



INSTITUTO TECNOLÓGICO DE AERONÁUTICA
CONSELHO NACIONAL DE DESENVOLVIMENTO CIENTÍFICO E
TECNOLÓGICO - CNPq



**PROGRAMA INSTITUCIONAL DE BOLSAS DE INICIAÇÃO
CIENTÍFICA - PIBIC**

**Sistema de marcação de Futebol de Robôs
Soccer 3D usando aprendizado de máquina
guiado por conhecimento humano**

Isabelle Ferreira de Oliveira

RELATÓRIO PARCIAL DE ATIVIDADES

Orientador: Edgar Toshiro Yano

Março / 2019



INSTITUTO TECNOLÓGICO DE AERONÁUTICA
CONSELHO NACIONAL DE DESENVOLVIMENTO CIENTÍFICO E
TECNOLÓGICO - CNPq



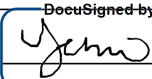
PROGRAMA INSTITUCIONAL DE BOLSAS DE INICIAÇÃO CIENTÍFICA - PIBIC

Relatório Parcial de Atividades

Sistema de marcação de Futebol de Robôs Soccer 3D usando aprendizado de máquina guiado por conhecimento humano

São José dos Campos, 11/03/2019

Nome do aluno	Isabelle Ferraira de Oliveira
Assinatura do aluno	

Nome do orientador	Edgar Toshiro Yano
Assinatura do orientador	 B47745D2C3F241F...

INSTITUTO TECNOLÓGICO DE AERONÁUTICA
PROGRAMA INSTITUCIONAL DE BOLSAS DE INICIAÇÃO CIENTÍFICA -
PIBIC

Formulário de Aprovação de Relatório pelo Orientador

Relatório: ☒ Rel. Parcial ☐ Rel. Final

1- CONSIDERO O RELATÓRIO APROVADO COM BASE NOS SEGUINTE
ASPECTOS

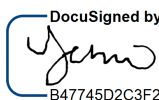
Todos os objetivos propostos foram atendidos

2- APRECIÇÕES DO ORIENTADOR SOBRE O DESEMPENHO DO BOLSISTA NA
EXECUÇÃO DO TRABALHO DE INICIAÇÃO CIENTÍFICA

A bolsista atendeu todos os requisitos esperados na execução do
trabalho de iniciação científica

Local e data: São José dos Campos, 11/03/2019.

Assinatura do Orientador:

DocuSigned by:

B47745D2C3F241F...

Sumário

1	Resumo do Plano Inicial	5
2	Resumo das Atividades Realizadas	6
3	Descrição do Problema	6
4	Resultados Obtidos	9
4.1	Sistema de Marcação Atual	9
4.2	Ferramenta de Interface Gráfica	10
4.2.1	Arquivo de texto (.txt)	10
4.2.2	Classe Frame	11
4.2.3	Classe PositioningFileManager	12
4.2.4	Classe MarkingSystemWidget	12
4.3	Aprendizado de Máquina	13
5	Conclusões	15
6	Agradecimentos	15
7	Bibliografia	16

1 Resumo do Plano Inicial

O objetivo deste trabalho é desenvolver e implementar uma inteligência por meio de aprendizado de máquina ou rede neural que determine quais oponentes devem ser marcados em um determinado momento de partida futebol de robôs humanóides simulados. A confecção das entradas iniciais da rede neural será realizada por conhecimento humano, com o auxílio de uma interface gráfica que permita interagir com os frames da partida.

O contexto é uma partida de futebol segundo a RoboCup, competição de robótica anual internacional, categoria Soccer Simulation 3D (Soccer3D). A partir da posição dos robôs oponentes e da bola, deve-se obter quais jogadores adversários deveriam ou não ser marcados a fim de se evitar levar gols. Visa-se utilizar os resultados aqui alcançados pela equipe da ITAndroids, que representa o ITA, em competições nacionais e internacionais, como na competição Latin America Robotics Competition (LARC)/CBR e na própria RoboCup.

