RECIT
Revista Eletrônica Científica Inovação e Tecnologia
Revista Eletrônica Científica Inovação e Tecnologia
ISSN: 2175-1846

https://periodicos.utfpr.edu.br/recit

Jordan Alexander Machado da Silva
jordan.alexander@unoesc.edu.br
http://orcid.org/0000-0002-3408-1350
Universidade do Oeste de Santa Catarina
UNOESC São Miguel do Oeste

Felipe Gabriel Amado
felipegabrielamado@gmail.com
http://orcid.org/0000-0002-4482-0532
Universidade do Oeste de Santa Catarina
UNOESC São Miguel do Oeste

Conceptron: Framework para desenvolvimento de redes neurais artificiais

RESUMO

A inteligência artificial é um segmento da computação, que possibilita a criação de sistemas capazes de reproduzir a habilidade humana de pensar e resolver problemas. Dentro desta ramificação, existem as redes neurais artificiais, que podem ser definidas como programas, compostos por vários nós, ou neurônios, que são interligados e trocam informações de forma constante. Cada camada desses neurônios executa um algoritmo nos dados processados, atribuindo à rede a capacidade de reconhecer padrões e correlações, agrupar e classificar dados, ou ainda, aprender a realizar alguns tipos de tarefas. Muitas pessoas associam a criação de uma rede neural artificial diretamente com o desenvolvimento de software, o que pode se tornar uma barreira para quem esteja iniciando o aprendizado em inteligência artificial ou algum profissional de outra área, que queira aplicar esses modelos em seus estudos. Este trabalho tem como objetivo apresentar o desenvolvimento de um framework que facilita a criação de modelos de redes neurais artificiais, utilizando uma interface gráfica simples e didática, feita com ReactJS e Node.js, que permite a visualização da representação gráfica da rede neural e a geração do código fonte desse modelo. O código apresentado é escrito em Python e utiliza os algoritmos e abstrações da biblioteca PyTorch, podendo ser utilizado para o aprendizado ou acoplado em outras aplicações.

PALAVRAS-CHAVE: Inteligência Artificial. Redes Neurais. Framework. Python. Node.js.

INTRODUÇÃO

A inteligência artificial é uma ramificação da computação, que tem como objetivo construir sistemas que sejam capazes de simular a habilidade do ser humano de pensar e resolver problemas, e então serem considerados inteligentes. Com a evolução da inteligência artificial, surgiram as redes neurais artificiais, que podem ser definidas como sistemas computacionais compostos por vários nós interligados, semelhantes a um cérebro humano onde existem vários neurônios que também estão interligados, trocando informações. Cada um desses nós, é capaz de executar algoritmos matemáticos nos dados de entrada da rede neural artificial, podendo reconhecer padrões e correlações, agrupar e classificar dados e por último aprender e melhorar sua performance e precisão nas tarefas citadas anteriormente. A aplicação de uma rede neural artificial tem como objetivo auxiliar os profissionais de todos os segmentos a resolver problemas complexos em diversas situações cotidianas.

Muitas pessoas associam a inteligência artificial diretamente com o desenvolvimento de software, criando a falsa afirmação de que para construir qualquer tipo de rede neural artificial, é necessário somente o conhecimento em programação e algoritmos, quando na realidade, além de entender um pouco de programação, é preciso estar familiarizado com alguns conceitos matemáticos, como regressão linear e correlações, além de compreender a área de conhecimento para qual o sistema será concebido.

Atualmente, existem várias coleções de algoritmos e implementações que abstraem a construção de redes neurais artificiais, porém a maioria deles, ainda exigem um conhecimento intermediário em programação o que pode se tornar uma barreira para alguém que esteja iniciando o aprendizado em inteligência artificial ou alguma pessoa que não tenha compreensão na área de desenvolvimento de software e deseje criar uma rede neural artificial.

Pensando nessa situação, evidenciou-se a necessidade da criação de um framework que facilite a construção de redes neurais artificiais de uma forma mais simples e didática. Para isso será necessária a construção de uma interface gráfica didática e intuitiva, o desenvolvimento de uma biblioteca que converta os modelos gráficos em uma implementação em código fonte, do modelo da rede neural artificial, bem como a sua validação, garantindo que o modelo esteja funcionando de acordo com a arquitetura definida pelo usuário.

