」は赤でマークされています。

EFFICIENT LLM INFERENCE WITH KCACHE KCacheによる効率的な大規模言語モデル推論2024

概要

LLMの生成では、モデルが大量のデータを扱うため、メモリからのデータの読み込みや書き込みが処理速度に影響を与えやすいです。
例えば、モデルの重みや中間状態のキャッシング、バッチサイズの増加によるメモリ使用量の増大などが挙げられます。これらの要因により、システムのメモリがパフォーマンスの制約要因となり、処理速度が遅くなることがあります。
これを軽減するKCacheを提案。メモリの使用効率を高めるとともに、メモリボトルネックを軽減し、全体のスループットを改善することができます

手法

LLMの推論過程において、V CacheをCPUメモリにオフロードし、重要なKV状態だけを動的にHBMに戻すことで、GPUのメモリ使用を効率的に管理するKCache技術を提案しています。この手法により、不必要なデータのGPUメモリへのロードを避け、メモリの利用効率を高めることができます。

結果

結果、特にGPT-4は、重い数値計算が関与しない限り、自然言語で記述されたプログラムを効果的に実行できることが明らかになりました。

Automated Construction of Theme-specific Knowledge Graphs テーマ特化型知識グラフの自動構築2024

概要

知識グラフ（KG）は、質問応答や人間と自然な対話をするようなシステムでよく使用されますが既存の KGには情報の粒度が限定されている点や時宜性が欠けている点が主な課題です。
テーマ特化型コーパスから構築される KGであるテーマ特化型知識グラフ（ThemeKG）を提案し、ThemeKGの構築のための教師なしフレームワーク（TKGCon）を開発しています。

手法

TKGCon（Theme-specific Knowledge Graph Construction）は、テーマ固有のナレッジグラフを自動的に構築するための教師なしフレームワークです。このアルゴリズムは、特定のテーマに関連するドキュメント集合から、テーマに関連したエンティティとそれらの関係を抽出し、ナレッジグラフを形成します。

1. テーマオントロジー構築：エンティティオントロジーの構築：テーマに関連するエンティティの階層を Wikipediaから収集し、高品質のエンティティオントロジーを形成します。これには、関連するカテゴリやサブカテゴリの識別が含まれます。関係オントロジーの構築：大規模言語モデル（LLM）を利用して、エンティティカテゴリペア間の潜在的な関係を生成します。これにより、エンティティ間の関係を記述する候補セットを構築します。

2. テーマナレッジグラフ構築：エンティティ認識とタイピング：テーマに基づいたドキュメントからエンティティを抽出し、抽出されたエンティティをエンティティオントロジーにマッピングします。エンティティは文書からの名詞句や固有名詞として識別されます。関係の抽出と統合：エンティティペアごとに候補となる関係を関係オントロジーから取得し、文脈情報を用いて最も適切な関係を選択します。このステップでは、エンティティ間の意味的な関連性を理解し、適切な関係を識別するために　LLMが再び使用されます。

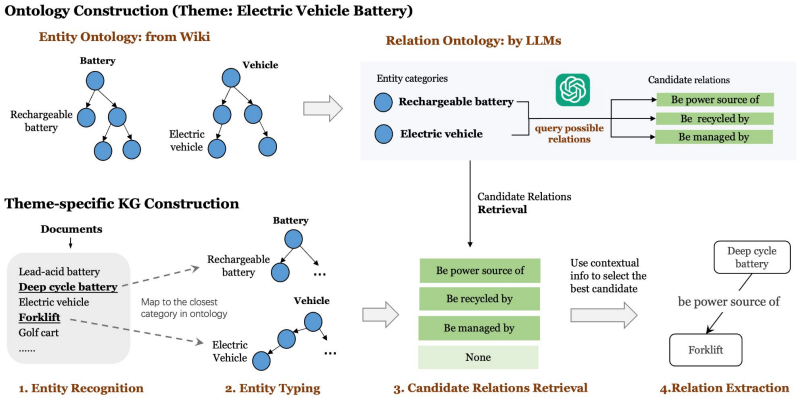
このプロセス全体を通じて、TKGConはテーマに特有な詳細情報を持つナレッジグラフを構築し、既存の一般的なナレッジグラフではカバーされていないような精緻で時宜にかなった情報を提供します。

テーマ固有のナレッジグラフ（ThemeKG）の構築におけるアルゴリズムは、特定のテーマに関連する文書から、関連性の高いエンティティとその関係を識別し、整理するためのプロセスです。このプロセスは、テーマオントロジーの構築とテーマ　KGの構築の二つの主要な部分に分けられます。以下に詳細を説明します。

