Projeto Orientado em Computação 2
Modelos de Linguagem para Recuperação de Informação

Arthur Pontes Nader[1]

[1]Universidade Federal de Minas Gerais

arthurnader@dcc.ufmg.br

Abstract. *This report presents the results obtained when using language models to perform information retrieval tasks. The tasks tested were: extracting structured information, searching for information in documents and generating SQL queries.*

Resumo. *Esse relatório apresenta os resultados obtidos na utilização de modelos de linguagem para realização de tarefas de recuperação de informação. As tarefas testadas foram: extração de informação estruturada, busca de informação em documentos e geração de consultas SQL.*

1. Introdução

No Projeto Orientado em Computação 1 (POC1), analisou-se diversas ferramentas capazes de facilitar a recuperação de informação por meio do uso de modelos generativos. Essas ferramentas, apesar de representarem um salto importante na área, apresentaram uma série de limitações.

O treinamento desses modelos generativos é uma tarefa que exige recursos computacionais significativos e um conjunto de dados adequado para garantir que o modelo seja capaz de gerar informações relevantes e precisas. Com isso, a exploração de modelos já prontos, juntamente com sua integração com outras ferramentas já existentes, se mostra como uma alternativa promissora e eficaz para adaptar esses modelos a tarefas específicas.

Nesse sentido, é útil estudar bibliotecas para processamento de linguagem natural (NLP), como a biblioteca Transformers da Hugging Face, ou frameworks de deep learning, como TensorFlow e PyTorch. Essas ferramentas fornecem implementações eficientes de modelos generativos pré-treinados, como GPT-3, BERT, entre outros, que podem ser ajustados para atender às necessidades específicas de recuperação de informação.

Outro framework que tem ganhado bastante destaque ultimamente é o LangChain, que permite o desenvolvimento de aplicativos alimentados por modelos de linguagem, com foco na capacidade de integração com fontes de dados e na interação com o ambiente. Tem-se também o Llama 2, uma família de modelos de linguagem de grande escala lançados pela Meta.

Assim, o principal objetivo desse trabalho é explorar essas bibliotecas e frameworks para determinar qual seria um modelo ou arquitetura mais adequado para os diversos tipos de tarefas em recuperação de informação explorados no POC1. Esses conhecimentos serão bastante úteis e podem servir de base para uma pesquisa de mestrado em computação.
Usou-se esses modelos para realização de três tarefas de recuperação de informação: extração de informação estruturada de uma página da web, busca de informação em documentos de texto extraídos de um conjunto de páginas e geração de consultas SQL a partir de comandos em linguagem natural.
Os códigos implementados para realização dessas tarefas de recuperação de informação por meio do uso de modelos de linguagem podem ser acessados no seguinte link: https://github.com/arthurnader/projeto_orientado_2

2. Referencial teórico

A arquitetura Transformer, exposta em (VASWANI, et al., 2017), tem sido muito utilizada em modelos para processamento de linguagem natural, sendo inclusive citada em (RADFORD, et al., 2018), um dos artigos iniciais que foram a base para o surgimento dos modelos GPT. Esses dois artigos merecem destaque quando se trata do entendimento desses modelos de inteligência artificial.
Por sua vez, em (WOLF et al., 2020) é apresentada a biblioteca "transformers", que disponibiliza esses avanços na área para a comunidade de aprendizado de máquina, o que possibilita que os profissionais da área explorem e utilizem modelos de linguagem pré-treinados em suas próprias aplicações
Já em (TOUVRON et al., 2023) é mostrado uma coleção de modelos de linguagem de base com até 65 bilhões de parâmetros. Esses modelos são treinados em um volume massivo de dados e destacam a possibilidade de treinar modelos de alta qualidade usando apenas conjuntos de dados publicamente disponíveis, sem depender de dados proprietários.
Por fim, no âmbito de eficiência computacional, é exposto em (GEIPING, GOLDSTEIN, 2022) a possibilidade de treinar um modelo de linguagem em uma única GPU em um dia, propondo modificações para alcançar desempenho comparável a modelos escalados, o que é interessante de se pensar tendo em vista a limitação de recursos computacionais.

3. Resultados

As atividades desenvolvidas consistiram tanto em explorar determinadas bibliotecas e frameworks voltados para processamento de linguagem natural como em adaptar os modelos de linguagem para as tarefas de recuperação de informação citadas. Assim os resultados podem ser divididos entre essas duas diferentes etapas do projeto: exploração e implementação.

