Universidade Católica Dom Bosco (UCDB)
Engenharia da Computação

Sistema de Apoio ao Diagnóstico Médico Baseado em Lógica Fuzzy

João Henrique Schweitzer Rezende
Luan Henrique Santos Miranda

Campo Grande – MS
2025

João Henrique Schweitzer Rezende
Luan Henrique Santos Miranda

Sistema de Apoio ao Diagnóstico Médico Baseado em Lógica Fuzzy

Memorial Descritivo apresentado ao Curso de Engenharia da Computação da Universidade Católica Dom Bosco.

Campo Grande – MS
2025

Conteúdo

1 Resumo

Este memorial descreve o desenvolvimento e a arquitetura de um **sistema de apoio ao diagnóstico médico baseado em lógica fuzzy**. O software, desenvolvido em **Python**, avalia o risco clínico de um paciente por meio de três variáveis de entrada: **febre**, **tosse** e **saturação de oxigênio**.

A partir dessas informações, o motor de inferência fuzzy — implementado com a biblioteca *scikit-fuzzy* — calcula um índice de risco. A aplicação possui interface gráfica desenvolvida em **PyQt5**, permitindo a visualização das funções de pertinência, o uso de diferentes conjuntos de regras e o registro de histórico de diagnósticos. O sistema demonstra a viabilidade da aplicação prática de lógica fuzzy em ferramentas de apoio à decisão clínica.

Palavras-chave: Lógica Fuzzy; Sistema Especialista; Apoio ao Diagnóstico; Python; PyQt5; scikit-fuzzy.

2 Introdução

A tomada de decisão clínica é frequentemente caracterizada pela incerteza e pela imprecisão, onde informações como a intensidade de uma tosse ou o grau de febre são descritas de forma vaga (e.g., "tosse moderada"). A **Lógica Fuzzy** oferece um arcabouço matemático robusto para modelar e gerenciar esse tipo de imprecisão linguística, tornando-a ideal para a criação de sistemas especialistas em diagnóstico.

Este trabalho visa desenvolver uma ferramenta interativa que utilize a lógica fuzzy para calcular um índice de risco clínico. O objetivo principal é facilitar a avaliação de sintomas por meio de uma interface gráfica intuitiva, promovendo a aplicação prática dos fundamentos da lógica fuzzy em um contexto real de apoio à decisão médica.

3 Estrutura do Software

O projeto é estruturado em módulos claros que separam a lógica de inferência, a interface de usuário e as utilidades de dados.

3.1 Escolha das Bibliotecas

Para a implementação do sistema, foram selecionadas bibliotecas que garantem eficiência no cálculo fuzzy e na construção da interface:

- **Python:** Linguagem principal para todo o desenvolvimento.
- ***scikit-fuzzy* e *numpy*:** Implementação do motor de inferência fuzzy, universos de discurso e Funções de Pertinência.

- **PyQt5:** Desenvolvimento da Interface Gráfica do Usuário (GUI) para um ambiente visual moderno.
- **Matplotlib:** Geração dinâmica dos gráficos das funções de pertinência para visualização.
- **FPDF:** Exportação de relatórios de diagnóstico em formato PDF (`pdf_exporter.py`).
- **json / pathlib:** Gerenciamento e armazenamento do histórico de diagnósticos (`history_manager.py`).

3.2 Interface Gráfica e Interação com o Usuário

A interface é construída em **PyQt5** e utiliza o *styling* **QSS** (`ui_main.py`) para um design contemporâneo. A aplicação é organizada em abas funcionais:

- **Aba Diagnóstico:** Recebe as entradas clínicas e permite a seleção do conjunto de regras (*ruleset*).
- **Aba Gráficos Fuzzy:** Exibe os gráficos de pertinência de todas as variáveis.
- **Aba Regras Fuzzy:** Permite a consulta textual dos conjuntos de regras.
- **Aba Histórico / Exportar:** Lista os diagnósticos realizados e permite a gestão do histórico.