METODOLOGIA

Antes de iniciar a construção do *framework*, é preciso revisar a literatura para conhecer e compreender os termos e tecnologias que serão trabalhados, e como eles serão aplicados para resolução do problema proposto no início deste trabalho.

1. Referencial teórico

O cérebro humano é a máquina mais poderosa e complexa existente na natureza, sendo capaz de processar uma grande quantidade de informações em um curto período. É formado por uma malha de pequenas unidades responsáveis por armazenar, processar e transmitir informações, esses pequenos microprocessadores orgânicos são chamados de neurônios. Toda essa complexidade e desempenho, culminou na tentativa da criação de um cérebro artificial.

2. O que são redes neurais

De acordo com Haykin (2007) o cérebro é um sistema de processamento de informações altamente complexo e paralelo. Possui a capacidade de ordenar seus microprocessadores orgânicos, os neurônios, de forma a realizar processamentos, muito mais rapidamente do que o mais potente computador existente, e isso não se resume apenas a cérebros humanos, o mesmo ocorre com o cérebro de um cachorro, por exemplo.

Ilustrando essa capacidade, o ser humano possui um sistema de percepção visual que o permite reconhecer o ambiente à sua volta através de imagens enviadas para o cérebro na forma de impulsos nervosos. Todas essas fotografias são processadas dentro de um minúsculo intervalo de 100 a 200 milissegundos, resultando em informações necessárias para a interação com o ambiente, reconhecendo o rosto de pessoas, árvores, carros, animais e outras coisas. O que permite o cérebro humano receber, processar esses dados e transformá-los em informações de uma maneira tão veloz e precisa é a sua plasticidade.

A plasticidade do cérebro é a capacidade que esse órgão possui de remodelar a sua grande estrutura desenvolvendo novas conexões sinápticas através das experiências que vão se acumulando ao longo do tempo. Essa plasticidade é essencial no desenvolvimento dos neurônios e para o processamento de informações no cérebro e é o fator chave na criação de redes neurais artificiais, pois esse fenômeno comprova que o cérebro não é imutável. (FONTES, 2007)

De uma forma mais geral, uma rede neural artificial é um sistema projetado para modelar a maneira como o cérebro humano realiza alguma tarefa específica ou alguma função de interesse, as redes neurais artificiais ou RNAs podem ser implementadas de forma física, através de microprocessadores, ou simuladas pela programação dentro de um ambiente virtual. Para obter um bom desempenho e precisão em seus resultados, as RNAs contam com componentes, chamados de nós, responsáveis por executar uma função de ativação. O conhecimento é adquirido através de um processo de aprendizagem que reforça a conexão entre estes nós.

A rede neural artificial não consegue replicar de uma forma perfeita o funcionamento do cérebro, uma vez que paralelamente e de forma abstrata ele processa uma enorme quantidade de informações, porém quando restringido o cenário a somente uma tarefa, esse modelo funciona de forma satisfatória para representar o conhecimento.

3. Como funcionam as redes neurais

Em 1958 o modelo Perceptron foi introduzido ao mundo da computação, e era capaz de resolver os problemas de inteligência artificial através de uma função de ativação linear, onde os dados de entrada eram colocados em um plano cartesiano e agrupados e classificados por meio de uma reta.

No Perceptron o valor de entrada é multiplicado pelo peso sináptico, e, após isso, somado por cada neurônio que passa, formando um conjunto de entrada. Após formado o conjunto de entrada as informações passam por uma função de ativação, essa função tem como objetivo limitar a amplitude de saída do neurônio, ou seja, o valor de entrada vira um valor normalizado dentro de um intervalo fechado. Existem funções lineares, como a função degrau e identidade, e funções não lineares como a gaussiana, tangente hiperbólica, sigmóide entre outras. Cada tipo de função de ativação resolve um problema específico na RNA, como tarefas de classificação, regressão, predição e outros tipos, depois de executada essa função o neurônio produz um novo valor.

