# Projeto Orientado em Computação 1
# Modelos Generativos 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 from the use of generative models to facilitate information retrieval tasks. The aim is to compare the results obtained with other methods in order to evaluate the effectiveness of generative model-based web scraping tools.*

***Resumo.*** *Esse relatório apresenta os resultados obtidos na utilização de modelos generativos para facilitar a realização de tarefas de recuperação de informação. Busca-se comparar os resultados obtidos com outros métodos, com intuito de avaliar a eficácia das ferramentas de web scraping baseadas em modelos generativos.*

## 1. Introdução

Os modelos de inteligência artificial generativa têm sido cada vez mais utilizados nas mais diversas atividades humanas. Esses são modelos de IA são capazes de aprender as relações entre as características dos dados de treinamento e usar esse conhecimento para gerar novas amostras que possuam as mesmas características. Desde resumir um tópico de interesse até a criação de códigos em diversas linguagens de programação, essas ferramentas apresentam potencial para alavancar o desenvolvimento de muitos campos da Ciência da Computação.

Uma dessas áreas é a Recuperação de Informação, que, de acordo com (BAEZA-YATES, RIBEIRO-NETO, 2011) no livro "Modern Information Retrieval", é uma área de pesquisa que lida com o armazenamento de documentos e a recuperação da informação associada a eles a partir de uma necessidade de informação do usuário. A importância desse campo se torna ainda evidente quando se considera a recuperação de informação da Web, que teve um grande avanço desde então. Algumas tarefas frequentes dessa área são:

- Extração de informação: consiste em identificar entidades, relacionamentos e eventos em um texto.
- Web Crawling: relacionado a atividade de gerar links de uma página e seguir esses links.
- Web Scraping: criação de modelos automatizados que interagem com uma webpage, por exemplo, clicando em botões, preenchendo formulários, identificando informações nas tags do HTML. Tem como principal objetivo baixar documentos, recuperar dados tabelas, etc.

Essas tarefas podem ser realizadas no cotidiano por humanos, mas demandam muito tempo e se tornam repetitivas para grandes volumes de dados a serem recuperados. Para

serem automatizadas, frequentemente é necessário conhecimento sobre algoritmos e linguagens de programação.

O fato é que esses recentes modelos generativos podem estar alterando esse cenário, em que essas tarefas poderão ser feitas por pessoas que não possuem todo o conhecimento técnico necessário, por meio de uma interação direta em linguagem natural com o modelo, tais como: “Extraia os dados de determinada coluna da tabela”, “Clique no botão Download”, “Extraia as entidades do texto a seguir” e “Construa um crawler que a partir de uma URL base, visite todas as URLs até a profundidade 3”.

Assim, o principal objetivo desse trabalho é reproduzir essas tarefas utilizando esses modelos generativos e avaliar os resultados obtidos, comparando com métodos usualmente utilizados.

Os códigos implementados e as interações com os modelos generativos estão disponíveis no seguinte link: https://github.com/arthurnader/projeto_orientado_1

## 2. Referencial teórico

As redes neurais têm sido um dos métodos de inteligência artificial mais populares em modelos de aprendizado de máquina atualmente. Dentre os vários tipos de redes, 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.

A extração de informação estruturada é um dos desafios da inteligência artificial. Em um texto, podem ter várias entidades relacionadas com um mesmo fato, e, o que é facilmente compreendido por um humano, às vezes se torna uma tarefa árdua para um computador.

Uma das estratégias que usa modelos generativos para resolver isso é exposta em (WEI, et al., 2023) em que os autores apresentam a ferramenta ChatIE, um framework baseado em ChatGPT para extração de entidades em um texto.

Já para a tarefa de web scraping, há ferramentas bem recentes que usam esses modelos generativos que, apesar de não possuírem artigo científico relacionado, merecem a devida atenção. Exemplos de ferramentas baseadas em GPT-3 e GPT-4 são o ExtractGPT, Scrapeghost e TaxyAI. Todas essas ferramentas possuem uma ampla descrição em seus repositórios/sites.

Por fim, uma referência muito útil para implementações práticas de web crawling e web scraping em Python é o livro “Web Scraping com Python: Coletando Dados na Web Moderna” da autora Ryan Mitchell.

## 3. Resultados

Para melhor acompanhamento da discussão dos resultados obtidos, aconselha-se abrir os notebooks presentes no repositório associados a cada tópico a seguir. Cada notebook produzido possui códigos relacionados à tarefa de recuperação de informação em questão, bem como um conjunto de imagens que mostram a interação com os modelos generativos.