3.3 Inserção e Validação dos Dados de Entrada

O sistema requer três variáveis de entrada numéricas fornecidas pelo usuário:

Variável	Unidade	Universo de Discurso
Febre	°C	35.0 a 41.1
Tosse	Escala 0 a 10	0 a 11
Saturação	%	70 a 101

O módulo principal (`main.py`) valida as entradas, aceitando vírgula ou ponto como separador decimal e verificando se os valores estão dentro dos limites do universo de discurso definido no motor fuzzy.

3.4 Motor de Inferência Fuzzy e Cálculo de Risco

O motor de inferência (`diagnostico_fuzzy.py`) define a lógica central do sistema:

- **Antecedentes:** `febre` (*normal, moderada, alta*), `tosse` (*leve, moderada, forte*), `saturação` (*boa, moderada, baixa*).
- **Consequente:** `risco` (*baixo, medio, alto*) na escala de 0% a 100%.
- **Regras:** O sistema suporta a seleção de múltiplos *rulesets* (Respiratória, Viral e Bacteriana), permitindo flexibilidade diagnóstica.

A função `calcular_risco` executa a simulação fuzzy e retorna o valor de risco **defuzzificado** e as regras que foram acionadas (com seu grau de ativação).

3.5 Exibição Visual e Relatório

Após o cálculo, o sistema exibe o valor do **Índice de Risco** e as regras mais ativadas. Uma janela pop-up apresenta os mini-gráficos das funções de pertinência (`fuzzy_plotter.py`), destacando o valor de entrada e o risco resultante. A funcionalidade de **Exportar Relatório (PDF)** gera um documento consolidado com os dados de entrada, o risco final e as figuras dos gráficos.

4 Resultados

O software demonstrou **robustez e usabilidade**. A validação de entrada eficaz impediu cálculos com dados inconsistentes. O suporte a múltiplos *rulesets* garante a versatilidade do sistema. A **visualização dinâmica** dos gráficos e das regras ativadas é uma característica didática importante, que facilita a compreensão do processo de inferência fuzzy. As funcionalidades de **Histórico** e **Exportação para PDF** complementam a ferramenta, garantindo a rastreabilidade e utilidade prática dos diagnósticos.

5 Discussão

A abordagem fuzzy provou ser altamente adequada para modelar sistemas de apoio ao diagnóstico que lidam com a incerteza intrínseca dos dados clínicos. A arquitetura modular do projeto simplifica a manutenção e a expansão. A capacidade de selecionar diferentes conjuntos de regras permite que o sistema seja facilmente adaptado para outros protocolos de diagnóstico sem a necessidade de reescrever a infraestrutura. A transparência do processo de inferência, reforçada pela exibição das regras ativadas e dos gráficos, é crucial para a aceitação da ferramenta em um contexto clínico.

6 Conclusão

O desenvolvimento do Sistema de Apoio ao Diagnóstico Médico baseado em Lógica Fuzzy alcançou com sucesso os objetivos propostos. A ferramenta é funcional, didática e demonstra a aplicação prática dos conceitos de Lógica Fuzzy em um sistema especialista.

Melhorias futuras podem incluir a inclusão de um editor gráfico de funções de pertinência, suporte a mais variáveis de entrada/doenças e a implementação de um mecanismo de otimização para ajuste automático dos parâmetros fuzzy.

7 Tempo Gasto

Estima-se que o desenvolvimento do projeto foi realizado em cerca de **2 semanas** de trabalho.

7.1 Distribuição do Tempo:

Atividade	**Horas Estimadas**
Pesquisa e Aprendizado (Lógica Fuzzy / scikit-fuzzy)	$\sim$ 10 horas
Construção do Código (Motor Fuzzy, UI, Utilitários)	$\sim$ 30 horas
Testes e Debugging	$\sim$ 5 horas
Interface Gráfica (PyQt5 e QSS)	$\sim$ 6 horas
Documentação	$\sim$ 5 horas

8 Modo de Uso

Pré-requisitos

- Acesso ao terminal (shell) no sistema onde a aplicação será executada.
- Conexão à internet (para baixar Miniconda e dependências, caso necessário).
- Os arquivos do projeto devem estar numa pasta, por exemplo: `/MeusProjetos/Fuzzy`, contendo:
  - `setup_env.sh`
  - `run_fuzzy.sh`
  - a pasta `app/` com `main.py` e demais arquivos da aplicação.

