CENTRO REGIONAL UNIVERSITÁRIO DE E. S. DO PINHAL

UNIPINHAL

Curso de Engenharia da Computação

Arthur Elidio da Silva

Leonardo Oliveira de Freitas

Projeto e implementação de um sistema inteligente para coleta e análise de sintomas de pacientes em unidades de pronto-atendimento

Espírito Santo do Pinhal - SP

2019

CENTRO REGIONAL UNIVERSITÁRIO DE E. S. DO PINHAL

UNIPINHAL

Curso de Engenharia da Computação

Arthur Elidio da Silva

Leonardo Oliveira de Freitas

Projeto e implementação de um sistema inteligente para coleta e análise de sintomas de pacientes em unidades de pronto-atendimento

Projeto de Pesquisa apresentado para avaliação da disciplina Projeto Integrado I do Curso de Engenharia da Computação do Centro Regional Universitário de Espírito Santo do Pinhal. Orientador: Prof. Dr. José Tarcísio Franco de Camargo.

Espírito Santo do Pinhal - SP

2019

SUMÁRIO

1. Introdução

Atualmente, em qualquer unidade de saúde, existe um processo para o atendimento do paciente a ser respeitado. Inicialmente o mesmo deve passar pela etapa de triagem, que consiste em um sistema de seleção, coleta de dados e sintomas, sendo a seguir classificado de acordo com seu risco vital, a partir da análise dos sintomas apresentados. Uma vez feita a análise, o paciente é classificado de acordo com a sua respectiva urgência vital e, a partir desse momento, o paciente é encaminhado à espera para o tratamento do médico responsável. Essa espera é relacionada com a classificação atribuída, podendo chegar a até mais de 4 (quatro) horas.

O objetivo proposto pelo sistema em desenvolvimento é construir um software utilizando conhecimentos em inteligência artificial e lógica nebulosa que seja capaz de contribuir para a solução da latência do processo de coleta de sintomas e classificação de pacientes, tornando assim tanto a triagem como a pós-triagem mais ágeis.

2. Fundamentos Teóricos

Para o desenvolvimento do projeto foi necessário um estudo sobre a área da saúde, especificamente a parte de triagem, coleta de sintomas e tratamento de pacientes, para que a partir do conhecimento obtido fossemos capazes de desenvolver um sistema fuzzy apto a replicar o comportamento humano, sendo assim, capaz de escolher uma classificação acurada do paciente a partir de seus sintomas apresentados.

2.1. Conhecimentos na área da saúde

Após estudos em campo e conteúdos já publicados, foram obtidas as seguintes informações a respeito do tratamento de pacientes em postos de pronto atendimento. A triagem tem como objetivo classificar o risco vital do paciente a partir de sintomas coletados, tais como, pressão, pulso, respiração, temperatura, glicemia capilar, peso, saturação e outros. Uma vez feita a análise de sintomas, o paciente é classificado de acordo com o seu grau de urgência vital seguindo o padrão de classificação de manchester.

De acordo com a classificação de manchester, o paciente pode ser classificado em 5 graus diferentes, são eles: Emergência, Muita Urgência, Urgente, Pouco Urgente e Não Urgente, os quais são distinguidos a partir das cores vermelho, laranja, amarelo, verde e azul, respectivamente.

Cada tipo de classificação contém uma previsão de atendimento, seguindo a classificação de coloração vermelha até azul. São estas as previsões de atendimento: imediato, em até 20 minutos, em até 60 minutos, em até 120 minutos e em até 240 minutos. A figura 1 ilustra esta situação.

É importante ressaltar que a triagem hospitalar tem como objetivo obter informações do paciente, tais como a dados pessoais, coleta de sintomas e classificação do paciente e não realizar diagnóstico final do paciente.

	Prioridade	COR	TEMPO
1	Emergente	Vermelho	0 minutos
2	Muito Urgente	Laranja	10 minutos
3	Urgente	Amarelo	60 minutos
4	Pouco Urgente	Verde	120 minutos
5	Não Urgente	Azul	240 minutos

Figura 1 - Classificação de manchester

2.2. Inteligência Artificial

O estudo em inteligência artificial tem como objetivo replicar o comportamento humano contemplando a capacidade cognitiva, reconhecimento de contexto e tomada de decisão. Ou seja, um “agente”, em uma determinada situação, para ser considerado inteligente, deve ser capaz de analisar, compreender e realizar uma tomada de decisão. (NORVIC; RUSSELL, 2010)

Para o desenvolvimento da Inteligência Artificial em questão foi utilizado o seguinte conhecimento.