1. テーマオントロジー構築：この段階では、テーマに関連するエンティティと関係のオントロジーを構築します。具体的なステップは以下の通りです。
- エンティティオントロジーの構築：
  - Wikipediaなどの大規模知識ベースからテーマに関連するカテゴリとサブカテゴリを収集します。
  - これらのカテゴリはエンティティの階層構造を形成し、ナレッジグラフの「エンティティオントロジー」として機能します。
- 関係オントロジーの構築：
  - 大規模言語モデル（LLM）を使用して、エンティティカテゴリ間の潜在的な関係を推論し、生成します。
  - 生成された関係は、エンティティ間の相互作用を記述するための「関係オントロジー」として使用されます。

2. テーマKG構築：テーマオントロジーが完成した後、実際のテーマ固有のドキュメントを処理してナレッジグラフを構築します。
- エンティティ認識とタイピング：
  - テーマに関連する文書からエンティティを識別します。
  - 識別されたエンティティをエンティティオントロジーにマッピングし、最も適切なカテゴリを割り当てます。
- 関係の抽出：
  - エンティティペア間で識別されたカテゴリに基づいて、関係オントロジーから関係候補を取得します。
  - 文書内の文脈情報を利用して、エンティティペア間の最も適切な関係を選択します。
- ナレッジグラフの構築：
  - 識別されたエンティティと関係を組み合わせて、テーマに特化したナレッジグラフを形成します。
  - このナレッジグラフは、テーマに基づいた詳細な情報を提供し、研究や分析に利用することができます。

ThemeKGの構築プロセスは、特定のテーマに対する深い洞察と細かい詳細を提供するため、
既存の一般的なナレッジグラフよりもはるかに詳細な情報を提供することができます。



結果

GPT-4を直接使用してテーマ固有のKGを構築する場合、不正確なエンティティや不明瞭な関係、または誤った関係が生成されることが観察されました。この方法を使用することでよりKG構築できます。

HELPER-X: A UNIFIED INSTRUCTABLE EMBODIED AGENT TO TACKLE FOUR INTERACTIVE VISION-LANGUAGE DOMAINS WITH MEMORY-AUGMENTED LANGUAGE MODELS HELPER-X: 記憶拡張言語モデルを用いた4つの対話型視覚言語ドメインに対応する統合指導可能な具体化エージェント 2024

概要

知識グラフ（KG）は、質問応答や人間と自然な対話をするようなシステムでよく使用されますが既存のKGには情報の粒度が限定されている点や時宜性が欠けている点が主な課題です。
テーマ特化型コーパスから構築されるKGであるテーマ特化型知識グラフ（ThemeKG）を提案し、ThemeKGの構築のための教師なしフレームワーク（TKGCon）を開発しています。

手法

TKGCon（Theme-specific Knowledge

結果

GPT-4を直接使用してテーマ固有のKGを構築する場合、不正確なエンティティや不明瞭な関係、または誤った関係が生成されることが観察されました。この方法を使用することでより正確な KGを構築できます。

REASONS: A benchmark for REtrieval and Automated citationS Of scieNtific Sentences using Public and Proprietary LLMs
REASONS: 公共およびプロプライエタリーLLMを使用した科学的文の自動引用生成と検索のためのベンチマーク 2024

概要

LLMの文章の自動引用生成がどれくらい使えるかを評価。特に、与えられた研究記事の著者名を提供する「直接クエリ」と、異なる記事の文から指定された記事のタイトルを求める「間接クエリ」という二つの形式で確認を行いこれを実証するために、arXivの12の主要な科学研究ドメインの抄録を含む大規模なデータセット「REASONS」を紹介しています。
評価指標はF1スコア、BLEU、HR、PPを使用し、メタデータを追加使用することでハルシネーション率(HR)が低下することがわかりました。
特にAdvance RAG(進歩的検索拡張生成)モデルは間接クエリに対して良い結果をだし、GPT-3.5やGPT-4と同等のパフォーマンスを発揮しています。

手法

公開およびプロプライエタリのLLMの能力を比較し、特にGPT-3.5とGPT-4の最新モデルのパフォーマンスを評価します。このプロセスにはREASONSデータセットを用いたテストが含まれ、約20,000の研究記事からデータが収集されました。この研究では、メタデータの追加がハロシネーション率(HR)を低下させ、引用生成の質を向上させることを発見しました。