Porém isso ilustra somente os problemas que são possíveis de ser separados de uma forma linear, para os problemas não lineares utilizamos a arquitetura *FeedForward* que possui múltiplas camadas de neurônios e pode ser chamada de MLP, *Multi Layer Perceptron*. Em uma rede MLP cada um dos neurônios recebem os valores de entrada, que são multiplicados pelos pesos sinápticos do neurônio e somados entre si com uma constante de polarização, essa constante é necessária pois os valores são representados em um modelo de plano cartesiano, e é necessário centralizar a curva de função da ativação do neurônio. A soma ponderada gerada após a execução da função de ativação dentro do neurônio determina sua ativação e repassa o seu valor de saída para outros neurônios. Quanto maior for a quantidade de camadas ocultas maior é a complexidade da rede neural artificial. Ainda é possível utilizar uma função de ativação que existe somente na última camada chamada *SoftMax*, essa função transforma as saídas em valores probabilísticos, comumente utilizados em problemas de classificação. (GRÜBLER, 2018)

Tendo conhecimento de como os neurônios processam as informações, é possível definir com mais facilidade qual tipo de função de ativação uma rede neural vai utilizar para resolver de uma forma eficiente o problema proposto em sua criação.

Framework para o desenvolvimento de redes neurais artificais

Levando em consideração os componentes de uma rede neural artificial e como ela funciona, chegou-se ao conceito do desenvolvimento de um *framework* para a modelagem de uma rede neural artificial, batizado de Conceptron.

4. Tecnologias utilizadas

Para o desenvolvimento da *front-end* da aplicação foi utilizada a biblioteca declarativa ReactJS, da linguagem JavaScript. O ReactJS apresenta alta eficiência e facilidade na criação de interfaces de usuários, uma vez que proporciona um desenvolvimento mais limpo e rápido, pois todos os componentes podem ser reaproveitados, além disso, a aplicação fica mais organizada facilitando a sua escalabilidade. Por fim, essa biblioteca implementa o Virtual DOM, onde o estado de cada componente fica salvo e, após analisar as mudanças do componente, procura um caminho menos custoso para atualizar a árvore DOM (*Document Object Model*), aumentando consideravelmente o desempenho da aplicação. (ROVEDA, 2020)

Na criação do modelo visual da rede neural, foi utilizado o componente Neural Network Visualizer, criado pelo desenvolvedor Kevin Scott. Este elemento é responsável por criar de forma visual um modelo da rede neural, de acordo com o número de neurônios em cada camada.

O *back-end* foi desenvolvido utilizando Node.js, e baseia-se em uma API (Interface de Programação de Aplicativos), responsável por localizar, salvar e excluir as informações do projeto no banco de dados. Essa API retorna para a aplicação (*front-end*) um arquivo .json, contendo todas as informações do modelo, que então é carregado na área de trabalho do Conceptron.

Para guardar as informações dos projetos modelados, foi empregado o sistema de banco de dados PostgreSQL, porque possui uma vasta parcela de recursos que armazenam e escalam com segurança independentemente da quantidade de dados, e trabalha de uma forma bastante estável. Ainda conforme Medeiros (2017), “Possui mecanismos de bloqueio, suporta tamanhos ilimitados de linhas, bancos de dados e tabelas, concede segurança contra falhas (caso tenha um pique de luz, por exemplo), entre tantas outras qualidades.”.

A biblioteca responsável por realizar a conversão do modelo construído através da interface gráfica em um código fonte, foi escrita utilizando as implementações de outra biblioteca, chamada de PyTorch. O principal trunfo da PyTorch é a sua acessibilidade, pois possui uma documentação ampla e acessível, e uma vasta comunidade de desenvolvedores melhorando e adicionando novas funcionalidades a cada dia. Ainda, em comparação a biblioteca TensorFlow, não apresenta tantos problemas de desempenho, sendo ideal para o aprendizado de redes neurais artificiais. (INTERESSADO... 2020)

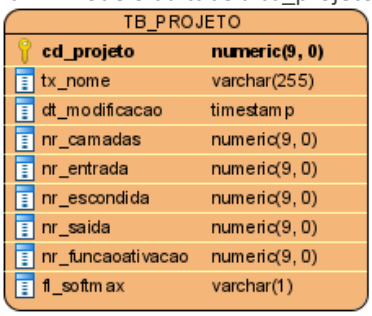
5. Arquitetura e funcionamento

A arquitetura do Conceptron é dividida em três partes: banco de dados, API e aplicação. A seguir os itens são apresentados de forma detalhada.

