UM SISTEMA DE CHATBOT INTELIGENTE PARA A CONFIGURAÇÃO DE COMPONENTES DO COMPUTADOR

GIOVANI G. L. SILVA[1], FLÁVIA B. R. P. CUNHA[2]

[1] Graduando em Tecnologia de Análise e Desenvolvimento de Sistemas, IFSP, Campus Jacareí, giovani.g@aluno.ifsp.edu.br.
[2] Professora EBTT, Doutora em Computação Aplicada, IFSP, Campus Jacareí, flavia.beatriz@ifsp.edu.br.
Área de conhecimento (Tabela CNPq): 1.03.03.04-9 Sistemas de Informação

RESUMO: Este trabalho apresenta um sistema computacional destinado a auxiliar o usuário na tarefa de definir as peças necessárias para a obtenção de um bom computador. Para o desenvolvimento do sistema, foram estudadas diversas áreas da computação, dentre elas a área de Inteligência Artificial, com ênfase na subárea de Processamento da Linguagem Natural. O objetivo deste trabalho foi o desenvolvimento de um *chatbot* para usuários que não tenham conhecimentos suficientes sobre as peças que compõem um computador, de modo que, através de uma conversa em linguagem natural, o sistema lhes ofereça apoio na tarefa de escolher uma configuração de *hardware* adequada às suas necessidades.

PALAVRAS-CHAVE: Processamento da Linguagem Natural; Inteligência Artificial; Computador; Chat.

AN INTELLIGENT CHATBOT SYSTEM FOR CONFIGURING COMPUTER COMPONENTS

ABSTRACT: This works presents a computer system designed to help the user define the parts needed to make a good computer. In order to develop the system, several areas of computing were studied, including Artificial Intelligence, with an emphasis on Natural Language Processing. The aim of this work was to develop a chatbot for users who do not have sufficient knowledge about the parts that compose a computer, so that through a conversation in natural language, the system can offer them support in the task of choosing a hardware configuration that suits their needs.

KEYWORDS: Natural Language Processing; Artificial Intelligence; Computer; Chat.

INTRODUÇÃO

Para adquirir um PC (*Personal Computer*), *desktop*, por exemplo, surgem várias necessidades e custos, que vão desde a aquisição de teclado e mouse até a placa mãe e placa de vídeo. Além disso, para a montagem de uma máquina que funcione, há a necessidade de se entender a diferença de desempenho e compatibilidade desses componentes. Alguns sistemas, embora possam fornecer informações sobre a compatibilidade e desempenho, não possuem um assistente virtual ou *chatbot* para auxiliar os usuários, e quando possuem, geralmente não utilizam IA (Inteligência Artificial), apenas respostas pré-programadas.

Segundo o estudo de Caldarini (2022), os *chatbots* eficientes ajudam os usuários a concluir tarefas com menos interações e respostas mais rápidas e precisas, melhorando a usabilidade e a satisfação geral do usuário. Isso pode ser de muita ajuda para tarefas que exigem velocidade na conclusão.

"O Processamento de Linguagem Natural (PLN) é uma área que utiliza métodos computacionais para o estudo de linguagem natural (falada ou escrita)" (Pereira et.al, 2020; Caseli e Nunes, 2024). Portanto, sistemas ou modelos que utilizam PLN têm potencial para serem amigáveis e interativos, em razão do uso da linguagem natural (LN) que permite ao usuário uma comunicação em sua própria linguagem. Por outro lado, a falta de uma linguagem natural, que provenha do PLN, pode fazer com que os usuários desistam de utilizar um sistema, devido ao receio de encontrarem dificuldades na comunicação ou insatisfação nas interações com o assistente virtual.

Com base nessa necessidade, foi identificado o potencial de uma ferramenta que aplica PLN e utiliza uma linguagem natural. Dessa forma, um *chatbot* pode ser mais eficiente e amigável, melhorando a experiência e usabilidade dos usuários para a configuração do *hardware*, principalmente aqueles com pouco entendimento de computadores.