Exploração

a) Transformers

A biblioteca transformers, desenvolvida pela Hugging Face, representa um grande avanço em aprendizado de máquina para as plataformas PyTorch e TensorFlow. Essa biblioteca oferece interfaces de programação de aplicativos (APIs) e ferramentas que simplificam o processo de download e treinamento de modelos pré-treinados de última geração.

A principal vantagem dessa biblioteca reside na sua capacidade de suportar uma variedade de tarefas comuns em diferentes modalidades. No contexto de processamento de linguagem natural, esses modelos pré-treinados oferecem suporte a um conjunto diversificado de tarefas, tais como: classificação de texto, reconhecimento de entidades nomeadas, resposta a perguntas, modelagem de linguagem, resumo, tradução e geração de texto.

Outro aspecto interessante da biblioteca é sua facilidade de uso. É possível realizar essas tarefas complexas com menos de dez linhas de código.

Como exemplo de utilização da biblioteca transformers, testou-se sua funcionalidade de classificação de texto em um cenário associado a um sistema de recomendação. Nesse contexto específico, a biblioteca foi empregada para analisar um comentário relativo a um produto, e classificá-la com uma avaliação numa escala de 1 a 5. A título de exemplo, a seguinte frase foi submetida ao modelo: "Gostei muito do filme, emocionante", sendo que o correspondente índice de avaliação atribuído foi 5, indicando, assim, uma boa relação entre o comentário e sua classificação.

b) Langchain

O LangChain se destaca como um framework inovador que proporciona a capacidade de criar uma ampla variedade de aplicativos por meio da integração de modelos de linguagem. Sua funcionalidade se deve principalmente na sua

capacidade de conectar de forma eficaz o modelo de linguagem a diversas fontes de contexto. Atualmente, o LangChain apresenta um conjunto de seis módulos de trabalho, sendo que os três a seguir foram experimentados com maior profundidade:

- Chains: esse módulo está relacionado à combinação de diferentes modelos de linguagem para realização de tarefas específicas. No caso testado, primeiramente usou-se um modelo gratuito da Google, sendo que sua saída foi a entrada de um modelo pago da OpenAI. No primeiro modelo, pediu-se uma sugestão de 5 lugares para se visitar em um país. Já o segundo modelo tem a função de estimar os custos e o tempo necessário para realizar essas visitas.
- Memory: consiste na capacidade do modelo em reter informação. Há diversos tipos de memória, sendo que o testado foi o ConversationBufferMemory. Basicamente, para experimentação do módulo, conversou-se com um certo modelo e ao final fez-se perguntas específicas sobre parte da conversa, sendo que ele foi capaz de responder adequadamente.
- Data Connection: por fim, este módulo implementa a capacidade de estabelecer uma conexão entre o modelo e diversos documentos por meio de uma série de passos. Ele foi usado na implementação da busca de informação em documentos, sendo que será melhor explicado nessa parte do relatório.

De forma geral, esse framework se mostrou muito útil para uso de modelos de linguagem e seu aprendizado é relativamente rápido.

Implementação

c) Extração de informação estruturada

Para realização dessa tarefa, foi necessário o uso da biblioteca urllib, para fazer a solicitação da página da web, BeautifulSoup, para o parser e a extração do texto do HTML e a biblioteca transformers para busca da informação específica no texto.

Além disso, o código produzido usou a implementação de resposta a pergunta do transformers e foi capaz de extrair corretamente as informações solicitadas de dois diferentes sites, como pode ser visto no seguinte arquivo do repositório: extracao_de_informacao_estruturada.ipynb

De certa forma, essa abordagem é um modo alternativo tanto à recuperação de informação clássica baseada na extração de partes específicas do HTML como

ao modelo pago explorado na POC1 para essa tarefa, chamado Scrapeghost. Os resultados foram bem similares ao modelo pago e bem mais simples de serem obtidos quando comparados com a abordagem clássica.

d) Busca em documentos

Para usar um modelo de linguagem para busca de informação em textos extraídos da web, foi necessário utilizar o LangChain, assim como as bibliotecas usadas na implementação anterior: urllib e BeautifulSoup. As seguintes etapas descrevem o processo realizado:

- Web Crawling: essa etapa consiste em gerar links a partir de uma url base e seguir esses links
- Extração de texto: cada url gerada tem o texto da sua página extraído e armazenado em um documento.
- Load: consiste em carregar o documento gerado pela união dos textos das diversas páginas.
- Transform: essa etapa realiza o processamento do documento, como por exemplo, dividindo-o em várias partes para cada uma delas poderem ser melhores representadas para utilização no modelo.
- Embed: cada divisão do documento é transformado em um vetor de embeddings representativo daquela parte do texto.
- Store: armazena-se os documentos e os embeddings associados.
- Retrieve: a partir de uma consulta passada pelo usuário, o modelo retorna o documento em que se encontra a resposta.