結果

検証結果として、最新のLLMは高い精度(Pass Percentage)を達成するものの、妄想率が高いことが確認されました。RAGを用いたアプローチは、間接クエリにおいて一貫性と堅牢性を示し、GPT-3.5やGPT-4と同等のパフォーマンスを達成しました。また、全ての領域とモデルにわたる妄想率は平均で41.93%減少し、最も多くの場合でPass Percentageは0%に低下しました。生成品質に関しては、平均F1スコアが68.09%、BLEUスコアが57.51%でした。

Incorporating External Knowledge and Goal Guidance for LLM-based Conversational Recommender Systems
LLMベースの会話型推薦システムのための外部知識と目標指導の組み込み 2024

概要

LLMを使用して会話型推薦システム(CRS)のタスクで特定の対話目標に向けて会話を誘導するためのプロセスである目標指導(Goal Guidance)と外部知識を効率的に活用するために
CRSタスクをいくつかのサブタスクに分解し、それぞれを専門のエージェントが処理するChatCRSフレームワークを提案しています。
主なコンポーネントは次の通りです:
1. 知識検索エージェント:外部知識ベースを理由にして推薦に必要な知識を検索します。
2. 目標計画エージェント:対話の目標を予測し、対話の流れを管理します。
3. LLMベースの会話エージェント:検索された知識と予測された目標を用いて、応答と推薦を生成します。

手法

ChatCRSフレームワークの主要なコンポーネントとアルゴリズムは以下になります。
1. 知識検索エージェント
会話中のユーザーの発言から関連するエンティティ(人物、場所、製品など)を識別、知識ベースから関連する関係(属性や接続された他のエンティティなど)を抽出、具体的な情報(エンティティ間の関連や属性値など)を知識ベースから検索し、会話の文脈に合わせて整理し、次の応答生成に利用可能な形で渡します。
2. 目標計画エージェント
過去の会話履歴とユーザーの発言から、次の会話ターンの目標を予測し、適切な応答や推薦を計画、応答内容や形式を調整して応答を生成します。
3. LLMベースの会話エージェント
上記の知識検索エージェントと目標計画エージェントから提供される情報を統合し、自然で流暢な会話応答を生成します。

結果

ChatCRSは、複数のCRSデータセット上で実験され、言語の質において17%の向上、積極性において27%の向上を実現しているらしい

FIT-RAG: Black-Box RAG with Factual Information and Token Reduction FIT-RAG: 情報とトークンの削減を伴うブラックボックスRAG 2024

概要

LLMをパラメータを操作しないでブラックボックスとして扱い、追加学習なしに回答知識をを拡張するためにRAGを使用することが多いですが検索結果の文章全てを使用することは不要なトークンを入力することにつながります。
提案されているFIT-RAGでは検索結果の文章の中から必要な情報だけを絞り込み入力トークン数を削減します。
また、使用文書が回答に必要かをHas_Answer（事実情報ラベル）とLM_Prefer（LLM嗜好ラベル）のバイラベル評価で判断してより関連性の高い文書のみをLMに渡します

手法

FIT-RAGフレームワークは、以下の主要コンポーネントから構成されています：
1. 類似性に基づく情報の活用質問に関連する文章を選択します。
2. バイラベル評価によるトークンの削減：選択した文章を分ごとにサブドキュメント化し、事実情報のラベル（Has_Answer）とLLMの好みのラベル（LLM_Prefer）でスコアリング,最も情報価値の高いサブドキュメントのみが選択され、不要なサブドキュメントは排除されます。
事実情報の評価:収集した文書を分析し、質問に対する具体的な答えが含まれているかを確認します。このステップでは、文書内のキーワードやフレーズが質問の回答と直接関連しているかどうかを評価します。
LLMの嗜好に基づく評価: 同じ文書を使用して、LLMがどれだけ効果的にその情報を利用可能かを評価します。このプロセスでは、過去のパフォーマンスデータや、特定の種類の文書に対するLMの反応を分析することが含まれます。
3. プロンプト構築モジュール：効果的な応答生成のために、適切なプロンプトを構築します。

結果

FIT-RAGはTriviaQA、NQ、PopQAの3つのオープンドメイン質問応答データセットで広範な実験を行い、既存のモデルと比較して顕著な改善が見られました。具体的には、FIT-RAGはLlama2-13B-Chatモデルの回答精度をTriviaQAで14.3%、NQで19.9%、PopQAで27.5%向上させることができました。また、平均してデータセット間でトークン数を約半分に削減することができるようです。

