MATEUS CORDEIRO GONÇALVES DE CARVALHO

ANÁLISES DE PERFIS DE PERSONAGENS E JOGADORES DE LEAGUE OF LEGENDS USANDO APRENDIZADO DE MÁQUINA

Este Trabalho de Graduação foi apresentado ao Departamento de Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Ricardo Araújo Rios

Salvador
19 de Dezembro de 2018

ATENÇão: O aluno deve ir até a Biblioteca Central da UFBA e solicitar uma FICHA CATALOGRAFICA com o formato abaixo e com números de CDD e CDU para seu trabalho.

Sistema de Bibliotecas - UFBA

.

Análises de Perfis de Personagens e Jogadores de League of Legends usando Aprendizado de Máquina / Mateus Cordeiro Gonçalves de Carvalho – Salvador, 2018.

66p.: il.

Orientador: Prof. Dr. Ricardo Araújo Rios.

Trabalho de Conclusão do Curso – Universidade Federal da Bahia, Instituto de Matemática e Estatística, 2018.

. I. Rios, Ricardo Araújo. II. Universidade Federal da Bahia. Instituto de Matemática e Estatística. III Título.

CDD – XXX.XX

CDU – XXX.XX.XXX

TERMO DE APROVAÇÃO

MATEUS CORDEIRO GONÇALVES DE CARVALHO

ANÁLISES DE PERFIS DE PERSONAGENS E JOGADORES DE LEAGUE OF LEGENDS USANDO APRENDIZADO DE MÁQUINA

Esta Trabalho de Graduação foi julgado adequado à obtenção do título de Bacharel em Ciência da Computação e aprovado em sua forma final pelo Departamento de Ciência da Computação da Universidade Federal da Bahia.

Salvador, 19 de Dezembro de 2018

Prof. Dr. Ricardo Araújo Rios
Universidade Federal da Bahia

RESUMO

Os esportes eletrônicos são jogos de natureza competitiva, disputados individualmente ou em equipe. Este segmento ganhou força nos últimos anos com a popularização de jogos como: CS:GO, League of Legends, Dota e Fortnite. Neste trabalho, League of Legends foi escolhido para ser alvo de estudos envolvendo Aprendizado de Máquina. O LoL é um título do gênero Multiplayer Online Battle Arena (MOBA) e um dos jogos mais populares do mundo. Foram criadas bases de dados utilizando informações dos personagens, de jogadores e de partidas, coletados diretamente das plataformas do jogo. Na primeira parte do trabalho, realizou-se experimentos de agrupamento e os resultados foram correlacionados aos rótulos predefinidos sobre os personagens e as funções dos jogadores dentro da sua equipe. Já na segunda etapa, o foco passou a ser a utilização de algoritmos de classificação para a predição do resultados de partidas, antes delas começarem, mas após a formação dos times e escolha dos personagens. O principal objetivo, nesta etapa, foi a utilização de dados sobre a experiência dos jogadores para representar as suas habilidades.

Palavras-chave: Aprendizado de Máquina; Esportes eletrônicos;

ABSTRACT

Electronic sports are competitive games, played individually or in teams. This segment has grown in the last years with the popularization of games, such as: CS:GO, League of Legends, Dota and Fortnite. In this work, League of Legends was used for Machine Learning studies. League of Legends (LoL) is a MOBA title and one of the most popular games in the world. Data about characters, players and matches were collected from the game and used to create the datasets. In the first part of this work, clustering analysis was applied and the groups formed were compared with predefined information about the characters and the roles of the players in team work. The second part focused on using classification algorithms to predict the winning team, before the match begins, but after teams formation and selection of characters. The main goal was to use players' experience data to represent their abilities.

Keywords: Machine Learning; Electronic sports;

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

FPS First-Person Shooter

RTS Real Time Strategy

MOBA Multiplayer Online Battle Arena

LoL League of Legends

DotA Defence of the Ancients

API Aplication Programming Interface

CBLOL Campeonato Brasileiro de League of Legends

MSI Mid-Season Invitational

MLP Multilayer Perceptron

CNN Convolutional Neural Network

PCA Principal Component Analysis

RPG Role Playing Game

Capítulo 1

INTRODUÇÃO

Os esportes eletrônicos, ou e-sports, são jogos de natureza competitiva, em sua grande maioria, no formato de partidas que podem ser jogadas online. Este segmento popularizou-se nos últimos anos e, hoje, representa um grande mercado formado por jogos com milhões de fãs, além de times e ligas profissionais. Existem alguns tipos mais conhecidos de esportes eletrônicos como: tiro em primeira pessoa (First-Person Shooter (FPS)), jogos de luta, estratégia em tempo real (Real Time Strategy (RTS)).

O e-sport escolhido neste trabalho foi o League of Legends. LoL é um dos jogos mais populares do mundo, com milhões de usuários jogando todos os dias (TASSI, 2014). League of Legends é um jogo do estilo MOBA, no qual, dois times batalham em uma arena para coletarem recursos, se fortalecerem e destruírem a base da equipe inimiga para vencerem a partida.

