Projeto - Tomada de Decisão (IN1087)

Novembro de 2023

Igor Victor Lucena do Nascimento
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
ivln@cin.ufpe.br

Monique Louise de Barros Monteiro
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
mlbm@cin.ufpe.br

I. Framework de modelagem de problemas sequenciais

A. Narrativa

Um framework de agente tendo um *Large Language Model* (LLM) como o agente que aprende a tomar decisões melhores com base na memória de experiências de episódios passados do processo de aprendizagem por reforço. O objetivo é atualizar a memória de experiências com os dados de observações, ações e recompensas para que o agente LLM possa tomar melhores decisões futuras.

B. Elementos centrais

- **Métricas** - O sistema deve retornar a máxima expectância das estimativas dos valores baseado na última observação, ação, recompensa e a nova observação. Nos exemplos descritos no artigo [1], métricas mais concretas incluem a taxa de sucesso em dois *benchmarks* recentes: navegação por um site de comércio eletrônico (WebShop [2]), onde o objetivo é comprar um determinado produto especificado como entrada, e navegaçáo por um site de perguntas e respostas (WikiHow [3]), onde o objetivo é chegar à página que contém a resposta correta para uma dada pergunta.
- **Decisões** - Precisa-se decidir qual é a melhor ação a ser tomada com base nas informações da memória de experiências do sitema. Seguindo os exemplos fornecidos nos benchmarks realizados, as decisões referem-se à descrição textual da opção de navegação a ser tomada pelo usuário (no caso, o agente), como selecionar um link, realizar uma busca, retornar à página inicial, confirmar uma compra etc.
- **Fontes de incertezas** - Uma das fontes de incerteza é o resultado das observações do ambiente nas futuras tomadas de decisão. Assim, a cada etapa do seu processo de decisão, o LLM obtém uma observação do ambiente, no caso concreto, a descrição textual dos elementos da tela no qual ela está navegando, em ambos os exemplos.

C. Modelo matemático

1) Variáveis de estado: A variável de estado pode ser modelada como a memória de experiências resultantes das interações com o ambiente até o momento t. Assim, podemos representar cada estado S_t como sendo:

$$S_t = (O_t, A_t, Q_t)$$

no qual as experiências são representadas como a sequência de observações O_t até o momento, bem como as respectivas ações A_t e estimativas de valor Q correspondentes Q_t.

Assumimos que a memória de experiências é inicializada com alguns registros iniciais. Tais registos são necessários para informar ao LLM o formato da entrada e da saída. Logo, a variável de estado inicial é dada por:

$$S_0 = (O_0, A_0, Q_0)$$

2) Variáveis de decisão: A variável de decisão x_t é a ação que o LLM vai executar dependendo da tarefa a ser realizada. Por exemplo, no caso do WebShop, o agente é instruído a navegar no site de uma loja para comprar os produtos desejados. Assim, assumindo-se um LLM auto-regressivo que prevê o próximo token de acordo com a distribuição probabilística sobre o vocabulário, a decisão x_t rerefe-se à sequência de tokens predita pelo modelo.

Em geral, LLMs não são determinísticos. Assumindo que sua saída é uma distribuição de probabilidades pelos tokens do vocabulário (incluindo, por exemplo, o token que indica o fim da geração e portanto o tamanho da sequência gerada), a técnica de amostragem a partir dessa probabilidade é variável e muitas vezes configurável. Por exemplo, pode ser adotada a heurística de sempre prever o token de maior probabilidade (greedy-decoding), mas também podem ser adotadas variações como beam search, top-k, top-p, temperatura, etc. Todos esses fatores acabam por influenciar a sequência a ser gerada, de modo que para uma mesma entrada, o modelo pode gerar diferentes saídas em diferentes momentos.

3) Informação exógena: De acordo com os autores, a cada passo de decisão, o LLM recebe uma observação do ambiente o_t, que por sua vez é utilizada para recuperar várias experiëncias relacionadas a partir da memória, de acordo com uma função de similaridade. Mas a observação é resultante da ação executada no passo $t-1$, e conforme citado anteiormente, tal ação não é determinística, podemos

assumir que a observação também não é determinística, vindo a constituir portanto um elemento de incerteza. Sendo assim, vamos modificar ligeiramente a notação para adaptá-la à do framework de Warren Powell [4], adotando a terminologia W_{t+1}:

$$W_{t+1} = o_{t+1}$$

4) Função de transição: Durante a interação com o ambiente, novas experiências são adicionadas à memória de experiências. Logo, dada a tupla (o_t, a_t, r_t, o_{t+1}), contendo a última observação, a ação executada, a recompensa associada e a nova observação, temos as seguinte equações:

$$O_{t+1} = O_t \cup \{o_t\}$$

$$A_{t+1} = A_t \cup \{a_t\}$$

$$Q_{t+1} = Q_t \cup \{Q'(g, o_t, a_t)\}$$

sendo g a informação sobre a tarefa a ser realizada, e

$$Q'(g, o_t, a_t) = r_t + \gamma \max_a Q(g, o_{t+1}, a)$$

, caso não exista um registro associado a (g, o_t, a_t) na memória. Caso contrário, o valor Q estimado será atualizado por meio do algoritmo Q-Learning:

$$Q'(g, o_t, a_t) \leftarrow (1 - \alpha)Q(g, o_t, a_t) + \alpha Q'(g, o_t, a_t)$$

, sendo $\alpha = \frac{1}{N}$, e N denota o número de vezes em que esse valor é atualizado.

