

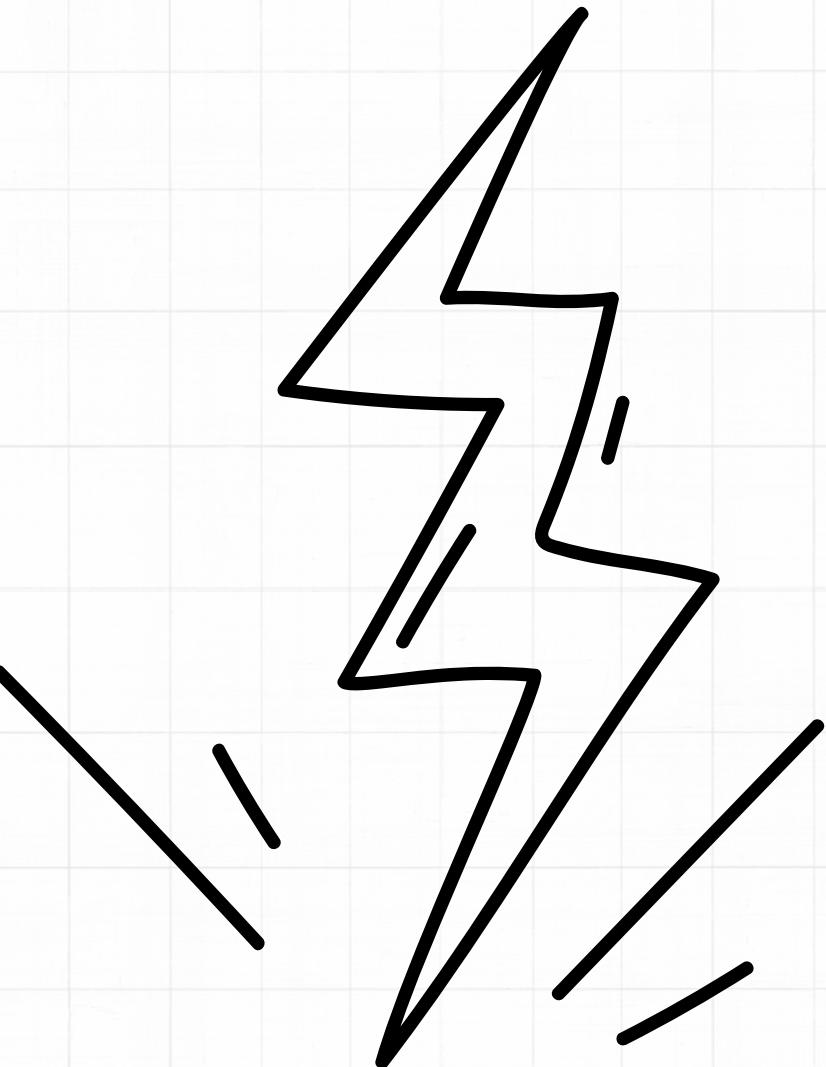


MINDMATE TEAM

→ EQUIPE ✓

FORMADA PELOS ALUNOS(AS)

- Pedro Henrique Bellone
- Pedro Henrique Gaioso
- Vitor Alexandre Moreira Amaral