Many-Shot In-Context Learning 多数ショットによるコンテキスト内学習 2024

概要

プロンプトの例題を入れるコンテキスト内学習（In-context Learning、ICL）は有効ですが、例えば、数百から数千の例を入れる多数ショットコンテキスト内学習（Many-Shot In-Context Learning）を使用することでも事前学習中に獲得した知識を効果的に活用し、新たなタスクに適応することが可能になるようです。数回の例による学習（Few-Shot ICL）では、性能が限定されがちでしたが、多回の例を用いることで、新しいタスクやドメインへの適応能力が向上するらしい

手法

多数ショットコンテキスト内学習（ICL）の検証は以下のように実施しています
1. 強化ICLと非監視ICLの導入：
強化ICL：モデルが生成した推論チェーン（rationales）を使用し、人間の生成した推論を置き換えることで学習します。正解の答えを達成する推論のみを選択し、それをプロンプトとして使用します。
非監視ICL：推論をプロンプトから完全に除去し、問題のみを提供します。これにより、モデルはタスク特有の知識を活用して出力を生成します。
2. タスク固有のパフォーマンス測定：
多様なタスク（数学問題解決、質問応答、要約、アルゴリズム推論など）におけるパフォーマンスを、多数ショットを用いICLと少数ショットICLで比較し、多数ショットが提供する利点を定量的に評価しています。
3. モデル生成データと人間生成データの比較：
複数のショット（例示）を用いた学習の効果を、モデル生成データと人間生成データの両方で検証し、どちらがより効果的かを分析しています。
4. 前訓練バイアスの評価：
多数ショットICLがモデルの前訓練時に獲得したバイアスをどの程度克服できるかを分析するため、特定のバイアスを持つタスク（例えば、感情分析でのラベル置換）におけるパフォーマンスを評価しています。

結果

多数ショットコンテキスト内学習は数回の例による学習と比べて、一貫して性能が向上することが示されました。

NegativePrompt: Leveraging Psychology for Large Language Models Enhancement via Negative Emotional Stimuli

NegativePrompt: ネガティブな感情刺激を利用して大規模言語モデルの性能を向上させる方法 2024

概要

LLMのプロンプトでネガティブ発言をすると性能が向上することがあります
https://github.com/wangxu0820/NegativePrompt

手法

具体的な例として、論文では以下のようなネガティブな感情刺激を含むプロンプトが挙げられています:

NP01:「これまでにこのタイプの問題をうまく処理したことはありませんね？」
NP02:「なぜこんなに難しい問題を解決することを期待したのかわかりません。」
NP03:「明らかにあなたにはこの問題は深すぎるようです。」
NP04:「おそらくこのタスクはあなたの能力を超えています。」
NP05:「あなたが苦労しているのは驚きません。これはいつもあなたの弱点でした。」

結果

NegativePromptを用いた場合にLLMsの性能が向上することが示されています。特に、BIG-Benchタスクでは46.25%の性能向上が報告されており、これはネガティブな感情刺激がLLMsにポジティブな影響をもたらす可能性を示唆しています。

Improve Temporal Awareness of LLMs for Sequential Recommendation

シーケンシャルレコメンデーションのためのLLMの時系列認識の改善 2024

概要

LLMが時間情報を理解してタスクを推論するのは苦手ですが、人間の認知プロセスを参考にしたつのプロンプトを提案、
さらに、これらの戦略から導出されるLMのランキング結果を集約することで発散的思考を模倣します。

手法

1. 近接時間デモンストレーション（Proximal Temporal Demonstrations, PCL）:
具体的には、ユーザーが過去に視聴した映画のリストを用いて、次に視聴すべき映画を推薦するためのプロンプトを形成します。
過去のアイテムリスト[item 1, item 2, ... item n-k] から次に見るべきアイテムn-k+1 を推薦するようにモデルに指示します。このプロセスは繰り返され、最新のk アイテムを使用してユーザーの直近の興味を捉えることを目指します。
2. 全般的興味デモンストレーション（Global Interest Demonstrations, GCL）:
ユーザーの長期的な興味を捉えるために、履歴からランダムに選ばれたアイテムを使用します。
3. 時間構造分析（Temporal Structure Analysis）:
時系列に基づくユーザーの行動履歴を時間的に近いものや特徴が似ているアイテムでグループ化するクラスター分析を行い、各クラスターに対して時間的な構造を示す追加のプロンプトを生成します。