LoL possui um cenário competitivo e profissional bem estabelecido em várias partes do mundo. Para tornar-se um bom jogador, é necessário conhecer, em detalhes, as mecânicas do jogo e praticar bastante. O modo de jogo mais tradicional de League of Legends é disputado por 2 (dois) times com 5 (cinco) integrantes e o trabalho em equipe pode fazer toda diferença. O LoL está em constante mudança, seja por alterações realizadas pelos desenvolvedores ou por novas estratégias, padrões e práticas estabelecidos pelos jogadores.

A procura constante por melhores desempenhos no jogo impulsiona os estudos e análises de dados, além da criação de aplicações, com o objetivo de agregar novos conhecimentos e contribuir com a evolução dos jogadores. A própria desenvolvedora do LoL, a empresa *Riot Games*, procura incentivar projetos desta natureza, fornecendo, por exemplo, uma Aplication Programming Interface (API) (GAMES, 2018), que é uma interface de comunicação para coleta de dados do jogo, jogadores, partidas e torneios.

Este trabalho tem como objetivo a realização de análises em dados do League of Legends, utilizando métodos de Aprendizado de Máquina. O trabalho foi dividido em duas partes. A primeira foi voltada à utilização de aprendizado não supervisionado, mais especificamente, a aplicação de técnicas de agrupamento em dados sobre as características dos

personagens e também sobre o desempenho de jogadores em partidas reais. A principal análise desses experimentos foi a comparação dos resultados com padrões (agrupamentos) predefinidos pelo jogo sobre os tipos de personagens ou pela comunidade em relação às funções de cada jogador durante uma partida, através de métricas de avaliação externa.

A segunda parte deste trabalho foi focada na aplicação de aprendizado supervisionado para predizer qual time venceu uma partida, que é o principal objetivo do jogo. O LoL possui um sistema de seleção de jogadores para uma partida que procura balancear o nível de habilidade dos dois times.

Alguns trabalhos similares já foram realizados, principalmente com o jogo Dota 2. No entanto, a maioria deles limitaram-se à lista dos personagens escolhidos. Considerando que essas informações são insuficientes, se comparado a todos os fatores que influenciam o resultado de uma partida, neste trabalho, foram utilizadas, também, estatísticas sobre os personagens e informações sobre a experiência dos jogadores, com o objetivo de representar o nível de habilidade deles. Todos os dados escolhidos podem ser observados antes do início da partida.

Os resultados mostram a complexidade desta tarefa e a eficiência do sistema de formação das equipes. Ainda assim, foi possível verificar a importância dos dados de experiência dos jogadores nas análises que envolvem os métodos de Aprendizado.

O restante deste texto é dividido da seguinte forma. O Capítulo 2 explica mais detalhadamente os esportes eletrônicos e o League of Legends. O Capítulo 3 relata um pouco a evolução da utilização de análises estatísticas e aprendizado de máquina em jogos eletrônicos, com a discussão de diversos trabalhos envolvendo e-sports. O Capítulo 4 traz uma introdução teórica aos métodos e algoritmos utilizados em todo o trabalho. O Capítulo 5 explica os experimentos de agrupamento e seus resultados. O equivalente para a tarefa de predição do resultado de partidas é realizado no Capítulo 6. O Capítulo 7 apresenta as conclusões deste trabalho.

Capítulo 2

ESPORTES ELETRÔNICOS E LEAGUE OF LEGENDS

Os esportes eletrônicos ou e-sports são um segmento do mercado de jogos eletrônicos formado por jogos com alto potencial competitivo. Atualmente esse mercado possui milhões de consumidores e movimenta bilhões de dólares anualmente (NEWZOO, 2018). Os e-sports surgiram através do interesse na natureza competitiva dos jogos eletrônicos, assim como as pessoas se interessam por futebol ou outros esportes, o que tornou oportuno a criação de jogos voltados a esse mercado. Os esportes eletrônicos normalmente possuem formato de partidas disputadas por equipes ou jogadores adversários, com regras, objetivos e resultados claros.

Os esportes eletrônicos são jogados na maioria das vezes em rede, e os avanços na qualidade de conexão banda larga foram essenciais para a expansão desses jogos. Outro fator importante é que eles são atrativos aos jogadores mais dedicados, mas também aos casuais, por serem em partidas relativamente curtas e pelo nivelamento automático entre os adversários, que acontece na grande maioria desses jogos.

A associação de jogos eletrônicos ao termo esporte gera bastante discussão e é motivada pelas grandes semelhanças a um esporte tradicional. Jogadores se tornam mais habilidosos ao treinarem e passam a ter resultados melhores. Jogadores formam equipes para participarem de competições amadoras e profissionais com apoio dos seus fãs presentes nos eventos ou assistindo à transmissão ao vivo. A maior diferença é que um e-sport não exige exercício físico intenso, ao contrário da maioria das modalidades tradicionais. O que pode mudar no futuro com os avanços e diversificações dos equipamentos utilizados para se jogar. Um exemplo disso são as câmeras 3D, controles sensíveis ao movimento e óculos de realidade virtual, que são tecnologias já presentes no mercado.