IS CANVAS

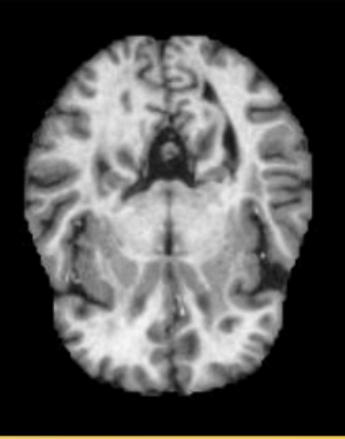


APLICAÇÃO DE INTELIGÊNCIA ARTIFICIAL

Detector de Alzheimer do MindMate 

nonDem0.jpg

Imagen

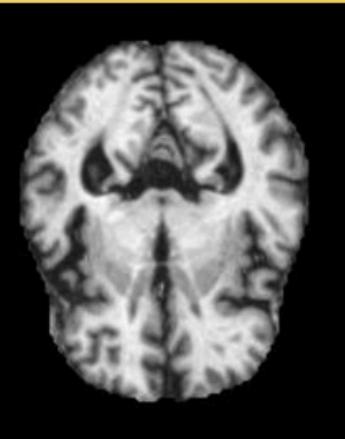


Resultados
Nenhum traço de Alzheimer encontrado

Detector de Alzheimer do MindMate 

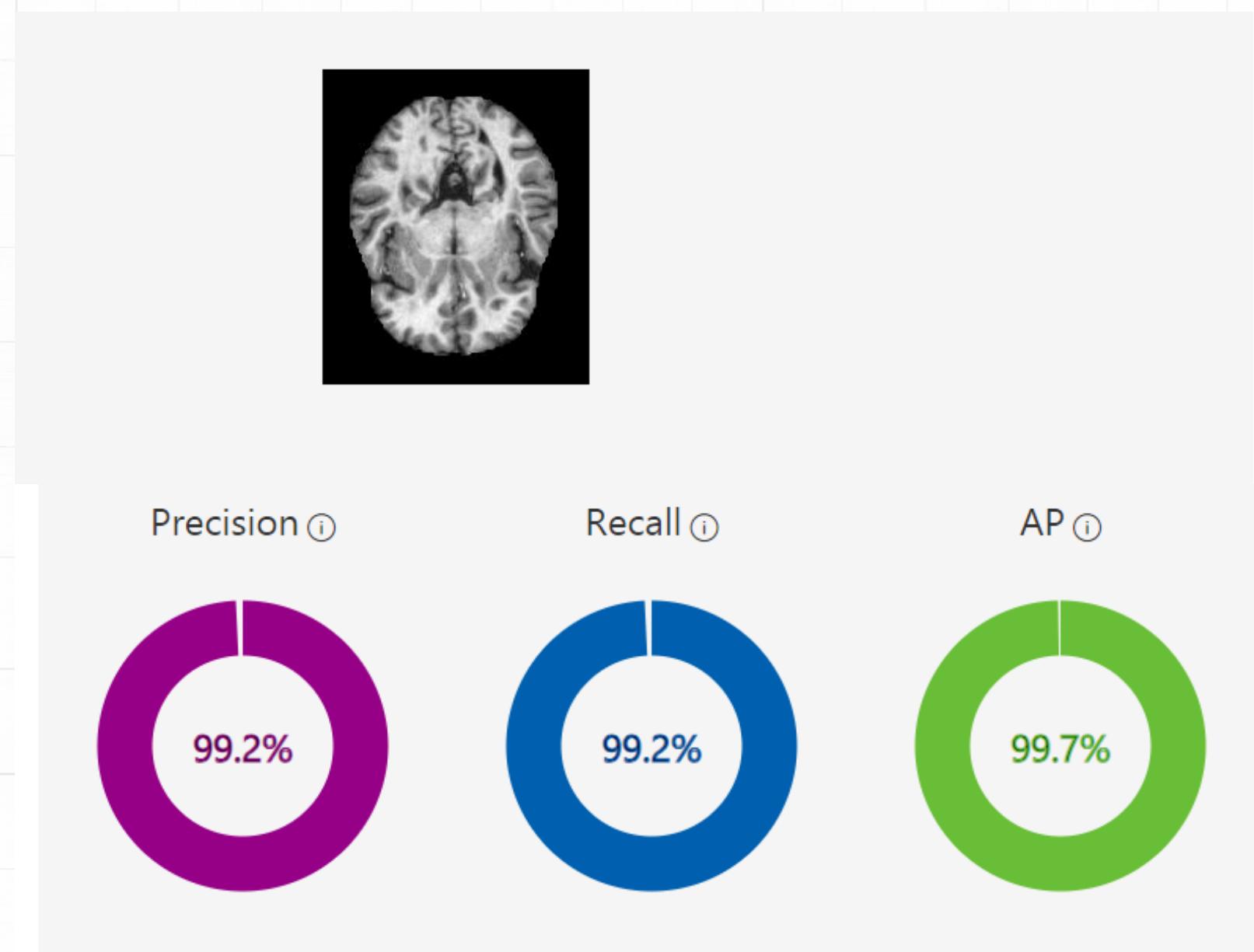
moderateDem1.jpg

Imagen



Resultados
Você possui Alzheimer, favor procurar um médico

APLICAÇÃO DE INTELIGÊNCIA ARTIFICIAL



Performance Per Tag

Tag	Precision	Recall	A.P.	Image count
Normal	99.4%	99.8%	99.8%	2560
Alzheimer	88.9%	72.7%	86.3%	52