2.2.1. Lógica Fuzzy

O conhecimento em lógica fuzzy consiste em uma lógica multi-valorada, capaz de capturar informações vagas, em geral descritas em uma linguagem natural, e convertê-las para um formato numérico, de fácil manipulação pelos computadores atuais. Ou seja, ela permite modos de raciocínio aproximados e não exatos, podendo assim criar um conjunto de regras, onde, a partir de uma função de pertinência é possível criar uma transição gradual da “não-verdade” para a “verdade”. É apresentada na figura 2 a diferença da lógica booleana aristotélica, baseada em “verdadeiro” ou “falso”, e a lógica fuzzy, capaz de admitir vários valores (CAVALCANTI et al., 2012).

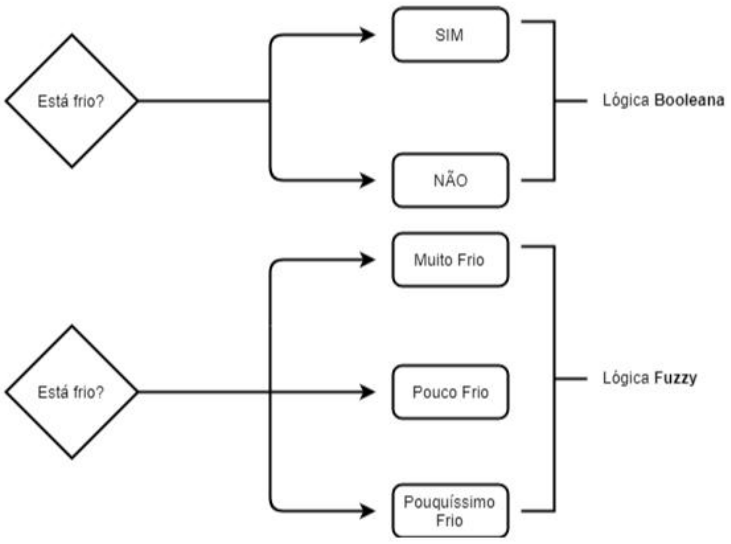

Figura 2 - Diferença da Lógica Booleana para Lógica Fuzzy

2.3. Serviços Web

De acordo com Pereira (2016), a evolução tecnológica da sociedade humana tem proporcionado a criação de sistemas apoiados em “serviços web”. Um serviço web consiste na manipulação de informações de forma centralizada, permitindo que estas sejam tratadas separadamente da interface, a qual será utilizada apenas para a visualização pelo usuário final.

Portanto, para trabalhar com a disposição dos dados aplicaremos o modelo Rest, que é uma forma de arquitetura para organização de serviços web.

2.3.1. Modelo REST

O modelo REST (Representational State Transfer) representa a arquitetura atual para criação de serviços web. Nesse modelo é utilizada a semântica dos métodos HTTP (GET, POST, PUT e DELETE), o que torna esse padrão de envio de dados mais simples, leve e dinâmico.

3. Materiais e Métodos

3.1. Implementação camada serviços web

Para o desenvolvimento da camada de serviços estamos utilizando a IDE Visual Code, com a linguagem de programação JavaScript e Framework Express, que é o core para criação dos serviços**.**

3.1.1. Execução dos Serviços Web

Para execução dos serviços web será utilizado o servidor Node, que se encarregará do processo de execução da aplicação, para que fiquem visíveis os serviços para consumo da camada de interface.

3.2. Interface para demonstração do processo de triagem

Será utilizada a biblioteca React do JavaScript para desenvolvimento da interface de visualização do processo de triagem. Essa biblioteca facilita a criação páginas web que contenham interação com usuário entre outros benefícios.

3.3. Armazenamento da triagem com banco de dados MongoDB

MongoDB é um banco de dados não relacional, o qual será usado para guardar o processo de triagem e informações do paciente, além de conter um histórico para futuras consultas.

3.4. Implementação Lógica Fuzzy

Para o desenvolvimento do sistema em questão será utilizada a linguagem Python na IDE Spyder, disponível através do programa Anaconda Navigator, que contém várias outras IDEs para desenvolvimento em Python. O Spyder foi escolhido pois disponibiliza de forma simplificada o uso das bibliotecas Scikit-Fuzzy e Numpy.

3.5. NumPy

NumPy é um pacote para a linguagem Python que suporta vetores e matrizes multidimensionais, possuindo uma larga coleção de funções matemáticas para trabalhar com estas estruturas.

