

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

Isabelle Ferreira de Oliveira

Instituto Tecnológico de Aeronáutica
Rua H8A, 103, CTA
12.228-460 - São José dos Campos/SP
Bolsista PIBIC - CNPq
isabelle.ferreira3000@gmail.com

Edgar Toshiro Yano

Instituto Tecnológico de Aeronáutica
Divisão de Ciência da Computação
Praça Marechal Eduardo Gomes, 50
12.229-900 – São José dos Campos / SP
etyano2@gmail.com

Luckeciano Carvalho Melo

Instituto Tecnológico de Aeronáutica
LabSCA
Praça Marechal Eduardo Gomes, 50
12.229-900 – São José dos Campos / SP
luckeciano@gmail.com

Resumo: Devido à alta dinamicidade existente nas partidas de futebol simulado de robôs humanoide, é fundamental possuir estratégias também dinâmicas para as mais diversas situações de jogo. Nesse sentido, agregar conhecimento e percepções humanas de qual oponente está em uma posição perigosa e deveria ser marcado - para além dessas regras simples e fixas - pode ajudar a melhorar a defesa desse time em questão.

Após uma fase de pesquisa, foi decidido utilizar Qt, uma framework para desenvolvimento de interfaces gráficas em C++, a fim de criar um sistema de aquisição de dados de conhecimento humano sistema para marcação de oponentes. Nessa interface, o usuário pode interagir com quadros da tela da partida fornecidos periodicamente, montando o dataset de oponentes marcáveis. Esses dados adquiridos alimentaram, então, um algoritmo de aprendizado supervisionado, uma rede neural implementado em Keras, framework para desenvolvimento de redes neurais em Python, treinando, por fim, um modelo de marcação.

Palavras-chave: robótica, aprendizado supervisionado, interface gráfica, Qt, Keras.

1. INTRODUÇÃO

A ITAndroids é uma equipe de alunos do ITA, supervisionada por um professor, que participa de diversas competições de robótica nacionais e internacionais. Uma das categorias em que a ITAndroids participa é a do robô humanoide simulado, que consiste em desenvolver um time de robôs simulados capazes de jogar futebol. Esta tarefa envolve uma série de desafios complexos que variam desde a movimentação do robô até a sua tomada de decisões.

Nesse sentido de estratégia e tomada de decisões, tem-se a questão de marcação de oponentes que estejam ofensivamente perigosos durante a partida. No futebol de robôs humanóides simulados, a marcação de oponentes pode ser muito crucial em uma partida, melhorando significativamente a defesa de um time. Essa marcação consiste em 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 são esses jogadores adversários que estão em situação mais privilegiada em um determinado instante e deveriam ser marcados por agentes aliados.

1.1 Interface Gráfica de Aquisição de Conhecimento Humano

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 no time da ITAndroids.

Para resolver esse problema, então, foi feito a interface em Qt de C++ (QtCompany and KDAB, 1995). Nessa interface, o usuário pode interagir com quadros da tela da partida fornecidos periodicamente, clicando nos jogadores a serem marcados, montando, assim, o dataset de oponentes marcáveis.

1.2 Rede Neural de Classificação com Aprendizado Supervisionado

Os dados adquiridos a partir da interação do usuário - membro da ITAndroids - com a interface gráfica são intencionados a alimentar um algoritmo de classificação com aprendizado supervisionado (Kotsiantis *et al.*, 2007), a fim de tentar automatizar esse processo de marcação com conhecimento humano.

Esse algoritmo foi escolhido dessa forma pela própria natureza do problema. Classifica-se cada um dos oponentes em dois tipos discretos (marcado ou não marcado), e a inteligência deve ser capaz de, para novas entradas (novas posições dos 22 jogadores e da bola, ainda sem classificação), vinculá-las a uma dessas classes pré-definidas. Isso é feito dando à inteligência acesso a informações como entradas e saídas esperadas, treinando-a como se por através de um "professor", objetivando-se encontrar a função que leva das entradas às saídas.

Na implementação dessa rede neural, foi utilizado a framework Keras (Gulli and Pal, 2017), em Python, e, com essa rede, foi treinado um modelo de marcação.