APLICAÇÃO DE INTELIGÊNCIA ARTIFICIAL

```
let imageLoaded = false;
$("#image-selector").change(function () {
  imageLoaded = false;
  let reader = new FileReader();
  reader.onload = function () {
    let dataURL = reader.result;
    $("#selected-image").attr("src", dataURL);
    $("#prediction-list").empty();
    imageLoaded = true;
  }

  let file = $("#image-selector").prop('files')[0];
  reader.readAsDataURL(file);
});

let model;
let modelLoaded = false;
$(document).ready(async function () {
  modelLoaded = false;
  $('.progress-bar').show();
  console.log("Loading model...");
  model = await tf.loadGraphModel('model/model.json');
  console.log("Model loaded.");
  $('.progress-bar').hide();
  modelLoaded = true;
});

$("#predict-button").click(async function () {
  if (!modelLoaded) { alert("The model must be loaded first"); return; }
  if (!imageLoaded) { alert("Please select an image first"); return; }

  let image = $('#selected-image').get(0);

  console.log("Loading image...");
  let tensor = tf.browser.fromPixels(image, 3)
    .resizeNearestNeighbor([224, 224])
    .expandDims()
    .toFloat()
    .reverse(-1);
  let predictions = await model.predict(tensor).data();
  console.log(predictions);
  let top5 = Array.from(predictions)
    .map(function (p, i) {
      return {
        probability: p,
        className: TARGET_CLASSES[i]
      };
    });
});
```

```
TARGET_CLASSES = [
  1: "Normal",
  0: "Alzheimer"
];
```



TensorFlow

CÓDIGO DE BANCO DE DADOS

```
package app;

import static spark.Spark.*;
import service.MedicoService;
import service.QuestoesService;
import service.UsuarioService;
import java.util.HashMap;
import spark.Filter;
import spark.Request;
import spark.Response;
import spark.Spark;

public class Aplicacao {

    private static UsuarioService usuarioService = new UsuarioService();
    private static MedicoService medicoService = new MedicoService();
    private static QuestoesService questoesService = new QuestoesService();

    Run|Debug
    public static void main(String[] args) {
        port(6789);
        staticFiles.location("/public");
        CorsFilter.apply();
    }

    options("/*", (request, response) -> {

        String accessControlRequestHeaders = request.headers("Access-Control-Request-Headers");
        if (accessControlRequestHeaders != null) {
            response.header("Access-Control-Allow-Headers", accessControlRequestHeaders);
        }

        String accessControlRequestMethod = request.headers("Access-Control-Request-Method");
        if (accessControlRequestMethod != null) {
            response.header("Access-Control-Allow-Methods", accessControlRequestMethod);
        }

        return "OK";
    });

    //USUARIOS
    post("/usuario	insert", (request, response) -> usuarioService.insert(request, response));
    put("/usuario/put/:id", (request, response) -> usuarioService.update(request, response));
    get("/usuario/get", (request, response) -> usuarioService.get(request, response));
}
```

```
//USUARIOS
post("/usuario/insert", (request, response) -> usuarioService.insert(request, response));
put("/usuario/put/:id", (request, response) -> usuarioService.update(request, response));
get("/usuario/get", (request, response) -> usuarioService.get(request, response));
get("/usuario/getAll", (request, response) -> usuarioService.getAll(request, response));

post("/usuario/authenticate", (request, response) -> usuarioService.authenticate(request, response));

//MEDICO
post("/medico/insert", (request, response) -> medicoService.insert(request, response));
get("/medico/get/:id", (request, response) -> medicoService.get(request, response));

//PERGUNTAS E RESPOSTAS
get("/questoes/getAll", (request, response) -> questoesService.get(request, response));
```