Planejamento Atualizado:

- 1o Bimestre (ago / set): Estudo dos critérios para marcação de oponentes e estudo de algoritmo de aprendizado de máquina.
- 2o Bimestre (out / nov): Estudo de linguagem de ferramenta de interface gráfica e início da implementação da ferramenta de geração de dataset para a inteligência.
- 3o Bimestre (dez / jan): Conclusão da implementação da ferramenta da geração de dataset para a inteligência e confecção do relatório parcial.
- 4o Bimestre (fev / mar): Geração de amostras para a inteligência através da ferramenta gráfica, seleção da estrutura a ser usada na inteligência artificial e implementação inicial, além de primeira análise dos resultados.
- 5o Bimestre (abr / mai): Aprimoração dos resultados da inteligência através de novos dados de entrada, testes para um dataset de teste no qual se tem uma resposta esperada e alterações finais estruturais para a IA após análise dos novos resultados.
- 6o Bimestre (jun / jul): Análise final dos resultados, análise das melhorias do time com a nova estratégia de marcação, confecção do relatório final e redação do artigo para o ENCITA.

2 Resumo das Atividades Realizadas

Ao longo dos dois primeiros meses, foram realizados estudos sobre como a marcação era feita atualmente na equipe ITAndroids e qual seriam os algoritmos de aprendizado de máquina utilizados. O método utilizado se baseava em algumas condições heurísticas referentes aos oponentes.

Foi decidido inicialmente que será usado um algoritmo supervisionado de classificação. Esse algoritmo foi escolhido entre as opções de algoritmos supervisionados de regressão, algoritmos não supervisionados e algoritmos de reforço. Uma introdução a esse assuntos pode ser encontrada em [1]. Uma descrição sobre porque seria o ideal entre essas opções aplicar um algoritmo supervisionado de classificação está apresentada na seção 4.

Ao longo do segundo bimestre, estudou-se quais linguagens se usaria para a ferramenta de interface gráfica. Entre Java e Qt em C++, preferiu-se a última opção, tendo em vista que todo o restante de código da equipe ITAndroids está escrito em C++, inclusive outras ferramentas de interfaces gráficas, facilitando a integração e o suporte técnico. Para fins de simplificação do projeto, escolheu-se também utilizar de arquivos de texto (.txt) para armazenar os dados referentes às posições dos jogadores, oponentes e bola em campo, que serviriam de entrada para a inteligência. Dessa forma, nesse período foi iniciada a implementação da interface.

No terceiro bimestre, concluiu-se a implementação da interface, codificando a lógica de leitura e escrita dos arquivos de texto (.txt), além do desenho do campo e dos jogadores e bola na tela.

3 Descrição do Problema

O sistema de marcação é um processo sequencial usado para marcar jogadores oponentes que estão ofensivamente perigosos durante a partida. No contexto do futebol de robôs humanóides simulados, a marcação de oponentes pode ser muito crucial em uma partida, melhorando significativamente a defesa de um time. A marcação consiste no fato de se implementar um algoritmo que consiga utilizar o planejamento de trajetórias do robô de forma a dificultar que oponentes em situações de vantagem façam gols. Isso traz um desafio: decidir quais jogadores adversários estão em situação mais privilegiada em um determinado instante e deveriam ser marcados por agentes aliados.

Nesse projeto, trabalhou-se com os robôs da competição Soccer Simulation 3D (Soccer3D): uma competição que robôs humanóides competem uns contra os outros em uma simulação realista das regras e da física de um jogo de futebol [2]. Cada time tem 11 jogadores: um goleiro e outros 10 que podem assumir posições dinâmicas, como zagueiro, atacante, etc., a critério de escolha de estratégia das equipes. A Figura 1 mostra uma cena de partida de Soccer3D (nesse exemplo, com apenas 9 jogadores em cada time).



Figura 1: Um frame de uma partida de futebol simulado Soccer3D de 9 vs 9 jogadores [2].

As dimensões do campo de futebol são de 30 por 20 metros. Além disso, o campo é cercado por uma borda de 10 metros nas dimensões x e y, e o espaço além dessa área não é acessível para os jogadores. Uma representação do campo pode ser vista na Figura 2.