Etapas importantes também inerentes a esse treinamento de modelo foram: pré-processamento dos dados provenientes da interface, geração de gráficos para análise das performances da rede e processamento dos resultados do algoritmo de aprendizado.

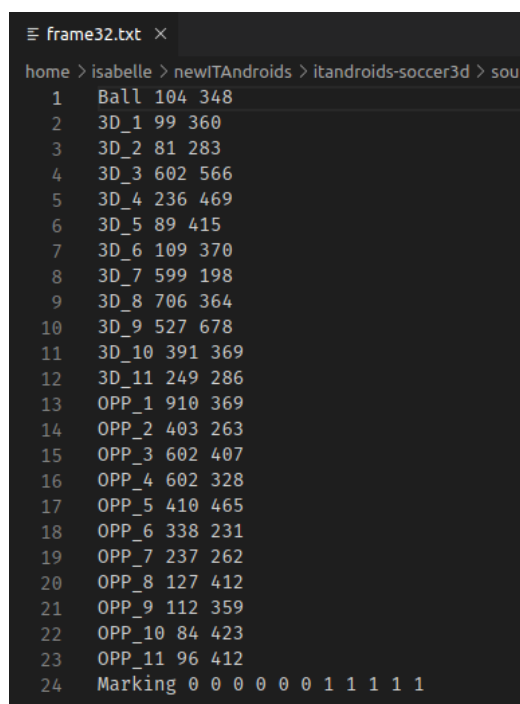
2. RESULTADOS OBTIDOS

Todas as ferramentas foram desenvolvidas com sucesso.

2.1 Interface Gráfica de Aquisição de Conhecimento Humano

Com o uso da plataforma Qt, foi desenvolvida esta ferramenta.

Ela utiliza arquivos de texto (.txt) como entrada e gera como saída alterações nesse mesmo arquivo de texto. Esses arquivos apresentam as posições dos 22 jogadores e da bola, além de um vetor de zeros e uns (1 para oponente marcado e 0 para não marcado). A Figura 1 trata-se de um desses arquivos.



```
frame32.txt x
home > isabelle > newITAndroids > itandroids-soccer3d > sou
1 Ball 104 348
2 3D_1 99 360
3 3D_2 81 283
4 3D_3 602 566
5 3D_4 236 469
6 3D_5 89 415
7 3D_6 109 370
8 3D_7 599 198
9 3D_8 706 364
10 3D_9 527 678
11 3D_10 391 369
12 3D_11 249 286
13 OPP_1 910 369
14 OPP_2 403 263
15 OPP_3 602 407
16 OPP_4 602 328
17 OPP_5 410 465
18 OPP_6 338 231
19 OPP_7 237 262
20 OPP_8 127 412
21 OPP_9 112 359
22 OPP_10 84 423
23 OPP_11 96 412
24 Marking 0 0 0 0 0 0 1 1 1 1 1
```

Figura 1: Arquivo de entrada da interface. No exemplo, esse arquivo se trata do 32º frame analisado.

A interação na interface se dá clicando no oponente a ser marcado, o que automaticamente altera o vetor de 0s e 1s do arquivo de entrada, além da coloração do jogador (de vermelho para verde) para fins de visualização. A ferramenta também possui quatro botões:

- prevFrame: Apresenta a visualização do frame anterior.
- nextFrame: Apresenta a visualização do frame seguinte.
- unmarkAll: Desmarca todos os oponentes do frame atual.
- confirm: Atualiza o arquivo de entrada para o caso de o usuário ter desmarcado todos os oponentes.