5) Função objetivo: No benchmark WebShop, um escore entre 0 e 1 é fornecido depois da compra por meio da correspondência entre o produto comprado e a instrução original.

Seja $a^{\pi,N}$ a escolha final de a produzida pela política $X^\pi(S|\theta)$ depois que um orçamento com N experimentos é consumido. Para avaliar o antingimento do objetivo, temos que testar ao longo de muitos possíveis resultados de W. Para distinguir os resultados de W observados durante as iterações de treinamento dos resultados obtidos em tempo de teste/inferência do modelo final, denotamos as observações de W durante o treinamento por $W^1, ..., W^N$, seguidos por $\hat{W}$, que denota o resultado observado pelo teste.

$$\max_\theta F^\pi(\theta) = \mathbb{E}_{S_0}\mathbb{E}_{W^1,...,W^N|S_0}\mathbb{E}_{\hat{W}|S_0} Sim(y, \hat{o}^N)$$

sendo $Sim(\hat{a}^N, \hat{o}^N)$ uma função de similaridade entre o produto comprado $\hat{o}^N$ e o produto y da instrução original.

Já no benchmark WikiHow, recompenas e instruções intermediárias podem ser fornecidas ao longo do episódio de duração T. Assim, temos:

$$\max_\theta F^\pi(\theta) = \mathbb{E}_{S_0}\mathbb{E}_{W^1,...,W^T|S_0} \left\{ \sum_{t=0}^{T} Sim(y_{t+1}, s_{t+1}) \right\}$$

sendo $Sim(y_{t+1}, s_{t+1}) = r_t$ a similaridade entre cada página esperada e cada página s_{t+1} (observação/estado) visitada pelo LLM (agente) como resultado da execuçãp da ação x_t.

D. Modelagem da incerteza

Segundo Powell [4], quando estimativas são baseadas em observações de campo, o método é dito data-driven, o que significa que não precisamos de um modelo de incerteza - apenas observamos sua saída.

No caso concreto, o LLM é o modelo, baseado em rede neural, responsável por gerar sequências de tokens que descrevem ações a serem realizadas, de forma auto-regressiva, que também pode ser compreendido como um modelo de Markov, no qual a probabilidade de geração de cada token é condicionada à probabilidade de geração dos tokens anteriores.

Além disso, cada saída do agente LLM é utilizada para atualizar a memória de experimentos, conforme explicado nos itens 3 e 4 do modelo matemático.

E. Projetando políticas

Para cada passo (estado/observação), refletido pela descrição da tela atual, em ambos os benchmarks, o sistema recupera um conjunto de experiências - tuplas (observação, ação, recomensa) - selecionadas de acordo com similaridade com a observação atual.

Tal histórico de experiências é repassado para o LLM, por meio de prompts, onde tanto a melhor ação como a pior ação são informadas, de acordo com suas respectivas recompensas. Uma vez que o LLM recebe exemplos de experiências passadas, os respectivos exemplos de ações recomendadas e não recomendadas, tal estratégia pode ser considerada uma forma de *few-shot learning*, ou *few-shot prompting*.

Uma vez que os parâmetros do modelo de linguagm prétreinaddos não são atualizados, os únicos pontos sobre os quais o sistema tem controle é a memória de experiências e os prompts que são utilizados como entrada para o LLM. Nesse contexto, um elemento numérico crucial é a atualização dos valores Q para cada tupla (tarefa, observação, ação), por meio do algoritmo Q-Learning. São esses valores que permitem informar, nos exemplos repassados ao LLM, quais são as melhores e as piores ações para cada situação. Assim, temos uma política do tipo VFA (*value function approximation*), que pode ser expressa por:

$$X^\pi(S_t|\theta) = arg\max_x \ F^\pi(\theta)$$

REFERÊNCIAS

[1] D. Zhang, L. Chen, S. Zhang, H. Xu, Z. Zhao, and K. Yu, "Large language models are semi-parametric reinforcement learning agents," 2023.

[2] S. Yao, H. Chen, J. Yang, and K. Narasimhan, "Webshop: Towards scalable real-world web interaction with grounded language agents," 2023.

[3] D. Zhang, L. Chen, Z. Zhao, R. Cao, and K. Yu, "Mobile-env: An evaluation platform and benchmark for interactive agents in llm era," 2023.

[4] W. B. Powell, "Sequential decision analytics and modeling: Modeling with python - part ii," *Foundations and Trends in Technology, Information and Operations Management*, vol. 16, no. 1-2, pp. 1–176, 2022. [Online]. Available: https://doi.org/10.1561/0200000103-II