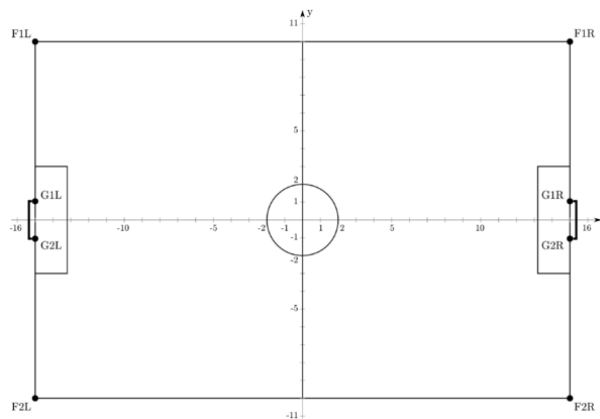


Figura 2: Representação de um campo de futebol simulado para partidas de Soccer3D [2].

O modelo atual de robô utilizado nas competições é baseado no robô *Nao*, da empresa *Aldebaran Robotics*, apresentado na Figura 3. Esse robô possui 22 articulações para controlar o movimento de seu corpo, uma câmera direcional com campo de visão de 120 graus de largura na cabeça, e cabeça que pode girar em dois graus de liberdade, variando de -120 a 120 graus da esquerda para a direita e de -45 a 45 graus de baixo para cima [3].



Figura 3: Robô *Nao*, da empresa *Aldebaran Robotics* [3].

As regras completas podem ser lidas em [4].

A competição utiliza o simulador *Simspark* [5] e o servidor *Rcserver3D* [6] para simular tanto a física, quanto a partida entre os multiagentes robóticos. Os times (computadores *clients*) se conectam ao computador servidor (*server*) através de um *agentproxy* chamdo *MagmaProxy* [7]. Os jogos são executados em *Ubuntu 16.04* de 64 Bits e são exibidos na ferramenta de visualização e monitoramento *Roboviz* (*monitor*) [8]. A Figura 4 a seguir mostra o esquema dessa comunicação.

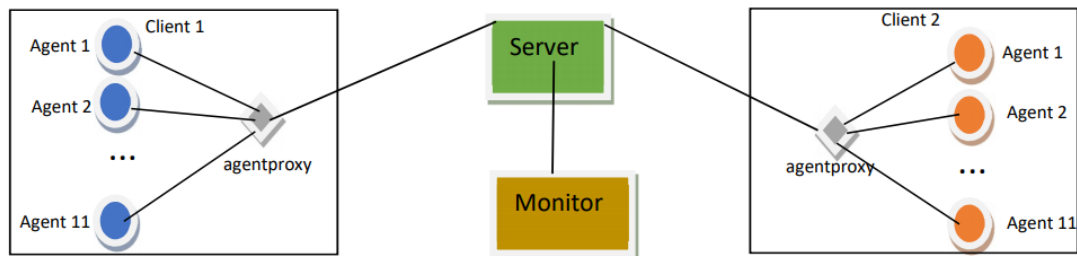


Figura 4: Representação de como o software e hardware do ambiente da competição são utilizados. Existem dois computadores *clients*, um para cada equipe, chamados *client1* e *client2*; um servidor; e um computador monitor (*Roboviz*) [4].

A linguagem utilizada para o projeto da equipe ITAndroids foi C++, pois é uma linguagem que tem uma velocidade de execução alta e tem boa escalabilidade para um projeto grande.

Nesse contexto, surgem várias questões a serem resolvidas para se ter um time vencedor. Nesse projeto, será abordado o seguinte problema: dada as posições dos jogadores oponentes e da bola, deve-se escolher quais adversários são os mais ofensivamente perigosos e deveriam ser marcados.

Grande parte do projeto se trata da ferramenta de interface gráfica que vai captar o conhecimento humano e formar o dataset da inteligência artificial. O intuito é fazer um compilado das ideias inconscientes acerca de quais oponentes deveriam ser marcados, para posteriormente automatizar o processo por meio da inteligência artificial, indo além das condições heurísticas utilizadas atualmente. Para resolver esse problema, então, foi feito nesse primeiro momento a interface em Qt de C++.

4 Resultados Obtidos

Nessa seção, serão apresentados os resultados obtidos do estudo do problema, desde como está a sendo resolvido atualmente pela equipe até uma ideia inicial de substituir a atual resolução do problema por uma com aprendizado de máquina na tomada de decisão para o sistema de marcação pra robôs jogadores de futebol Soccer3D. Além disso, foi explicado a lógica da implementação da ferramenta de interface gráfica.