6. Banco de dados

O sistema possui uma única tabela em seu banco de dados, e ela é responsável por representar o projeto de uma rede neural modelada dentro do sistema.

Figura 1 - Modelo da tabela tb_projeto.



Fonte: Os autores (2021).

De acordo com a Figura 01, cada coluna representa:

cd_projeto: identificador do projeto.

tx_nome: nome do projeto.

dt_modificacao: data de modificação ou criação do projeto.

nr_camadas: número de camadas escondidas.

nr_entrada: número de neurônios na camada de entrada.

nr_escondida: número de neurônios nas camadas escondidas.

nr_saida: número de neurônios na camada de saída.

nr_funcaoativacao: número identificador da função de ativação utilizada pelos neurônios da camada escondida.

fl_softmax: flag que define se a rede neural utiliza a função SoftMax nos neurônios de saída.

Inicialmente o projeto possuía mais tabelas relacionadas ao cadastro de usuários e áreas de trabalho, porém, após o *feedback* dos testadores, o sistema de autenticação foi removido, pois grande parte das pessoas que testaram, apontaram como desnecessário e moroso o processo de cadastro, justificando que iriam acessar a plataforma pontualmente. A manipulação desses dados é realizada pela API, conforme o item a seguir.

7. API

A API, ou *back-end*, é responsável por fazer a comunicação da aplicação com o banco de dados, localizando, atualizando, salvando ou excluindo as redes neurais modeladas. A API foi desenvolvida utilizando Node.js, e apresenta as seguintes funções:

getProjetos: Responsável por localizar todos os projetos salvos no banco de dados, que serão carregados na tela principal do Conceptron.

getProjetosPorId: Utilizada para carregar as informações específicas de um projeto salvo. Essas informações são utilizadas para montar o modelo visual da rede neural.

createProjeto: Executada para salvar um novo projeto no banco de dados.

updateProjeto: Atualiza as informações de um projeto já salvo.

deleteProjeto: Exclui um projeto salvo.

Ambas as funções getProjetos e getProjetosPorId retornam um arquivo json para aplicação que apresenta a seguinte estrutura:

Figura 2 - Estrutura do arquivo json retornado pela API.


```
[
  {
    "id": 1,
    "tx_nome": "Felipe Amado",
    "dt_modificacao": null,
    "nr_camadas": "3",
    "nr_entrada": "3",
    "nr_escondida": "5",
    "nr_saida": "2",
    "nr_funcaoativacao": "3",
    "fl_softmax": "1"
  }
]
```