MATERIAL E MÉTODOS

Em resumo, o desenvolvimento do *chatbot* (modelo) envolveu o levantamento de requisitos, a definição do fluxo de processamento, a codificação em *Python* (2024), linguagem disponível no *Kaggle* (2024), que é o ambiente de desenvolvimento do modelo, e utilizando a biblioteca *Transformers* (2024). A abordagem utilizada neste trabalho para desenvolver o *chatbot* é conhecida como *fine tuning* (ajuste fino).

"Um modelo *fine-tuned* é construído a partir de um treinamento adicional em cima de um modelo base, para executar uma tarefa específica ou absorver conhecimento sobre um domínio. Os casos de uso geralmente envolvem *datasets* de classificação de texto, perguntas e respostas e tradução" (Zanini, 2024).

"A ideia central do processo de *fine tuning* é aproveitar o conhecimento presente nos modelos base e suas capacidades de generalização" (Zanini, 2024). Vale ressaltar que treinar um modelo do zero requer um alto gasto com recursos de *hardware*, principalmente quando se trata de médios e grandes modelos, e o tempo de treinamento é muito maior. Dependendo do tamanho do modelo que se quer alcançar e do tamanho do *dataset*, o tempo de treinamento pode levar horas ou até dias.

Para este projeto, foi criado um pequeno *dataset* de 300 linhas, utilizando perguntas e respostas relacionadas principalmente a *hardware* de computadores. Mas também contendo informações aleatórias sobre outros assuntos, para evitar que o modelo perca o conhecimento adquirido anteriormente do modelo base.

Conforme ilustra o diagrama na Figura 1, para formar o *chatbot* (novo modelo), foi utilizado como modelo base o *pierreguillou/gpt2-small-portuguese* (Guillou, 2020), treinado com um *dataset* menor que contém dados específicos sobre *hardware*. Esse modelo base foi anteriormente treinado com dados em português, e é voltado para geração de texto.

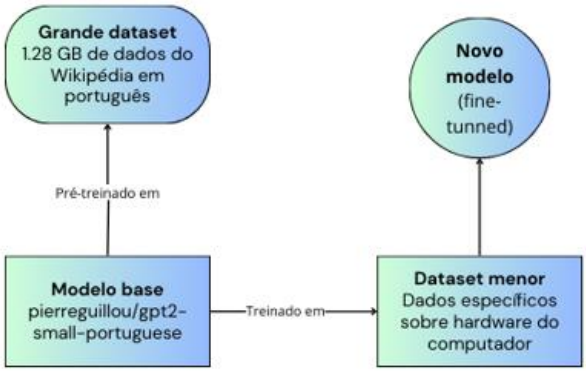

FIGURA 1. Diagrama geral do *fine tuning* do modelo.

De início, o modelo base fornece a configuração da arquitetura e pesos pré-treinados, como também a configuração e o vocabulário do *tokenizer*. Esses dados são armazenados nos novos objetos *model* e *tokenizer* para serem usados ao longo do processo. Em seguida, vem a etapa do pré-processamento de texto, onde o *tokenizer* faz a "Tokenização" do texto, provido do *dataset*, transformando o texto em *tokens* e cada *token* tendo seu próprio *ID* (Identidade). Depois deste processo, os *tokens* voltam à forma de texto.

Uma vez finalizado o pré-processamento, o texto é dividido em dados de treino e de validação. São definidos os parâmetros de treinamento, tais como número de épocas de treino, que representa a quantidade de vezes que os dados são treinados, em seguida a taxa de aprendizagem, que representa a capacidade de aprendizagem do modelo. O parâmetro redução de peso representa um valor utilizado para manter o equilíbrio no treinamento do modelo e o *batch_size* representa o tamanho dos subconjuntos de treino e de validação.

Após a definição dos parâmetros de treino, é iniciado o treinamento com dados divididos em dados de treino e validação. Durante o treinamento, são exibidos os registros como: acurácia (Accuracy), que representa a precisão do modelo para cada época; perda de validação (Validation loss), que representa o erro calculado da quantidade de dados que o modelo consegue generalizar; e perda de treinamento (Training loss), que representa o erro calculado na adequação do modelo ao conjunto de dados para cada época de treino.