4.1 Sistema de Marcação Atual

Atualmente na ITAndroids [9], o sistema de marcação é responsável por decidir os jogadores que serão marcados, definir as funções para marcar esses jogadores e, por fim, usa o sistema de Atribuição de Funções (*Role Assignment*) para designar agentes aliados para essas funções definidas.

No primeiro momento, para decidir quais oponentes serão marcados, um método heurístico é atualmente usado com base nas seguintes condições sobre o oponente, isto é, marca-se o oponente caso ele:

- Esteja perto o suficiente para dar um chute a gol;

- Não seja o adversário mais próximo da bola;
- Não esteja muito perto da bola;
- Não esteja muito longe atrás da bola.

Depois disso, um conjunto de agentes do time aliado é selecionado para marcar os adversários escolhidos. Para isso, as posições onde os agentes aliados ficarão para marcar os oponentes são calculadas como 1,5 metros contados a partir do oponente marcado, na reta que o conecta ao centro do gol do time aliado. Um esquema do dessa regra está exposto na Figura 5.

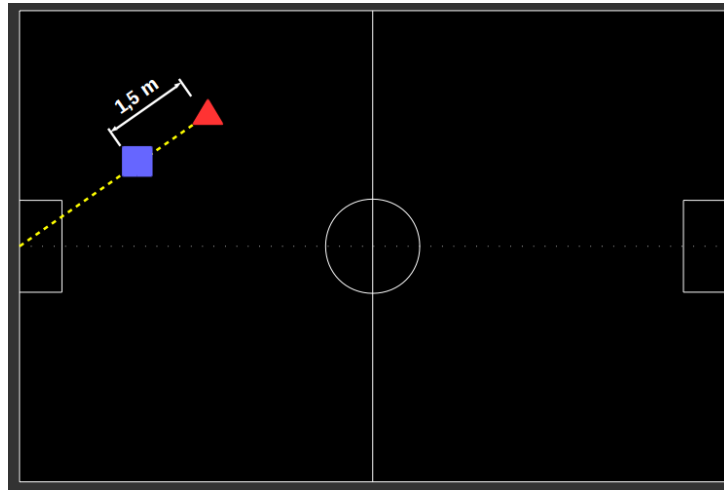


Figura 5: Esquema de como é escolhido a posição de marcação (quadrado azul) para um determinado oponente a ser marcado (representado em triângulo vermelho). O lado esquerdo do campo é o do time aliado.

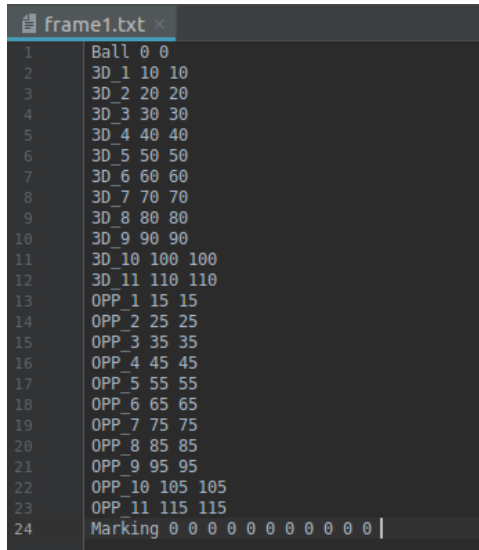
As posições de formação naturais a serem substituídas por posições de marcação são as mais próximas de cada posição de marcação, e sua seleção é feita usando o algoritmo húngaro, que calcula a soma mínima de distâncias entre as posições de formação anteriores e as posições de marcação.