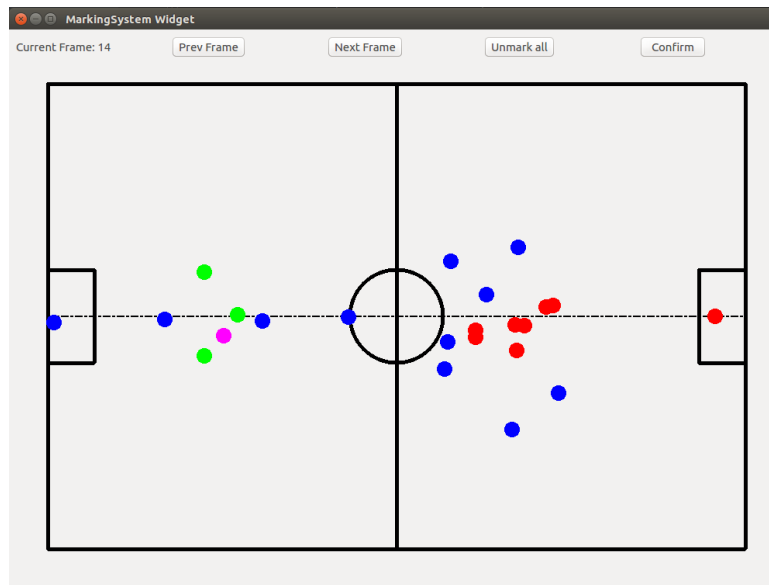


Figura 2: Aparência da ferramenta de interface gráfica, com alguns oponentes marcados. Jogadores aliados são azuis, oponentes são vermelhos e passam a ser verdes caso sejam marcados. A bola é rosa. No exemplo, esse arquivo se trata do 14º frame analisado.

Pode visualizar a aparência da ferramenta de interface gráfica desenvolvida na Figura 2.

2.2 Rede Neural de Classificação com Aprendizado Supervisionado

O framework utilizado para escrever a rede neural foi Keras, uma biblioteca de rede neural de código aberto escrita em Python. Keras foi escolhido por ser fácil de usar, modular e extensível, oferecendo um conjunto de abstrações de nível mais intuitivo, o que facilita o desenvolvimento de modelos de aprendizagem, até mesmo para deep learning (Chollet *et al.*, 2015).

A implementação do modelo foi feita, então, em Keras, e uma representação resumida do seu modelo foi apresentado na Figura 3. Esse resumo é o produzido através da chamada do método `model.summary()`, fornecido pela própria framework Keras. A rede consiste de 3 camadas escondidas: as duas primeiras com 64 neurônios e a última com 11 neurônios, referente as informações dos 11 jogadores oponentes (se eles devem ser marcados, ou não). A função de ativação utilizada foi a tangente hiperbólica para as duas primeiras camadas escondidas e sigmoide para a última.

Layer (type)	Output Shape	Param #
dense 1 (Dense)	(None, 64)	3008
dense 2 (Dense)	(None, 64)	4160
dense 3 (Dense)	(None, 11)	715
Total params: 7,883		
Trainable params: 7,883		
Non-trainable params: 0		

Figura 3: Resumo do modelo implantado utilizando Keras, em Python.

Utilizou-se inicialmente um dataset de 500 frames, referentes a mais de 4 horas seguidas de partida. Esse dataset foi gerado conforme explicado anteriormente, a partir de conhecimento humano. Desse conjunto de frames, 300 (60% do total) foram escolhidos para o treinamento da rede, 100 (20% do total) para set de validação cruzada. Já os últimos 100 (20% do total), foram escolhidos para o teste propriamente dito, ou seja, um dataset de entradas novas, as quais a rede jamais havia tido contato anteriormente. O treino da rede obteve um modelo com acurácia de 89.15% no dataset de treino, 85.57% no dataset de validação cruzada, e 86.92% no dataset de teste.

A evolução da acurácia e valor de custo com o passar das épocas foram apresentadas na Figura 4. A partir dessas figuras, pode-se notar a convergência desses valores.