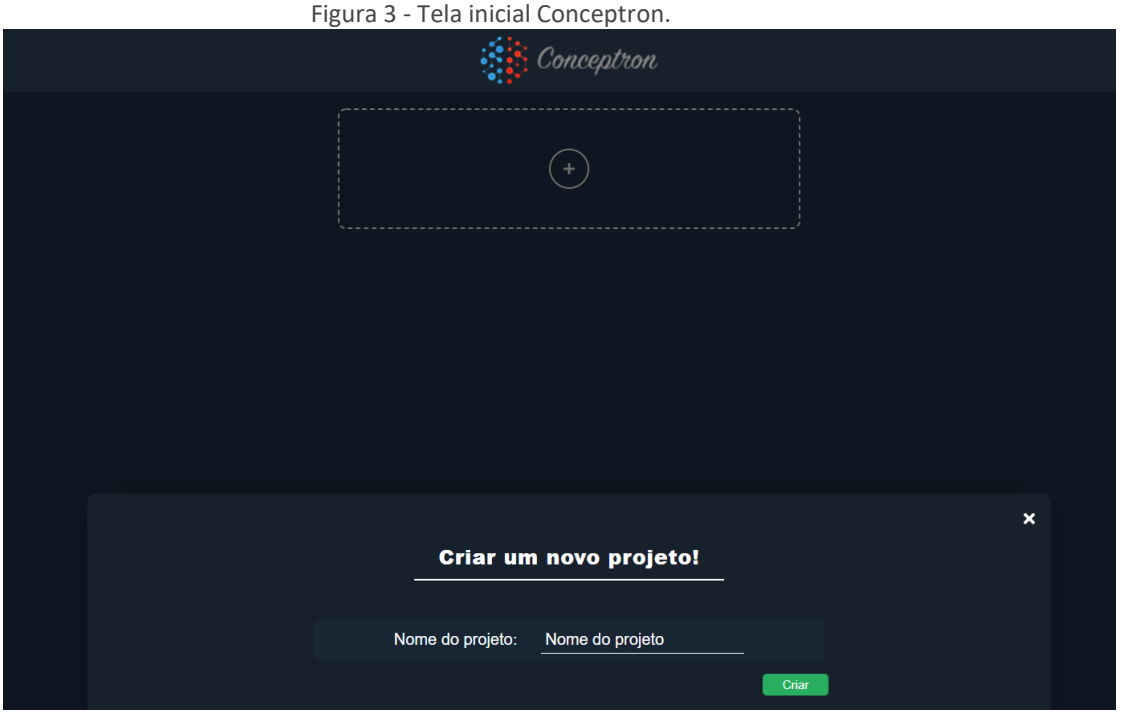

Fonte: Os autores (2021).

Os campos do arquivo json, são um espelho das colunas criadas no banco de dados, conforme descrito na seção API.

8. Aplicação

A aplicação, ou *front-end*, foi desenvolvido utilizando ReactJS e é responsável pela criação, edição, exclusão dos projetos, bem como a geração do código fonte da rede neural artificial.

Figura 3 - Tela inicial Conceptron.



Fonte: Os autores (2021).

Conforme a Figura 03, ao clicar no botão adicionar, representado pelo sinal de mais (+), o usuário deve informar um nome ao projeto e confirmar, após isso, será criado um item na tela inicial. Ao clicar no botão editar, será aberta a tela que permite efetuar a modelagem da rede neural artificial.

Figura 4 - Opções da interface de modelagem.

Fonte: Os autores(2021).

A interface de modelagem é apresentada na Figura 04 e as opções no topo representam:

Neurônios de entrada: Permite informar os neurônios da camada de entrada.

Camadas escondidas: Permite informar o número de camadas de neurônios escondidos.

Neurônios escondidos: Permite informar os neurônios da camada escondida.

Neurônios de saída: Permite informar os neurônios da camada de saída.

Função de ativação: Permite selecionar a função de ativação utilizada na camada escondida.

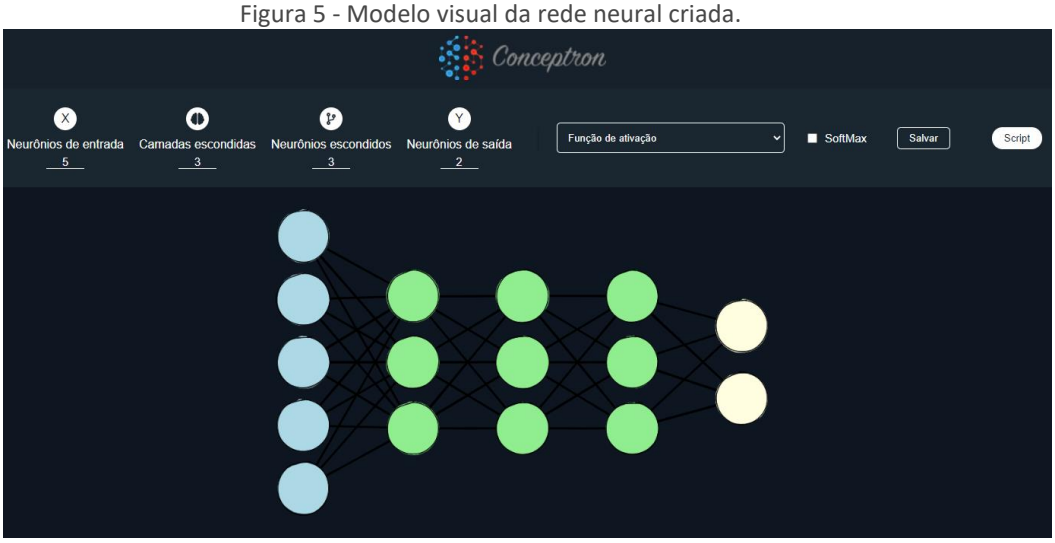
SoftMax: Permite definir se os neurônios da camada de saída irão utilizar a função SoftMax.