4.2 Ferramenta de Interface Gráfica

4.2.1 Arquivo de texto (.txt)

Os arquivos estão em um diretório próprio chamado /positions para fins de organização e são nomeados como frameX.txt, no qual X é o número do frame ao qual esse arquivo se

refere. Dentro dos arquivos, é seguido o padrão apresentado na Figura 6.



```

1 Ball 0 0
2 3D_1 10 10
3 3D_2 20 20
4 3D_3 30 30
5 3D_4 40 40
6 3D_5 50 50
7 3D_6 60 60
8 3D_7 70 70
9 3D_8 80 80
10 3D_9 90 90
11 3D_10 100 100
12 3D_11 110 110
13 OPP_1 15 15
14 OPP_2 25 25
15 OPP_3 35 35
16 OPP_4 45 45
17 OPP_5 55 55
18 OPP_6 65 65
19 OPP_7 75 75
20 OPP_8 85 85
21 OPP_9 95 95
22 OPP_10 105 105
23 OPP_11 115 115
24 Marking 0 0 0 0 0 0 0 0 0 0 0 |

```

Figura 6: Exemplo de arquivo de texto (.txt) que servia de entrada e saída da interface gráfica. Na figura, está apresentada o frame1.txt.

No arquivo, cada linha se inicia com uma palavra indentificadora, que pode ser Ball, 3D_i, OPP_j e Marking, no qual i e j são os números do agente aliado e do oponente em questão, respectivamente.

Os números após os indentificadores Ball, 3D_i e 3D_j são respectivamente as coordenadas x e y da bola, dos agentes aliados e dos oponentes. Já na última linha, após o identificador Marking, existe 11 inteiros, que podem ser 0 ou 1, indicando se quais oponentes deverão ser marcados naquele frame (1 para marcado e 0 para não marcado).

4.2.2 Classe Frame

Frame é a classe que guarda todas as informações dos agentes e bola em campo em um determinado frame.

Nessa classe, existe um vetor *teammates*, um vetor *opponents* e objeto *ball*, referentes aos agentes aliados, aos jogadores adversários e à bola, respectivamente. Nesses atributos são guardados, por exemplo, suas coordenadas x e y, se determinado agente está ou não caído, se determinado oponente está ou não selecionado, etc.

Sua utilização se dá na classe principal da interface gráfica, na qual existe um vetor de Frames, responsável por manter as informações contidas em todos os frames documentados nos arquivos de texto, que já foram lidos pela classe *PositioningFileManager*.

4.2.3 Classe PositioningFileManager

PositioningFileManager é o nome da classe que lida com a leitura e escrita dos arquivos de texto. Suas principais funções são *populateFrame* e *writeMarkingVector*.

- ***populateFrame***: lê o arquivo de texto e retorna um objeto da classe *Frame* com os dados salvos nesse arquivo, como posições dos agentes em campo, da bola, e vetor de marcação naquele momento.
- ***writeMarkingVector***: essa função é chamada ao botão confirm ser clicado. Ela recebe as marcações feitas pelo usuário na interface e reescreve o vetor após o indicativo *Marking* no arquivo de texto referente àquele frame.

4.2.4 Classe MarkingSystemWidget

MarkingSystemWidget é a classe principal da interface gráfica, que herda da *QMainWindow*, da *Framework Qt*.

Nessa classe, além de outros atributos referentes a detalhes mais internos, existe um vetor *frames* (que guarda todos os frames criados pela classe *PositioningFileManager* ao ler os arquivos da pasta /positions) e um inteiro *currentFrame* (utilizado para que a interface saiba qual frame deve exibir em sua tela).

No construtor dessa classe também é criado todo o design da tela da interface. Essa tela possui os seguintes 4 botões:

- prevFrame;
- nextFrame;
- unmarkAll;
- confirm.

Ao clicar nos botões prevFrame e nextFrame, a tela é atualizada para exibir os 22 jogadores e a bola nesse determinado frame selecionado. O botão unmarkAll desseleciona todos os oponentes que o usuário tinha selecionado para marcar naquele frame em questão. Por fim, o botão confirm chama a classe PositioningFileManager e escreve no arquivo de texto (.txt) quais os jogadores oponentes que deveriam ser marcados naquela situação de jogo.

Dessa forma, a aparência da interface está como representada na Figura 7.