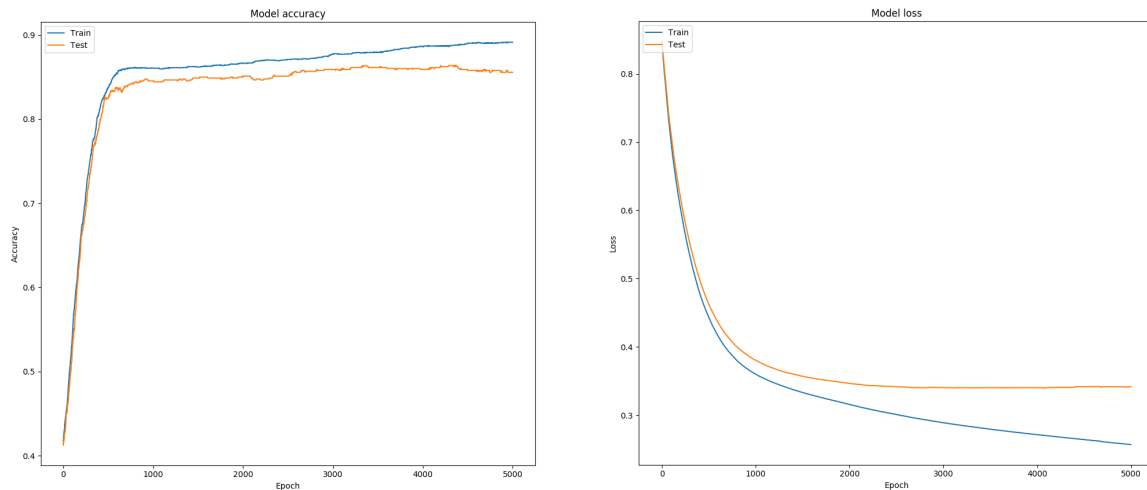


Figura 4: Acurácia e função de custo, da esquerda para direita, do modelo com o passar das épocas.

Alguns exemplos dos resultados no dataset de teste foram apresentados a seguir.

Nas Figuras 5 e 6, consegue-se perceber um funcionamento aceitável na escolha dos oponentes a serem marcados. Nota-se também que a rede desenvolveu uma tendência de se marcar poucos oponentes, o que não necessariamente é o desejado. Na Figura 7, por exemplo, pode-se perceber que essa escolha de se marcar poucos oponentes pode acabar sendo prejudicial em algumas situações de jogo.

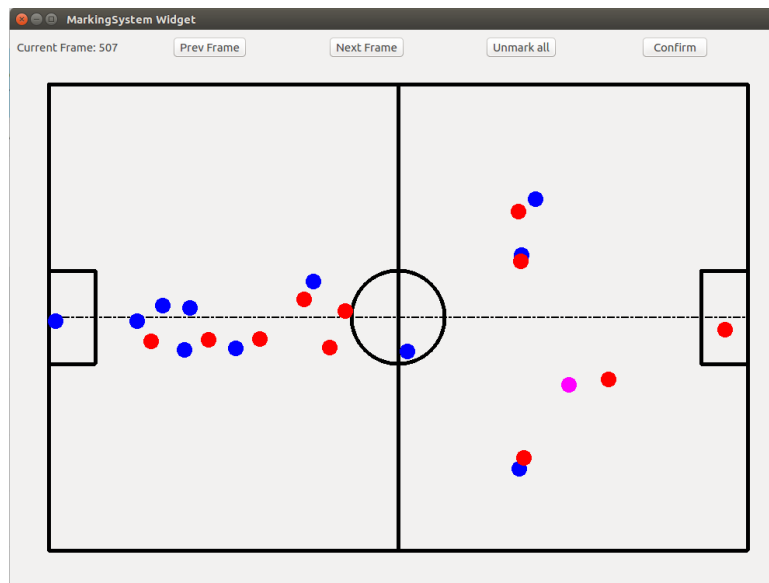


Figura 5: Exemplo de correta marcação. Para bola presente muito à direita, no lado atacante do campo, é entendido que não se tem necessidade de marcar oponentes.

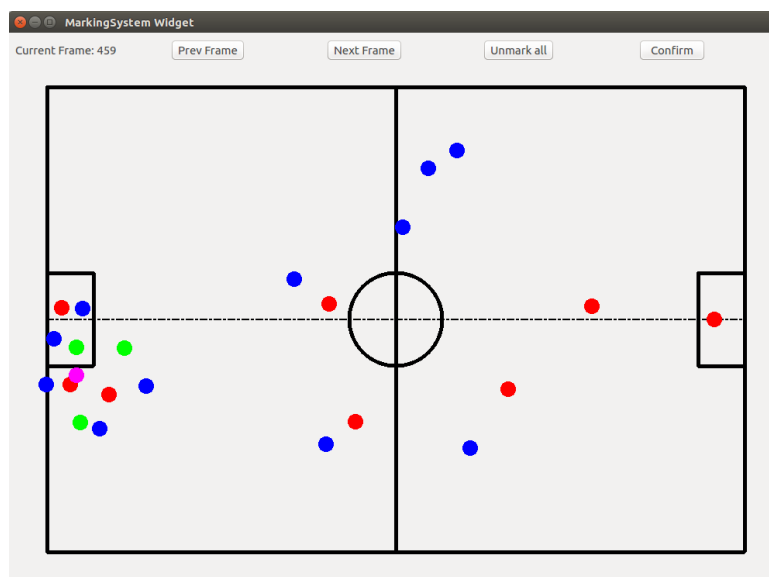


Figura 6: Exemplo de marcação satisfatória. Nota-se a tendência da rede de marcar poucos oponentes, uma vez que há outros oponentes em situação perigosa que não estão sendo marcados.