Salvar: Salvar e sair.

Script: gerar código fonte em Python da rede neural.

Para efetuar a modelagem de uma rede neural, apenas é necessário informar a quantidade de neurônios em cada uma das camadas, após isso definir o tipo de função de ativação e por fim selecionar se os neurônios de saída irão utilizar a função *SoftMax*, depois de finalizada a modelagem basta clicar no botão salvar. Depois de salvar o modelo, a página será atualizada e apresentará o modelo visual da rede neural.

Figura 5 - Modelo visual da rede neural criada.



Fonte: Os autores (2021).

Para gerar o código fonte em Python, basta clicar no botão Script. Na Figura 06 segue um exemplo de um código fonte gerado pela biblioteca.

Figura 6 - Código fonte da rede neural gerado pelo Conceptron.

Fonte: Os autores (2021).

Além das implementações da rede neural, a biblioteca também adiciona uma função ajuda, que ilustra como é feita a utilização da rede neural artificial.

RESULTADOS

O Conceptron foi testado por dez pessoas, sendo quatro delas da área de tecnologia da informação e seis de outras áreas. Como ponto positivo foi ressaltada a facilidade na modelagem e criação de redes neurais, uma vez que o usuário necessita apenas informar a quantidade de neurônios que cada camada irá utilizar, bem como selecionar a função de ativação dos neurônios e se a camada de saída irá aplicar a função SoftMax. O modelo visual da rede neural criada também facilita o entendimento de como os neurônios estão interligados trocando informações.

Como ponto de melhoria, alguns testadores ressaltaram que não gostariam de realizar o cadastro na aplicação, porque acreditam que o framework será utilizado apenas para criar a rede neural, e com o código fonte Python em mãos, o modelo não seria mais visualizado. Também, houveram sugestões, para exibir o código fonte na própria aplicação, sem a necessidade de baixar o arquivo, pois assim, quem estiver modelando, pode apenas copiar o código fonte da rede neural. Por fim, alterou-se a descrição do botão de geração do script de “Visualizar” para “Script”, pois acabava confundindo algumas pessoas, passando a impressão que esse botão iria exibir o modelo visual da rede neural ao invés de exibir o código fonte gerado. O framework está disponível para acesso através do endereço https://conceptron.herokuapp.com.

CONCLUSÕES

Os modelos de redes neurais artificiais que existem hoje não conseguem replicar o funcionamento do cérebro humano em sua totalidade, porém, quando os aplicados em uma tarefa específica, sendo ela de baixa ou alta complexidade, são capazes de realizar diversas funções, resolvendo problemas que antes eram tidos como impossíveis de serem solucionados.

A codificação de uma rede neural artificial não é extremamente complexa, pois as bibliotecas disponíveis hoje no mercado abstraem uma grande parte de suas implementações, todavia alguns conceitos de desenvolvimento de software ainda são extremamente necessários na sua criação. Durante o desenvolvimento deste framework, foi possível criar uma interface gráfica simples e didática, onde os neurônios podem ser desenhados, interligados e configurados com a função de ativação que melhor se encaixa no problema a ser resolvido. E após isso tudo, o usuário ainda tem a possibilidade de gerar um código fonte, escrito utilizando a linguagem Python, enxuto e funcional, que representa a rede neural artificial do modelo proposto. O framework desenvolvido está disponível para acesso via web, utilizando o endereço https://conceptron.herokuapp.com/.