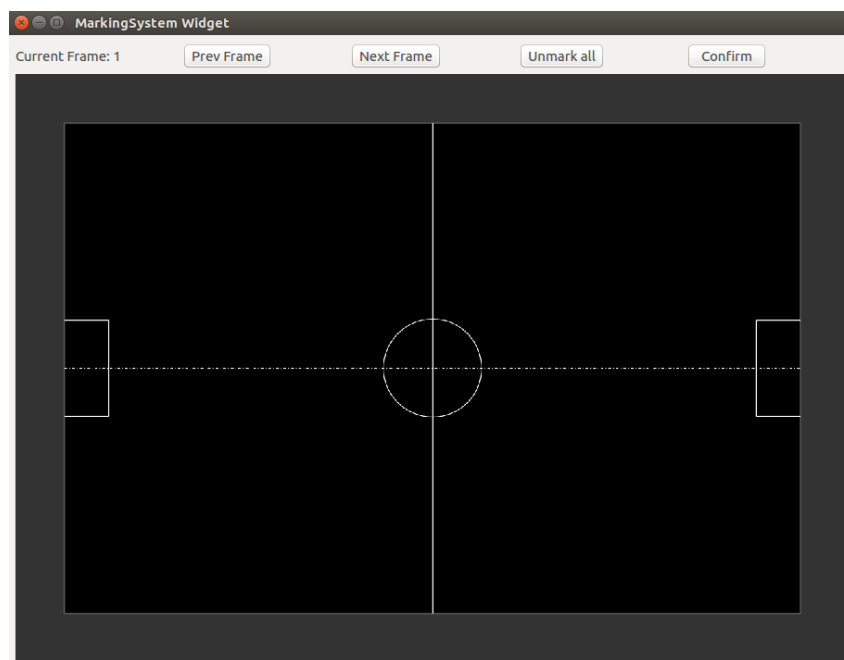


Figura 7: Aparência da ferramenta de interface gráfica Marking System Widget.

4.3 Aprendizado de Máquina

Aprendizado de máquina (ou *Machine Learning*, em inglês), de forma simplificada, é um método de análise de dados que automatiza o processo de aprender a partir de dados de entrada, identificar padrões e tomar decisões com o mínimo de intervenção humana. Uma descrição mais detalhada dessa seção pode ser encontrada na referência [1].

Os métodos mais adotados de machine learning podem ser classificados em três grandes grupos, tendo em vista os tipos de dados de entrada da inteligência:

- **Aprendizado supervisionado:** A inteligência tem acesso a informações como entradas e saídas esperadas, aprendendo como se por através de um "professor". O objetivo do método é encontrar a função que leva das entradas às saídas.
- **Aprendizado não supervisionado:** Não é fornecida como entrada as saídas esperadas, nem nenhum outro tipo de etiqueta acerca dos dados. Cabe ao algoritmo de aprendizado encontrar sozinho os padrões e estruturas nos dados, algumas vezes inclusive descobrindo novos padrões.
- **Aprendizado por reforço:** A inteligência é forçada a interagir com o ambiente

fornecido a ela, ou realizar determinada tarefa ou chegar a um estado em específico, etc., sem muitas ou nenhuma instrução de como efetuar suas ações. Dependendo, então, de uma função de custo idealizada pelo projetista, o computador recebe feedbacks de premiações ou punições com o passar do tempo, aprendendo quais as mais benéficas formas de executar o objetivo dado.

Outra classificação possível se refere as diferentes saídas possíveis de sistemas de aprendizado de máquina:

- **Classificação:** As entradas são classificadas entre duas ou mais tipos discretos e a inteligência deve ser capaz de, para novas entradas (ainda sem classificação), vinculá-las a uma dessas classes pré-definidas. Um exemplo disso é classificar emails como sendo ou não spams, dadas as palavras utilizadas em seu texto.
- **Regressão:** Similar ao problema de classificação, entretanto as saídas são valores contínuos. O processo de precificação de imóveis dado seu tamanho em metros quadrados, número de cômodos e localização pode ser um exemplo de problema de regressão, uma vez que o preço em reais é uma variável contínua.
- **Clustering:** Um problema é de clustering quando deseja-se separar uma entrada em grupo com características semelhantes e/ou análogas. Essa situação difere-se dos problemas de classificação, pois no caso do clustering os grupos não são conhecidos previamente, ou seja, é um método não supervisionado.

A Figura 8 apresenta de forma visual a diferença entre problemas de classificação, de regressão e de clustering.