### a) Extração de entidades

O objetivo dessa tarefa, tal como mencionado anteriormente, é identificar e extrair informações específicas, tais como nomes de pessoas, locais, organizações e datas presentes no texto.

Para esse fim, foi utilizada a biblioteca Spacy em Python, que possui um conjunto de ferramentas para se trabalhar com processamento de linguagem natural. Usou-se um texto literário e o texto de uma página para os testes. Durante o processo, observou-se algumas classificações incorretas, como a classificação do verbo "Releva" no início de uma frase como um local. Uma das características do Spacy é justamente essa, sua tendência em classificar entidades com base na inicial ser maiúscula ou minúscula.

Já no contexto de uso dos modelos generativos, usou-se o ChatGPT para realização da tarefa. Foi adotada uma abordagem em que se forneceram descrições das entidades e solicitou-se a extração dessas entidades do texto. Inicialmente, o modelo extraiu uma quantidade menor de entidades em relação ao Spacy, porém, ao persistir na solicitação, foi possível extrair uma quantidade maior de entidades do texto. O ChatGPT foi menos propenso ao viés de classificar de acordo com a inicial da palavra, apesar de também cometer certos erros de classificação.

### b) Web Crawling

Para avaliação da tarefa de web crawling, pensou-se em três atividades distintas, cada uma com uma funcionalidade específica. A primeira tinha como objetivo gerar os links presentes em uma página, enquanto a segunda tinha como propósito gerar apenas os links que contivessem certas palavras-chave e que não saíssem da URL base. Por fim, o terceiro código visava gerar e seguir links a partir de uma URL base até os links de profundidade 3.

Usualmente, essas atividades podem ser implementadas utilizando a biblioteca BeautifulSoup, um parser de HTML, capaz de identificar os elementos do HTML que contém os links desejados. Com base nesses elementos, é possível extrair as URLs correspondentes, o que com verificações adicionais e com a

realização do processo em loop, possibilita a realização devida dessas três atividades descritas.

Novamente, foi utilizado o modelo generativo ChatGPT para a tarefa de gerar funções equivalentes por meio de comandos em linguagem natural. O modelo generativo obteve êxito na construção do primeiro e terceiro códigos, apresentando resultados bastante semelhantes. Já no segundo código, que exige uma semântica um pouco mais complexa, o modelo não conseguiu produzir resultados adequados.

## c) Web Scraping

### - Extração de tabelas

A primeira tarefa de scraping realizada consiste na extração de tabelas de dados. Diversos sites são organizados de forma que o HTML é construído com intuito de exibir uma tabela. Como exemplo relacionado, tem-se o seguinte site http://www.pythonscraping.com/pages/page3.html citado no livro "Web Scraping com Python: Coletando Dados na Web Moderna". A tabela em questão contém informações como nome do item, custo, entre outros.

Para realizar essa tarefa de extração de colunas, foi utilizada a biblioteca BeautifulSoup em Python. Após um estudo do HTML, é possível localizar a coluna desejada e iterar pelos elementos irmãos, extraindo o texto correspondente. Essa abordagem resultou em um código conciso, mas que exige um certo nível de conhecimento sobre o assunto.

A ferramenta ExtractGPT foi avaliada para a extração dos mesmos dados, sendo que se obteve resultados bastante satisfatórios. Para utilizá-la, basta fornecer os nomes das colunas desejadas, e ela é capaz de extrair não apenas texto, mas também imagens presentes em colunas. Testou-se a ferramenta para diversos casos de tabelas de dados de portais de Transparência de municípios de Minas Gerais, e em todos eles a extração foi realizada com sucesso.

No entanto, foram identificadas algumas limitações na ferramenta ExtractGPT. A principal delas é a restrição de apenas poder passar nomes de colunas como parâmetro. Solicitações mais complexas, como a extração da coluna anterior a uma coluna específica, não produzirão resultados corretos, tarefa essa pode ser realizada com sucesso utilizando a biblioteca BeautifulSoup.

### - Extração de informação estruturada

A extração de informações específicas de um site possui algumas diferenças do caso anterior. Agora os dados não estão dispostos em colunas, o que exige

abordagens adicionais além da iteração pelas tags irmãs do HTML, estratégia utilizada anteriormente.

Novamente, utiliza-se a biblioteca BeautifulSoup e a extração da informação estruturada é feita logo após avaliação da localização dos dados de interesse no HTML.