Feito isso, o treinamento é salvo e utilizado para a etapa de geração de resposta do *chatbot*, onde a função Geração de resposta é usada para gerar sequências de *tokens* baseadas no *prompt* recebido, ou seja, na entrada do usuário. Por fim, a função principal inicia o *chatbot*, que pode ser encerrado quando o usuário digitar "sair". A Figura 2 contém as etapas do fluxo de processamento.

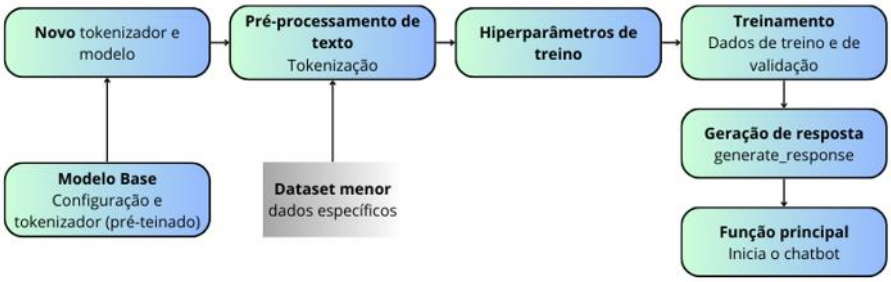

FIGURA 2. Diagrama contendo as principais etapas do fluxo de processamento do *fine tuning*.

RESULTADOS E DISCUSSÃO

No processo de treinamento, os dados do *dataset* são divididos para aprendizagem do modelo. Os dados de treino são utilizados para aprendizagem do modelo e os dados de validação/teste são utilizados para avaliar a performance do modelo nos intervalos após cada época.

Para o treinamento eficiente de um modelo, é necessário realizar testes de desempenho, pois não é possível avaliar somente com o registro obtido no treinamento, neste caso a acurácia, perda de treinamento e de validação. Mesmo com um valor de acurácia considerado bom, por exemplo: 0.882487, é possível que o desempenho não seja melhor do que de um treinamento que resulta em um valor de acurácia menor, e o mesmo vale para perda de treinamento e de validação.

Dessa forma, para cada treinamento realizado, foi necessária a realização de testes de desempenho com o *chatbot*, simulando uma conversa entre usuário e assistente virtual. Portanto, para estes testes em específico, era esperado uma conversa em turnos, ao invés do assistente virtual gerar a solução de uma só vez. Especificamente falando, era necessário que o assistente virtual pudesse entender a entrada do usuário, sabendo que muitas vezes os usuários costumam escrever de maneira informal, ou sem acentuação, como no dia a dia. Além disso, as respostas do assistente virtual para cada entrada do usuário deviam ser coerentes e naturais.

Também vale ressaltar a importância de usar o parâmetro *seed*, que foi definido com o valor 3 no início do código, para que os resultados obtidos no treinamento pudessem ser reproduzidos depois. A utilização do valor 3 se dá por ele ter tido o melhor desempenho em relação aos outros valores testados. A seguir, são apresentados os valores dos parâmetros utilizados no treinamento com o melhor

resultado obtido: taxa de aprendizagem: 0.0003, subconjunto de treino: 8, subconjunto de validação: 8, número de épocas de treino: 15, redução de peso: 0.04.

O treinamento teve uma boa acurácia, considerando que quanto mais próximo de 1 o valor, maior a precisão do modelo no treinamento. Embora, por padrão, a acurácia deveria aumentar a cada época de treino, ao invés de se manter a mesma como neste caso. A Figura 3 apresenta as saídas com as métricas de treinamento.

Epoch	Training Loss	Validation Loss	Accuracy
1	0.422300	0.379358	0.869582
2	0.295900	0.385671	0.869456
3	0.224400	0.410366	0.869456
4	0.152400	0.436447	0.869456
5	0.101600	0.457140	0.869456
6	0.067300	0.479796	0.869456
7	0.048400	0.488559	0.869456
8	0.039000	0.495329	0.869456
9	0.031900	0.500741	0.869456
10	0.026200	0.509239	0.869456
11	0.023100	0.512909	0.869456
12	0.022100	0.516299	0.869456
13	0.020400	0.517887	0.869456
14	0.018200	0.519899	0.869456
15	0.017600	0.520758	0.869456

FIGURA 3. Saídas das métricas de treinamento – resultado final.