O exemplo executado consistiu em partir de uma página da Wikipédia que fala sobre programação de computadores e extrair o texto dela e o texto de alguns links gerados a partir dessa página inicial. Em seguida, esse texto gerado foi passado para o modelo acima e fez-se quatro perguntas para o modelo sobre diferentes aspectos das linguagens de programação. O modelo foi capaz de retornar o documento exato em que se encontravam as respostas para as quatro perguntas.

Esses resultados podem ser reproduzidos a partir da execução do seguinte arquivo do repositório: busca_em_documentos.ipynb

e) Geração de consultas SQL

Por fim, a implementação de geração de consultas em SQL a partir de comandos em linguagem natural usou o pipeline de geração de texto do transformers, o modelo LlaMA2 disponibilizado pela HuggingFace e algumas funções de prompt e do módulo chain do LangChain.

Basicamente, a implementação consiste em utilizar o tokenizer do LlaMA2 com 7 bilhoes de parâmetros no pipeline criado e definir um prompt baseado em um template que solicita a criação de comandos em SQL a partir de um texto representativo da consulta em linguagem natural. O prompt e o modelo então são passados para instanciar um objeto Chain do LangChain.

As consultas geradas a partir de comandos em linguagem natural foram satisfatórias e parecem representar bem o que foi solicitado. Entretanto, elas não foram testadas em um caso real de consulta a um banco de dados.

O código dessa última implementação está disponível no seguinte arquivo do repositório: consultas_sql.ipynb

4. Conclusão

A realização desse projeto orientado em computação proporcionou uma oportunidade significativa de aprendizado. A exploração de bibliotecas como transformers revelou a acessibilidade e facilidade do uso dos modelos disponibilizados, permitindo a execução de tarefas complexas com eficiência. A experimentação com modelos pré-treinados, como o LlaMA2, revelou a capacidade dessas ferramentas em gerar respostas precisas a partir de comandos em linguagem natural.

O uso do LangChain para desenvolver aplicativos alimentados por modelos de linguagem destacou a importância das fontes de contexto para um bom desempenho dos modelos de inteligência artificial. A flexibilidade oferecida por esse framework permitiu a exploração de seus diferentes módulos de trabalho, ampliando consideravelmente o repertório computacional para resolução de uma grande variedade de problemas.

Os resultados obtidos nas implementações foram considerados bem-sucedidos. A biblioteca transformers, por exemplo, demonstrou eficácia na tarefa de pergunta e resposta sobre uma página da web, enquanto o LangChain possibilitou a criação de aplicativos com capacidade de conexão a diferentes modelos de linguagem e fontes de contexto.

As tarefas de extração de informação estruturada, busca em documentos e geração de consultas SQL apresentaram resultados bem satisfatórios, evidenciando a aplicabilidade prática desses modelos de linguagem em diferentes contextos da área de recuperação de informação.

Em conclusão, o projeto proporcionou não apenas avanços pessoais no uso de modelos de linguagem nas tarefas específicas de recuperação de informação, mas também uma base sólida para pesquisas futuras.

5. Referências

R. Baeza-Yates, B. Ribeiro-Neto. Modern Information Retrieval. 2011. 2nd ed. New York. Addison-Wesley.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems, pages 6000–6010, 2017.

A. Radford, K. Narasimhan, T. Salimans, I. Sutskever. Improving Language Understanding by Generative Pre-Training. OpenAI, 2019. Disponível em: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... Rush, A. M. (2020). HuggingFace's Transformers: State-of-the-art Natural Language Processing. arXiv preprint arXiv:1910.03771v5.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., ... Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. arXiv preprint arXiv:2302.13971v1.

Geiping, J., & Goldstein, T. (2022). Cramming: Training a Language Model on a Single GPU in One Day. arXiv preprint arXiv:2212.14034v1.