MEDICO DAO

```
package dao;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

import model.Medico;

public class MedicoDAO extends DAO {
    public MedicoDAO() {
        super();
        conectar();
    }

    public void finalize() {
        close();
    }

    public boolean insert(Medico medico) {
        boolean status = false;
        try {

            String sql = "INSERT INTO medico (desc_medico, fone_medico, email_medico, idade_medico, cidade_medico, endereco_medico, web_medico, id_usuario) "
                    + "VALUES ('" + medico.getDescMedico() + "', ''"
                    + medico.getFoneMedico() + "", "'"
                    + medico.getEmailMedico() + "", "'"
                    + medico.getIdadeMedico() + "", "'"
                    + medico.getCidadeMedico() + "", "'"
                    + medico.getEnderecoMedico() + "", "'"
                    + medico.getWebMedico() + "", "'"
                    + medico.getIdUsuario() + ");";

            PreparedStatement st = conexao.prepareStatement(sql);
            st.executeUpdate();
            st.close();
            status = true;
        } catch (SQLException u) {
            throw new RuntimeException(u);
        }
        return status;
    }
}
```

QUESTOES DAO

```
package dao;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.sql.Statement;  
import java.util.ArrayList;  
import java.util.Collections;  
import java.util.List;  
  
import model.Questoes;  
import model.Resposta;  
  
public class QuestoesDAO extends DAO {  
    public QuestoesDAO() {  
        super();  
        conectar();  
    }  
  
    public void finalize() {  
        close();  
    }  
  
    public List<Questoes> getAll() {  
        List<Questoes> questoes = new ArrayList<>();  
  
        try {  
            Statement st = conexao.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY);  
            String sql = "SELECT * FROM perguntas";  
            ResultSet rs = st.executeQuery(sql);  
            while (rs.next()) {  
                Questoes questao = new Questoes(  
                    rs.getInt(columnLabel:"id_perguntas"),  
                    rs.getString(columnLabel:"desc_perguntas")  
                );  
                questoes.add(questao);  
            }  
  
            for (Questoes q : questoes) {  
                ArrayList<Resposta> respostas = new ArrayList<Resposta>();  
                sql = "SELECT * FROM respostas WHERE id_perguntas = " + q.getId_pergunta() + ";"  
                rs = st.executeQuery(sql);  
  
                while (rs.next()) {  
                    Resposta resposta = new Resposta(  
                        rs.getInt(columnLabel:"id_respostas"),  
                        rs.getString(columnLabel:"nota_respostas"),  
                        rs.getDouble(columnLabel:"valor")  
                    );  
                    respostas.add(resposta);  
                }  
                q.setRespostas(respostas);  
            }  
        } catch (SQLException u) {  
            throw new RuntimeException(u);  
        }  
        return questoes;  
    }  
}
```

```
        respostas.add(resposta);  
    }  
    q.setRespostas(respostas);  
}  
  
Collections.shuffle(questoes);  
st.close();  
} catch (SQLException u) {  
    throw new RuntimeException(u);  
}  
return questoes;  
}  
}
```

QUESTOES DAO

```
package dao;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.sql.Statement;  
import java.util.ArrayList;  
import java.util.Collections;  
import java.util.List;  
  
import model.Questoes;  
import model.Resposta;  
  
public class QuestoesDAO extends DAO {  
    public QuestoesDAO() {  
        super();  
        conectar();  
    }  
  
    public void finalize() {  
        close();  
    }  
  
    public List<Questoes> getAll() {  
        List<Questoes> questoes = new ArrayList<>();  
  
        try {  
            Statement st = conexao.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY);  
            String sql = "SELECT * FROM perguntas";  
            ResultSet rs = st.executeQuery(sql);  
            while (rs.next()) {  
                Questoes questao = new Questoes(  
                    rs.getInt(columnLabel:"id_perguntas"),  
                    rs.getString(columnLabel:"desc_perguntas")  
                );  
                questoes.add(questao);  
            }  
  
            for (Questoes q : questoes) {  
                ArrayList<Resposta> respostas = new ArrayList<Resposta>();  
                sql = "SELECT * FROM respostas WHERE id_perguntas = " + q.getId_pergunta() + ";"  
                rs = st.executeQuery(sql);  
  
                while (rs.next()) {  
                    Resposta resposta = new Resposta(  
                        rs.getInt(columnLabel:"id_respostas"),  
                        rs.getString(columnLabel:"nota_respostas"),  
                        rs.getDouble(columnLabel:"valor")  
                    );  
                    respostas.add(resposta);  
                }  
                q.setRespostas(respostas);  
            }  
        } catch (SQLException u) {  
            throw new RuntimeException(u);  
        }  
        return questoes;  
    }  
}
```

```
        respostas.add(resposta);  
    }  
    q.setRespostas(respostas);  
}  
  
Collections.shuffle(questoes);  
st.close();  
} catch (SQLException u) {  
    throw new RuntimeException(u);  
}  
return questoes;  
}  
}
```