As Figuras 4, 5, 6 e 7 apresentam interações entre o usuário e o assistente virtual (*Bot*). Na Figura 4, é ilustrado o resultado do primeiro teste de desempenho do modelo pós-treinamento, no qual o *chatbot* conseguiu fornecer uma configuração de PC para jogos grandes ao usuário. Para tal resultado, o modelo precisou entender o contexto de "jogar" e entender a diferença entre os tipos de jogos.

```
Você:  Quero montar um PC, por onde eu devo começar?
Bot:  -> Primeiro você precisa definir se é para jogar, trabalhar com edição de
vídeo, imagens, ou só para usar a internet.
Você:  é para jogar
Bot:  ->São jogos grandes ou jogos menores e mais leves?
Você:  jogos grandes
Bot:  ->Então você vai precisar de uma placa de vídeo intermediária, de 20 GB d
e memória RAM, de um SSD de 500 GB de armazenamento ou mais, um processador int
ermediário e uma placa-mãe intermediária para suportar esses componentes, e é c
laro, uma fonte de alimentação adequada.
```

FIGURA 4. Teste de desempenho do *chatbot* simulando um usuário que procura um PC para jogos grandes.

Na Figura 5, o assistente virtual (*Bot*) gerou ao usuário uma configuração de PC simples para usar internet. Nesse caso, o modelo precisou entender o contexto de "usar internet" como ponto principal para alcançar o resultado.

```
Você:  Quero montar um PC, por onde eu devo começar?
Bot:  -> Primeiro você precisa definir se é para jogar, trabalhar com edição de
vídeo, imagens, ou só para usar a internet.
Você:  é para usar internet
Bot:  ->Se for só para internet, então você não precisa de placa de vídeo dedic
ada, 8 GB de memória RAM é suficiente, você vai precisar também de um processad
or com placa de vídeo integrada e placa mãe básicos, e uma fonte de alimentação
de baixa potência já é suficiente.
```

FIGURA 5. Teste de desempenho do *chatbot* simulando um usuário que procura um PC para usar a internet.

Na Figura 6, o modelo também mostrou um bom desempenho, pois o assistente virtual (*Bot*) forneceu ao usuário uma configuração de PC adequada para trabalho com edição de vídeo. Da mesma forma, ao compreender "trabalho com edição de vídeo" o modelo pode chegar ao resultado obtido.

```
Você:  Quero montar um PC, por onde eu devo começar?
Bot:  -> Primeiro você precisa definir se é para jogar, trabalhar com edição de
vídeo, imagens, ou só para usar a internet.
Você:  é para trabalhar com edição de video
Bot:  ->Bom, para trabalhar com edição de vídeo, você pode utilizar componentes
de nível intermediário ou robustos, por exemplo: uma placa de vídeo de alto des
empenho da geração atual, 32 GB de memória RAM ou mais, processador da atual ge
ração que suporta trabalhar em alta performance por um longo período, uma placa
mãe robusta que suporta esses recursos, 1 TB de armazenamento e uma fonte de al
imentação de alta potência e qualidade.
```

FIGURA 6. Teste de desempenho do *chatbot* simulando um usuário que procura um PC para trabalho com edição de vídeo.

A Figura 7 também apresenta o bom resultado pelo desempenho do assistente virtual (*Bot*), ao fornecer uma configuração PC para jogos leves adequada ao usuário. Portanto, conseguir diferenciar os tipos de usos para o PC e os tipos de jogos apresentados levou o modelo a este resultado.

```
Você:  Quero montar um PC, por onde eu devo começar?
Bot:  -> Primeiro você precisa definir se é para jogar, trabalhar com edição de
vídeo, imagens, ou só para usar a internet.
Você:  é para jogar
Bot:  ->São jogos grandes ou jogos menores e mais leves?
Você:  jogos leves
Bot:  ->Neste caso, você pode optar por componentes de entrada como uma placa d
e vídeo mais básica, 16 GB de memória RAM, um processador e placa mãe básicos p
ara jogos, um SSD de 250 GB de armazenamento, e uma fonte equivalente.
```

FIGURA 7. Teste de desempenho do *chatbot* simulando um usuário que procura um PC para jogos leves.