3.6. Scikit-Fuzzy

Scikit-Fuzzy é uma biblioteca que contém uma coleção de algoritmos lógicos escrito em Python para o uso e implementação da lógica fuzzy. Algumas das principais funções utilizadas.

3.6.1. Antecedent e Consequent

São funções utilizadas para demonstrar quais são os parâmetros de entrada e saída do sistema fuzzy criado. Para utilizá-los basta importar a biblioteca **skfuzzy** e passar seus parâmetros, que são um vetor e um label que irá ser usado de rótulo posteriormente. Segue um exemplo da utilização na figura 3.

```
import skfuzzy as fuzz
from skfuzzy import control as ctrl

vital_alter = ctrl.Antecedent(np.arange(0, 11, 1), 'Dados Vitais Alterados')
dor = ctrl.Antecedent(np.arange(0, 11, 1), 'Dor')
nuca = ctrl.Antecedent(np.arange(0, 11, 1), 'Nuca')
nausea = ctrl.Antecedent(np.arange(0,11,1), 'Nausea')
mental_alter = ctrl.Antecedent(np.arange(0,11,1), 'Alteração Estado Mental')
enxaqueca = ctrl.Antecedent(np.arange(0,11,1), 'Enxaqueca')
manchester = ctrl.Consequent(np.arange(0, 6, 1), 'Manchester')
```

Figura 3 - Exemplo Antecedent e Consequent

3.6.2. Membership functions

As membership functions, também conhecidas por funções de pertinência, tem como meta determinar quão verdadeiro é o valor de entrada dentro do universo criado. Por assim dizer, podemos afirmar que toda função de pertinência está associada a uma variável que define o quanto aquela entrada pertence ao conjunto criado.

A biblioteca skfuzzy já disponibiliza algumas funções de pertinência por padrão, cada uma delas tem um funcionamento específico, entradas e saídas diferentes.

Segue abaixo algumas das funções de pertinência usadas:

3.6.2.1. PieceMF

É uma função linear normalmente utilizada para filtros críticos. Para a utilização dessa função é necessário passar como parâmetro um vetor que será utilizado como universo e um outro vetor, de dimensão três, que será utilizado para definição dos pontos da função. Na figura 4 está um exemplo da implementação.

```
nuca['nao_rigida'] = fuzz.piecemf(nuca.universe, [0, 8, 8])
nuca['rigida'] = fuzz.piecemf(nuca.universe, [8, 10, 10])
```

Figura 4 - Implementação PieceMF

3.6.2.2. TrimMF

É uma função triangular. Para utilização dessa função é necessário passar como parâmetro um vetor que será utilizado como universo e um outro vetor de tamanho três que será utilizado para definição dos pontos da função. Na figura 5 é apresentado um exemplo da implementação.

```
manchester['vermelho'] = fuzz.trimf(manchester.universe, [0, 0, 3])
manchester['amarelo'] = fuzz.trimf(manchester.universe, [0, 3, 6])
manchester['verde'] = fuzz.trimf(manchester.universe, [3, 6, 6])
```

Figura 5 - Implementação TrimMF

3.6.3. Rule, ControlSystem e Compute

Nesse tópico será demonstrado como definir as regras que irão controlar e computar os valores do sistema fuzzy criado. Uma regra tem como intuito relacionar os antecedentes e consequentes (ver tópico 2.4.3.), criando assim uma interação entre os valores para que seja definido o resultado. Isto é, definimos que dado uma certa relação entre os “antecedentes” o resultado será tal “consequente”.

Uma vez criadas as regras, definimos o sistema que irá utilizá-las e, logo após, podemos utilizá-lo para computar uma dada entrada de valor no sistema.

A função “rule” tem como entrada a relação dos antecedentes com operadores lógicos e qual o resultado esperado.

A função “controlSystem” tem como entrada um vetor contendo as regras criadas posteriormente, nos retornando um sistema criado a partir da mesma.

A função “compute” não recebe uma entrada, porém utiliza o sistema criado com as regras para nos dar o resultado esperado.