Partindo desta afirmação, conclui-se que quanto mais fácil o aprendizado de redes neurais artificiais, maior é o número de pessoas que irão começar a estudar e contribuir para o avanço desse segmento da computação, permitindo que essa tecnologia avance cada vez mais com passos mais largos.

O desenvolvimento deste trabalho agregou de forma muito expressiva a vida profissional e acadêmica dos seus realizadores, pois proporcionou uma oportunidade de colocar à prova todos os conhecimentos adquiridos ao longo dos quatro anos de curso de ciências da computação, além de permitir, uma pequena experimentação de como funciona o segmento acadêmico, de pesquisar e produzir ciência.

Para uma futura evolução, idealiza-se implementar no Conceptron, a possibilidade de executar as próprias redes neurais na aplicação, permitindo o usuário a acompanhar em tempo real e de forma visual como a rede neural está processando as informações, além de permitir a entrada de dados de fontes externas e direcionar a saída dos dados para uma API que pode se interligar com outras aplicações.

Conceptron: Framework for development of artificial neural networks

ABSTRACT

Artificial intelligence is a segment of theory that enables the creation of systems capable of reproducing the human ability to think and solve problems. Within this branch, there are artificial neural networks, which can be defined as programs, composed of several nodes, or neurons, which are interconnected and exchange information constantly. Each neuron layer performs an algorithm on the processed data, giving the network an ability to recognize patterns and correlations, group and classify data, or even learn to perform some types of tasks. Many people associate the creation of an artificial neural network directly with software development, which can become a barrier for those starting to learn artificial intelligence or any professional in another area who wants to apply these models in their studies. This work aims to present the development of a framework that facilitates the creation of models of artificial neural networks, using a simple and didactic graphical interface, made with ReactJS and Node.js, which allows the visualization of the graphical representation of the neural network and the generation of the source code of this model. The code presented is written in Python and uses the algorithms and abstractions of the PyTorch library, which can be used for learning or coupled with other applications.

KEYWORDS: Artificial Intelligence. Neural networks. Structure. Python. Node.js.

REFERÊNCIAS

FONTES, Maria Alice. O que é plasticidade neural? 2007. Disponível em: http://www.plenamente.com.br/artigo.php?FhIdArtigo=73. Acesso em: 01 jun. 2021.

GRÜBLER, Murillo. Entendendo o funcionamento de uma Rede Neural Artificial. 2018. Disponível em: https://medium.com/brasil-ai/entendendo-o-funcionamento-de-uma-rede-neural-artificial-4463fcf44dd0. Acesso em: 05 jun. 2020.

HAYKIN, Simon. Redes Neurais: princípios e prática. 2. ed. Porto Alegre: Bookman, 2007. 900 p.

INTERESSADO em machine learning? Melhor aprender PyTorch. 2020. Disponível em: https://cio.com.br/carreira/interessado-em-machine-learning-melhor-aprender-pytorch/. Acesso em: 01 jul. 2021.

MEDEIROS, Alex. Conheça 5 motivos para usar PostgreSQL. 2017. Disponível em: https://blog.configr.com/qualidades-postgresql/. Acesso em: 01 jul. 2021.

ROVEDA, Ugo. React: o que é, como funciona e porque usar e como aprender. 2020. Disponível em: https://kenzie.com.br/blog/react/. Acesso em: 01 jul. 2021.

Recebido: 05/07/2021
Aprovado: 30/12/2023.
DOI: 10.3895/recit.v14i35 14485
Como citar: SILVA, J. A.M. AMADO, F. G. Conceptron: Framework para desenvolvimento de redes neurais artificiais R. Eletr. Cient. Inov. Tecnol, Medianeira, v. 14. n. 35, p. 32-44, set/dez 2023 Disponível em: <https://periodicos.utfpr.edu.br/recit>. Acesso em: XXX.

Correspondência:
Jordan Alexander Machado da Silva
Universidade do Oeste de Santa Catarina - UNOESC São Miguel do Oeste.