Instalação (primeira execução)

Execute os comandos abaixo no terminal a partir da pasta do projeto:

```
# Navegar até a pasta do projeto
cd ~/MeusProjetos/Fuzzy

# Dar permissão de execução ao instalador
chmod +x setup_env.sh

# Executar o instalador
./setup_env.sh
```

O script `setup_env.sh` realiza:

1. Verificação/instalação do Miniconda.
2. Criação do ambiente conda chamado `Fuzzy`.
3. Instalação das dependências (PyQt5, numpy, scipy, matplotlib, scikit-fuzzy, fpdf, etc.).
4. Criação do lançador `run_fuzzy.sh` e de um atalho `.desktop`.
5. Tentativa de iniciar a aplicação ao final da instalação.

Execução (após a instalação)

Existem duas formas simples de iniciar a aplicação:

Pelo terminal

```
cd ~/MeusProjetos/Fuzzy
./run_fuzzy.sh
```

Pelo menu do sistema (se o atalho .desktop foi criado) Procure no menu de aplicativos por "Fuzzy Diagnóstico" e execute o atalho gerado.

Solução de problemas comuns

- **Permissão negada:** rode `chmod +x run_fuzzy.sh` ou `chmod +x setup_env.sh`.
- **Comando `conda` não encontrado:** feche e reabra o terminal após a instalação do Miniconda; certifique-se de que `$HOME/miniconda3/bin` está no `PATH`.
- **Atalho não aparece no menu:** execute

```
update-desktop-database ~/.local/share/applications || true
```

- **Erros de dependência:** verifique a saída do instalador; você pode ativar manualmente o ambiente e reinstalar pacotes:

```
conda activate Fuzzy
pip install --upgrade PyQt5 scikit-fuzzy fpdf
```

9 Demonstração do Software

Esta seção apresenta a demonstração prática do software através de um vídeo exemplar. O objetivo desta demonstração é ilustrar de forma clara e concisa a usabilidade da aplicação e as principais funcionalidades implementadas.

Assista ao vídeo para ver o funcionamento do software.

Disponivel em: https://www.youtube.com

10 Referências

- LAÉCIO, C. B. Introdução à Lógica Fuzzy. Instituto de Matemática, Estatística e Computação Científica – UNICAMP. Disponível em: `https://www.ime.unicamp.br/~laeciocb/programa_ms580_segusem2008.pdf`. Acesso em: 2025.
- MÁRIO, R. Lógica Fuzzy – Notas de aula. COPPE/UFRJ. Disponível em: `https://www.cos.ufrj.br/~mario/logica/logicaFuzzy.pdf`. Acesso em: 2025.
- PISTORI, H. Introdução à Lógica Fuzzy – Aula 1. YouTube, 2020. Disponível em: `https://www.youtube.com/watch?v=_WxmxdtH8fY`. Acesso em: 2025.
- PISTORI, H. Sistemas Fuzzy – Aula 2. YouTube, 2020. Disponível em: `https://www.youtube.com/watch?v=ZLLbZLcCTcI`. Acesso em: 2025.
- SCITKIT-FUZZY. Biblioteca Python para sistemas fuzzy. Disponível em: `https://pythonhosted.org/scikit-fuzzy/`. Acesso em: 2025.
- PYQT5. Toolkit para desenvolvimento de interfaces gráficas em Python. Documentação oficial: `https://doc.qt.io/qtforpython/`. Acesso em: 2025.
- MATPLOTLIB. Biblioteca para visualização de dados em Python. Disponível em: `https://matplotlib.org/`. Acesso em: 2025.
- FPDF. Biblioteca Python para geração de arquivos PDF. Disponível em: `https://pyfpdf.github.io/fpdf2/`. Acesso em: 2025.
- OPENAI. ChatGPT. Disponível em: `https://chat.openai.com`. Acesso em: 2025.
- GITHUB. GitHub Copilot. Disponível em: `https://github.com/features/copilot`. Acesso em: 2025.