O modelo generativo avaliado para facilitar a recuperação das mesmas informações foi a ferramenta Scrapeghost, que permite a criação de um esquema descritivo dos dados desejados, seu formato e, por meio da passagem da URL, realiza a extração da informação do site em questão. Essa ferramenta proporcionou uma maneira mais fácil de extrair as informações, obtendo resultados iguais aos do uso do BeautifulSoup.

No entanto, também foram identificados alguns problemas na ferramenta Scrapeghost. A primeira limitação está relacionada à dependência da estrutura do HTML. Quando os dados de interesse estão contidos em tags relacionadas, mas diferentes, aparentemente não é possível realizar a recuperação adequada desses dados. Além disso, a ferramenta apresenta restrições em relação ao número de tokens, onde HTMLs com mais de 8192 tokens não podem ser processados, nem mesmo pelo GPT-4

- **Processamento dinâmico**

Por fim, tem-se a tarefa de processamento dinâmico, que envolve a interação com elementos da interface de um site, como clicar em botões e preencher campos de pesquisa.

Para executar essa tarefa, tem-se um conjunto de bibliotecas em Python que realizam o processo de automação web, tal como Scrapy e Selenium. Entretanto, utilizou-se a biblioteca Playwright, que ultimamente tem sido classificada como mais robusta que as demais.

Vamos supor que se deseja clicar em um botão de uma página para baixar um arquivo CSV. Uma das principais vantagens da programação tradicional é a modularização. Por meio da criação de uma função que realize essa tarefa, essa mesma função pode ser aplicada em diversas outras páginas com a mesma configuração, bastando passar a URL de cada página como parâmetro para recuperar diversos dados semelhantes.

O modelo generativo associado é a ferramenta TaxyAI. Essa ferramenta permite interagir com o site por meio do uso de comandos em linguagem natural pela interface. Como exemplo testado, foi dado um comando específico para clicar em um botão CSV de uma página, o que acabou resultando no correto download do arquivo correspondente.

Apesar disso, durante a análise da ferramenta TaxyAI, identificou-se uma série de problemas e limitações:

- A ferramenta tende a entrar em loop, repetindo o mesmo comando diversas vezes.
- Há restrições quanto às ações possíveis, limitando-se a cliques e preenchimento de campos de pesquisa.
- Novamente, existem limitações relacionadas à quantidade de tokens.
- Pelos testes realizados, o HTML deve ser fixo, ou seja, clicar em um botão que muda a estrutura do HTML pode comprometer as demais ações.

## 4. Conclusão

Em conclusão, torna-se evidente que os modelos generativos utilizados para gerar soluções mais práticas para tarefas na área de recuperação de informação ainda apresentam uma série de limitações e problemas. Essas limitações podem ser atribuídas ao atual estado da arte da inteligência artificial generativa na área de recuperação de informação.

Uma das principais limitações identificadas é a dificuldade dos modelos generativos em lidar com semânticas e situações mais complexas, como a extração precisa de informações a partir de estruturas não convencionais. Além disso, a dependência da estrutura do HTML em ferramentas de scraping e as restrições relacionadas ao número de tokens processados são obstáculos que impactam a eficiência e a flexibilidade desses modelos.

Além disso, um fato importante que deve ser destacado é que o uso desses modelos é pago, o que pode ocasionar em grandes gastos a longo prazo caso sejam amplamente utilizados.

Embora algumas ferramentas, como TaxyAI e ExtractGPT, já permitam interação com o modelo por linguagem natural, as limitações identificadas podem dificultar a automatização da recuperação de informação para pessoas sem conhecimento em computação, principalmente se a tarefa a ser executada for complexa. Esses aspectos devem ser considerados ao explorar a aplicação prática dessas ferramentas em cenários reais.

Portanto, uma interessante continuidade para esse projeto seria uma implementação própria de modelos generativos para recuperação de informação, o que possibilitaria a ampliação de funcionalidades a fim de contornar algumas dessas limitações. Essa implementação local dos modelos também eliminaria preocupações relacionadas a custos.

## 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. Acesso em: 07 abr. 2023.

X. Wei, et al. Zero-Shot Information Extraction via Chatting with ChatGPT. Disponível em: https://arxiv.org/pdf/2302.10205.pdf. Acesso em: 07 abr. 2023.

R. Mitchell. Web Scraping com Python: Coletando Dados na Web Moderna. 2019. 2nd ed. Novatec Editora.