Os resultados indicam que quanto menor o *dataset* utilizado para o treino, mais difícil será avaliar o desempenho do modelo com base somente nas métricas de treinamento, como *accuracy* e *validation loss* principalmente, pois, nessas condições, essas métricas podem ter um comportamento fora do padrão esperado. Também indicam que o *dataset* é uma das partes mais importantes, pois quanto melhor a qualidade do *dataset*, ou seja, do conjunto de dados que serão treinados, melhor será o desempenho do modelo. Um *dataset* maior também significa que o modelo terá mais dados para aprender, aumentando a aprendizagem, a capacidade de entender os contextos e de generalizar. Além disso, com o resultado apresentado, foi possível observar que o *chatbot* conseguiu entender a linguagem do usuário, mesmo que essa fosse levemente informal, e também gerou respostas intuitivas e coerentes, promovendo uma conversa atrativa e amigável com o usuário.

CONCLUSÕES

Após uma análise criteriosa, pode-se concluir que o presente projeto atingiu o objetivo de fornecer um *chatbot* capaz de auxiliar o usuário na escolha dos componentes do computador, interagir em uma linguagem natural e, ao mesmo tempo, mantendo os termos técnicos necessários para se referir ao *hardware*. Conclui-se, também, que o *fine tuning* é uma abordagem ou método muito interessante, pois reduz o esforço computacional, minimiza o tempo de treinamento e os recursos necessários à construção do *chatbot*, além de entregar resultados satisfatórios ou razoáveis.

CONTRIBUIÇÕES DOS AUTORES

Giovani Gomes Leandro da Silva (orientado) contribuiu com a curadoria, análise dos dados, metodologia, experimentos e redação do trabalho. Flávia Beatriz Rodrigues Prisco da Cunha (orientadora) contribuiu com auxilio na pesquisa, escrita e revisão do trabalho.

Todos os autores contribuíram com a revisão do trabalho e aprovaram a versão submetida.

AGRADECIMENTOS

A todos que participaram, direta ou indiretamente, do desenvolvimento deste trabalho de pesquisa, enriquecendo o meu processo de aprendizado.

REFERÊNCIAS

CALDARINI, G.; JAF, S.; MCGARRY, K. A Literature Survey of Recent Advances in Chatbots. **Information**, v. 13, n. 1, Basel, Switzerland, jan. 2022.

CASELI, H. M., NUNES, M.G.V. **Processamento de Linguagem Natural: Conceitos, Técnicas e Aplicações em Português.** 2a. Edição. São Carlos: BPLN, 2024.

GUILLOU, P. **pierreguillou/gpt2-small-portuguese**. Hugging Face, 2020. Disponível em: https://huggingface.co/pierreguillou/gpt2-small-portuguese. Acesso em: 12 mar. 2024.

HUGGING FACE. **Transformers: State-of-the-art Machine Learning for Pytorch, TensorFlow, and JAX.** Versão 4.33.0. Brooklyn, NY: Hugging Face, 2024. Disponível em: https://huggingface.co/docs/transformers. Acesso em: 07 mar. 2024.

KAGGLE. **Kaggle: A Machine Learning and Data Science Community**. São Francisco: Kaggle, 2024. Disponível em: https://www.kaggle.com/. Acesso em: 07 mar. 2024.

PEREIRA, Y. H. et al. **Interface em Linguagem Natural para uma Sublinguagem de Câncer de Pele**. Anais do XI Computer on the Beach, v. 11, n. 1, p. 469–474, Camboriú, SC, set. 2020.

PYTHON. **Python**: versão 3.10. Wilmington, DE: Python Software Foundation, 2024. Disponível em: https://www.python.org/. Acesso em: 07 mar. 2024.

ZANINI, H. **Desmistificando o fine tuning de LLMs na prática: PEFT, LoRA, QLORA e Hambúrgueres.** Medium, 13 maio. 2024. Disponível em: https://medium.com/data-hackers/desmistificando-o-fine-tuning-de-llms-na-prática-peft-lora-qlora-e-hambúrgueres-ca6e6008241f. Acesso em 16 jul. 2024.