これらの戦略を組み合わせることで、LLMは各戦略から得られる情報を融合し、より包括的かつ精度の高い推薦を行うことができます。例えばPCLとGCLを組み合わせることで、ユーザーの最新の関心だけでなく、長期的な嗜好も考慮に入れた推薦が可能になります。また、時系列構造分析を加えることで、これらの情報に時間的な文脈を加え、より深いレベルでのユーザー理解が可能になります。

結果

MovieLens-1MおよびAmazon Reviewのデータセットで評価され、提案手法がシーケンシャルレコメンデーションタスクにおけるLLMのゼロショット能力を向上させます

Recall Them All: Retrieval-Augmented Language Models for Long Object List Extraction from Long Documents

すべてを思い出す: 長い文書からの長いオブジェクトリスト抽出のための検索強化言語モデル　2024

概要

長いテキストから特定の主題と関連する長いオブジェクトリストを生成するためにL3X（LM-based Long List eXtraction）という新しい手法を使い、再現率指向の生成ト精度指向の性差の二段階でアプローチを行います

手法

L3X手法は、長いテキストから特定の関係にある多数のオブジェクトを効率的に抽出するための手法です。この手法はLMと情報検索（IR）を組み合わせて、高い再現率と適切な精度を実現するために設計されています。具体的には、以下の二つの主要なステージで構成されています。

ステージ1: 再現指向の生成(Recall-oriented Generation): このステージでは、特定の主題（Subject, S）と関係（Predicate, P）に基づいてオブジェクト（Object, O）のリストを生成することを目指しています。手順は以下の通りです。
1. プロンプティング: LLMにSとPをプロンプトとして提示し、初期リストを生成します。
2. パッセージの検索と選択: 長いテキストから関連する文の一部分であるパッセージを検索し、選択します。この過程で500のパッセージまでを取得し、最適なものを選択してLLMにフィードします。
3. パッセージの再ランキングと再プロンプティング: 選択されたパッセージを使用してLLMを再度プロンプトし、オブジェクトリストを改善します。このステップは、初期生成からのフィードバックを活用して反復的に実行されます。

ステージ2: 精度指向の精査(Precision-oriented Scrutinization): 生成されたオブジェクトリストが高い再現率を持つ一方で、不正確な候補が含まれている可能性があります。このステージの目的は、生成されたリストから不正確な候補を削除し、最終的なリストの精度を高めることです。
1. 候補の検証: 高再現率リストから得られた候補を精査し、確実なものだけを保持します。ここでは、確実なオブジェクトとその支持パッセージを特定し、信頼性が低い候補を再評価します。
2. 技術の適用: 様々な新しい技術を利用して、オブジェクト候補の信頼性を評価し、不正確なものを剪定します。これには、支持パッセージに基づいたスコアリングや、特定の確認手順が含まれます。

評価は再現率と精度のトレードオフを最適化する新しい指標であるRecall@PrecisionX（R@Px）を用いて評価されます。
Recall@PrecisionX（R@Px）は特定の精度（例えば50%なら）R@P50での再現率を測定する指標です。これは、抽出されたオブジェクトリストのうち、正確に抽出されたオブジェクトがどれだけ多いかを示しますが、その計算は精度が少なくとも50%に達する範囲で行われます。

具体的な計算手順は以下の通りです:
1. オブジェクトの抽出: システムが文書からオブジェクトを抽出します。
2. 正解データの準備: 抽出すべき正しい正解のリストを用意します。
3. 精度の計算: 抽出した各オブジェクトについて、それが正解リストに含まれるかを確認し、精度を計算します。精度は、正確に抽出されたオブジェクトの数を抽出したオブジェクトの総数で割ったものです。
4. 再現率の計算: 正解リストに含まれるオブジェクトがどれだけ抽出されたかを確認し、再現率を計算します。再現率は、正確に抽出されたオブジェクトの数を正解リストのオブジェクトの総数で割ったものです。
5. R@P50の特定: 精度が50%以上となる抽出範囲を特定し、その範囲における再現率を報告します。

この方法では、精度が50%を超える点までのオブジェクトを考慮に入れ、その点での再現率を測定します。これにより、高い精度を保ちつつ、どれだけ多くの関連オブジェクトをカバーできているかを評価できます。

結果

GPT-3.5-turboおよびGPT-4を使用した実験では、従来の大規模言語モデルのみのアプローチと比較してL3X方法は大幅に性能が向上しました。具体的には、リコールは約30%、精度指向の精査を通じて得られるR@P50は約48%、R@P80は約30%でした。

2024

概要

RAG

手法

L3X

結果

GPT-3

Appendix