Na figura 6 está um exemplo de como implementar as funções.

```
rule1 = ctrl.Rule(vital_alter['poor'] | vital_alter['good'] | dor['intensa'], manchester['vermelho'])
rule2 = ctrl.Rule(nuca['rigida'], manchester['amarelo'])
rule3 = ctrl.Rule(vital_alter['average'] | dor['nao_intensa'], manchester['verde'])

manchester_ctrl = ctrl.ControlSystem([rule1, rule2, rule3])

classificar = ctrl.ControlSystemSimulation(manchester_ctrl)

classificar.input['Dados Vitais Alterados'] = 5
classificar.input['Dor'] = 5
classificar.input['Nuca'] = 9

classificar.compute()
```

Figura 6 - Implementação rule, controlSystem e compute

4. Resultados Esperados

Pretende-se demonstrar que o uso da inteligência artificial é capaz de ajudar na área da saúde. Dessa forma, espera-se que o software venha a ser utilizado em hospitais e centros de pronto-atendimento, otimizando o processo de triagem e diminuindo o tempo de espera dos pacientes. Espera-se também que a coleta de informações nesse processo possa ser disponibilizada, respeitando-se as restrições legais, para outros estudos na área da saúde.

5. Referências Bibliográficas

NORVIC, Peter; RUSSUL, Stuart. **Inteligência Artificial**. [S. l.: s. n.], 2010.

BERNARDO, Elisângela Maria de Souza et al. **Procedimento Operacional Padrão Classificação de Risco**. Espírito Santo do Pinhal: [s. n.], 2017-2018.

PEREIRA, Caio Ribeiro. **Construindo APIs REST com Node.js**. [S. l.: s. n.], 2016.

NODEJS. **Node.js v9.11.2 Documentation**. Disponível em: https://nodejs.org/docs/latest-v9.x/api/ Acesso em: 21 de maio de 2019.

React. **Getting Started**. Disponível em: https://reactjs.org/docs/getting-started.html Acesso em: 21 de maio de 2019.

Express. **Roteamento**. Disponível em: https://expressjs.com/pt-br/guide/routing.html Acesso em: 21 de maio de 2019.

SKFUZZY. **Api Reference**. Disponível em: https://pythonhosted.org/scikit-fuzzy/api/api.html Acesso em: 26 de maio de 2019.

CAVALCANTI, José Homero Feitosa et al. **Lógica Fuzzy Aplicada Às Engenharias**. João Pessoa PB: [s. n.], 2012. Disponível em: http://www.logicafuzzy.com.br/wp-content/uploads/2013/04/logica_fuzzy_aplicada_as_engenharias.pdf. Acesso em: 22 maio 2019.

6. Cronograma de Atividades

	Atividades TCC	Data de vencimento	Concluído	Atribuído a	Status
2	Estudar Classificação de triagem hospitalar			Leonardo/Arthur	Em andamento
3	Estudar Logica Fuzzy			Leonardo/Arthur	Em andamento
4	~~Reunião Terça-Feira 26/02 com a Coordenadora Gisele~~	~~26/02/19~~	~~TRUE~~	~~Leonardo/Arthur~~	~~Concluído~~
5	~~Reunião Quinta-Feira 07/03 com a Coordenadora Gisele~~	~~07/03/19~~	~~TRUE~~	~~Leonardo/Arthur~~	~~Concluído~~
6	~~Criar Repositório GIT~~	~~28/02/19~~	~~TRUE~~	~~Leonardo/Arthur~~	~~Concluído~~
7	~~Criar Diagrama de processo CAHD~~	~~31/03/19~~	~~TRUE~~	~~Leonardo/Arthur~~	~~Concluído~~
8	Buscar Informações nos Hospitais e Posto de Saúde	31/05/19		Leonardo/Arthur	Em andamento
9	~~Arquitetura Interface React~~	~~31/03/19~~	~~TRUE~~	~~Arthur~~	~~Concluído~~
10	Desenvolver Interface React	31/07/19		Arthur	Em andamento
11	Criar Algoritmo Fuzzy	31/07/19		Leonardo/Arthur	Em andamento
12	Projeto Pesquisa TCC	27/05/19		Leonardo/Arthur	Em andamento
13	Modelar Base de Dados(MongoDB)	31/07/19		Leonardo	Não iniciado
14	Criar WebApi CRUD de Triagem	31/07/19		Leonardo	Não iniciado
15	Criar WebApi da IA	31/07/19		Leonardo	Não iniciado
16	Escrever o TCC	30/08/19		Leonardo/Arthur	Não iniciado
17	Entregar TCC	20/11/19		Leonardo/Arthur	Não iniciado
18	Apresentação do TCC	02/12/19 - 03/12/19		Leonardo/Arthur	Não iniciado

Figura 7 - Cronograma

_____

Orientador: Prof. Dr. José Tarcísio Franco de Camargo

_____

Aluno: Arthur Elídio da Silva

RA: 150271

_____

Aluno: Leonardo Oliveira de Freitas

RA: 150332