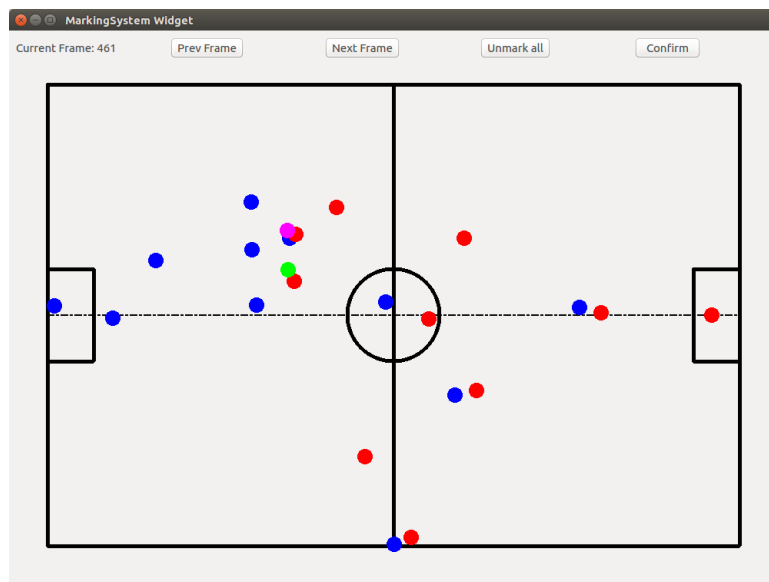


Figura 7: Exemplo de marcação não tão satisfatória, uma vez que apenas um oponente foi marcado, bastante próximo a outro também em posição perigosa.

3. CONCLUSÕES

O framework Qt é excelente para a criação rápida de ferramentas de interface com o usuário. Esta ferramenta inicial criada agora poderá ser aprimorada ou servir de modelo para a criação de mais ferramentas no futuro.

A seleção da estrutura a ser usada na rede neural e a sua implementação também se mostraram interessantes. Foi possível a partir dela analisar os resultados da estrutura implementada com os dados gerados pela interface, conforme foi apresentado anteriormente.

Pode-se ainda aprimorar a inteligência, a partir da aquisição de mais dataset, por exemplo, ou revisão do atualmente desenvolvido. A forma como está atualmente implementado possibilita essa posterior melhoria, conforme se percebe a necessidade.

Outras sugestões de melhorias são, também, a elaboração de scripts mais compactados ou simplificados para se executar o projeto do início (desde a captura dos quadros de partidas reais e elaboração de arquivos de entrada da interface) até o final (considerando o pré-processamento de dados adquiridos, treino da rede, aplicação do modelo em um dataset de treino, de validação cruzada e de teste).

É essencial para a progressão do time a aplicação dessa estratégia desenvolvida no código da ITAndroids, para verificar os resultados dessas mudanças também em partidas reais com times adversários.

4. 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 Yano, meu orientador, e ao mestre Luckeciano Carvalho Melo, co-orientador, da Divisão da Ciência da Computação do ITA e do LabSCA, respectivamente, pelo apoio nos estudos e no desenvolvimento do projeto.

5. REFERÊNCIAS

Chollet, F. *et al.*, 2015. *Keras Documentation: Why use Keras?* URL <https://keras.io/why-use-keras/>.

Gulli, A. and Pal, S., 2017. *Deep Learning with Keras*. Packt Publishing Ltd.

Kotsiantis, S.B., Zaharakis, I. and Pintelas, P., 2007. “Supervised machine learning: A review of classification techniques”. *Emerging artificial intelligence applications in computer engineering*, Vol. 160, pp. 3–24.

QtCompany and KDAB, 1995. *Qt Documentation*. URL <https://doc.qt.io/>.