BANCO DE DADOS

	id integer	desc_medico character varying	fone_medico bpchar	email_medico character varying	idade_medico integer	cidade_medico character varying	endereco_medico character varying	web_medico character varying	id_usuario integer
1	1	Dr. Gabriel Silva	12345678901	joao.silva@example.com	45	São Paulo	Rua das Flores, 123	www.joaosilva.com	1
2	4	Jucelino	31994343515	jucelino@gmail.com	100	Não informado	Não informado	https://picsum.photos/200/300	3
3	5	Vitor	31993539134	vitor@gmail.com	100	Não informado	Não informado	https://ibb.co/dk9fk08	5
4	6	Joao	31993539134	vitor@gmail.com	100	Não informado	Não informado	https://ibb.co/dk9fk08	6
5	7	Vitor	31999999999	jorge@gmail.com	100	Não informado	Não informado	https://ibb.co/dk9fk08	6
6	8	Joaquim Medico	31999407892	joaquimmedicina23@gmail.com	100	Não informado	Não informado	https://picsum.photos/200/300	7
7	9	Pedro Medico	31999407892	pedromedico@gmail.com	100	Não informado	Não informado	https://picsum.photos/200/300	7

	id integer	nome_completo character varying	email character varying	senha character varying
1	1	root	root@gmail.com	132475956107784875457507977471906551877
2	3	default	default@gmail.com	339743357143909072213173867436258735946
3	4	Vitor	Vitor Amaral	299132688689127175738334524183350839358
4	5	Vitor	vitor@gmail.com	336183414858843401077361536652139902809
5	6	Joao	joaobastos@gmail.com	336183414858843401077361536652139902809
6	7	VitorTeste	Vitorteste@gmail.com	252147601056529534498625177464897506607

	id_perguntas integer	desc_perguntas character varying
1	1	Você esquece acontecimentos ou informações importantes?
2	2	Você tem dificuldade para realizar tarefas habituais?
3	3	Você tem dificuldade de linguagem ou para encontrar as palavras certas?
4	4	Você já teve algum episódio de desorientação em relação ao tempo ou ao lugar?
5	5	Você tem variações de humor e comportamento?

	id_respostas integer	id_perguntas integer	nota_respostas character varying	valor integer
1	1	1	Sim, esqueço bastante	1
2	2	1	Muitas vezes, esqueço	2
3	3	1	Poucas vezes, esqueço	3
4	4	1	Nunca, esqueço	4
5	5	2	Sim, tenho dificuldade	1
6	6	2	Muitas vezes, tenho dificuldades	2
7	7	2	Parcialmente, tenho poucas dificuldades	3
8	8	2	Não, não tenho nenhuma dificuldade	4
9	9	3	Sim, tenho dificuldade	1
10	10	3	Raramente, as vezes encontro, as vezes não	2
11	11	3	Poucas vezes, tenho pouca dificuldade	3
12	12	3	De jeito nenhum, sempre acho as palavras certas	4
13	13	4	Sim, sempre sofro de desorientação	1
14	14	4	Muitas vezes, tenho desorientação	2
15	15	4	Raramente, me sinto desorientado	3
16	16	4	Não, me sinto desorientado em nenhuma hora	4
17	17	5	Sim, tenho um grande transtorno de humor	1
18	18	5	As vezes, tenho alguns transtornos de humor	2
19	19	5	Poucas vezes, tenho transtornos de humor	3
20	20	5	Não tenho nenhum transtorno de humor	4

CÓDIGO NO GIT

this is GitHub

GITHUB