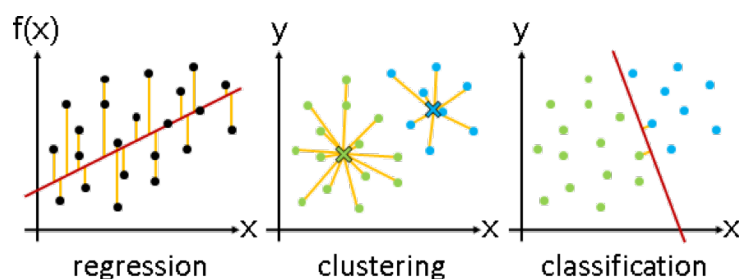


Figura 8: Diferença entre os problemas de aprendizado de máquina de regressão, de clustering e de classificação, da esquerda pra direita [10].

No contexto do projeto de pesquisa, tem-se o objetivo de, partindo de um dataset com as posições dos 22 jogadores e da bola (além de qual é o lado aliado e qual o adversário) e o conhecimento humano de quais jogadores devem ser marcados, classificar cada um dos 11 oponentes de uma situação ainda não vista pela inteligência como marcáveis ou não. Tendo em vista o exposto acima, fica evidente em primeiro momento que o método de aprendizado de máquina supervisionado de classificação aparenta-se ser o ideal para se atingir o resultado esperado.

5 Conclusões

Até o momento, o projeto de Iniciação Científica apresentou bons resultados, uma vez que a interface está em funcionamento e já começarão as aquisições de conhecimento humano para geração de dados para a inteligência. Concomitantemente a isso, será iniciada a seleção da estrutura a ser usada na rede neural e a implementação inicial. Em breve, já será possível analisar os resultados iniciais da estrutura implementada com os dados gerados pela interface.

Do ponto de vista técnico, o uso de aprendizado de máquina por algoritmo supervisionado de classificação, embora mais simples que uma rede neural, traz bastantes esperanças, dado o encaixe entre os problemas que essa abordagem resolve e nosso problema em questão. Espera-se para próxima etapa, testar esse método para comparar os resultados obtidos de ambas as abordagens.

6 Agradecimentos

- Ao CNPq, pelo apoio financeiro e motivacional.
- À ITAndroids, equipe que representa o ITA na competição da LARC/CBR e Robocup, pela ideia do projeto, pela oportunidade de aplicação dos métodos estudados.
- Ao professor Edgar Toshiro Yano, meu orientador, e ao aluno de mestrado Luckeciano Carvalho Melo, co-orientador, ambos da Divisão da Ciência da Computação do ITA, pelo apoio nos estudos e no desenvolvimento do projeto.

Referências

- [1] E. Alpaydin, *Introduction to machine learning*. MIT press, 2009.
- [2] S. Sourceforge, *Soccer simulation*, http://simspark.sourceforge.net/wiki/index.php/Soccer_Simulation, [Online; acessado em 05-Janeiro-2019], 2012.
- [3] Wikipédia, *Robocup 3d soccer simulation league: robot models*, https://en.wikipedia.org/wiki/RoboCup_3D_Soccer_Simulation_League, [Online; acessado em 05-Janeiro-2019], 2017.
- [4] RoboCup, *Rules for the 2018 competition in montreal, canada*, https://ssim.robocup.org/wp-content/uploads/2018/12/Rules_RoboCupSim3D2018.pdf, [Online; acessado em 10-Janeiro-2019], 2018.
- [5] Sourceforge, *Simspark*, <http://simspark.sourceforge.net/>, [Online; acessado em 15-Janeiro-2019].
- [6] RoboCup, *The robocup soccer simulator*, <https://sourceforge.net/projects/sserver/>, [Online; acessado em 15-Janeiro-2019].
- [7] M. Offenburg, *Magmaproxy*, <https://github.com/magmaOffenburg/magmaProxy>, [Online; acessado em 17-Janeiro-2019].
- [8] ———, *Roboviz*, <https://github.com/magmaOffenburg/RoboViz>, [Online; acessado em 20-Janeiro-2019].
- [9] ITAndroids, “Itandroids soccer3d team description paper 2019”, *RoboCup 2019*, vol. 2, p. 7, 2019.
- [10] N. View, *What is machine learning with sap*, <https://www.nextview.nl/what-is-machine-learning-with-sap/>, [Online; acessado em 20-Fevereiro-2019].