O grande crescimento dos esportes eletrônicos está abrindo espaço na mídia esportiva tradicional, como canais esportivos, sites de notícias, e atraindo cada vez mais investimentos. Segundo estimativas da Newzoo, em 2018 a movimentação econômica das competições será cerca de 906 milhões de dólares, dos quais 694 milhões será de investimentos das empresas em propaganda, patrocínio e direitos de mídia. O cenário competitivo ainda não é lucrativo para a maioria das desenvolvedoras dos jogos, mas esses investimentos

Figura 2.1: League of Legends 2017 world championship

são compensados pelo impacto das competições no engajamento dos fãs e na arrecadação com o jogo em si.

O Brasil ocupa a 11ª posição em relação ao total de dinheiro arrecadado por jogadores do país com premiação em competições, com quase 10 milhões de dólares (EARNINGS, 2018). No entanto boa parte desse valor foram de competições fora do Brasil. Em primeiro lugar estão os chineses com mais de 70 milhões.

2.1 GÊNEROS DOS E-SPORTS

O esportes eletrônicos geralmente se enquadram dentro de algum gênero, que identifica o estilo de ambiente, objetivo, jogabilidade e outros quesitos. Os gêneros mais comuns até hoje são:

- **Luta:** Esse estilo é um dos mais antigos e um dos primeiros a desenvolver um cenário competitivo. O gênero de luta é conhecido desde os tempos dos fliperamas, e utilizado até hoje em jogos de console e computador.

- **Tiro em primeira pessoa (FPS):** Simulação de tiroteio entre geralmente duas equipes adversárias. Nesse gênero a câmera do jogador simula a visualização da cena pelo olhar do personagem que deverá mirar e atirar nos oponentes

- **Estratégia em tempo real (em inglês, Real Time Strategy, RTS):** Nesses jogos os jogadores se tornam governantes e comandam aspectos econômicos, militares, tecnológicos, políticos de por exemplo um império, na tentativa de estar a frente de seus oponentes e vencê-los em combate militar. Este gênero foi um dos principais responsáveis pela expansão dos esportes eletrônicos.

- **Esportes:** Existem diversos jogos eletrônicos que simulam esportes tradicionais, como: futebol, futebol americano, basquete e hóquei. Esses jogos acabam atraindo muito os fãs da própria modalidade simulada.

- **Corrida:** Simulação de corrida automobilística de pista ou de rua. Este é um gênero muito popular entre os jogos apesar de não ter um cenário competitivo tão desenvolvido.

- **Arena de batalha multijogador online (em inglês, Multiplayer Online Battle Arena, MOBA):** Um dos gêneros mais recentes a ser desenvolvido, mas que já possui o maior cenário competitivo globalmente. O gênero MOBA se tornou popular no início dos anos 2000 com a modificação de um RTS fazendo com que o jogador controle apenas um personagem.

Pesquisas relacionadas à análise de dados desses jogos podem, por exemplo, ajudar os desenvolvedores a entenderem como os jogadores interagem com o jogo, como suas ações influenciam os possíveis resultados e até mesmo criar agentes inteligentes capazes de atuar como jogadores. Alguns trabalhos envolvendo esportes eletrônicos, principalmente do gênero MOBA, são discutidos no próximo capítulo.

2.2 LEAGUE OF LEGENDS

Todos os dados utilizados nas análises deste trabalho foram do MOBA LoL, o jogo mais jogado no mundo (em quantidade de horas mensais) e o maior título entre os e-sports. Além das possibilidades de aplicações, um dos maiores benefícios de se escolher o League of Legends para esse trabalho é a disponibilidade de dados. Utilizando a API do jogo e outros serviços é possível obter dados, por exemplo, sobre os jogadores e suas partidas ou sobre os personagens.

Um dos primeiros jogos MOBA surgiu a partir de uma modificação do jogo StarCraft (RTS) em que o jogador controla apenas um personagem (MINOTTI, 2014). Uma modificação similar foi realizada no jogo Warcraft III, dando origem ao título Defence of the Ancients (DotA), lançado em 2003, que foi o primeiro MOBA mundialmente popular e estabeleceu um padrão para este gênero, sendo base para o jogo League of Legends e outros títulos. League of Legends foi lançado no ano de 2009 obtendo grande popularidade. Em 2012 já era o jogo de computador mais jogado em diversas partes do mundo. Atualmente possui mais de 100 milhões de jogadores mensais (VOLK, 2016).

2.2.1 Funcionamento do jogo

league of Legends possui mais de uma arena e modo de jogo. O principal deles, e o único utilizado neste trabalho é o *Summoner's Rift*. Neste modo, 2 (duas) equipes formadas por 5 (cinco) jogadores disputam uma partida que dura, em média, entre 20 (vinte) e 40 (quarenta) minutos. O tempo de uma partida não é fixo porque ela só acaba quando uma equipe (vencedora) destrói o *Nexus* da equipe inimiga. Até chegar ao *Nexus*, o time precisa destruir diversas outras estruturas dispostas no território inimigo.

Figura 2.2: Visão aérea do Summoner's Rift: 1. Base da equipe. 2. Rota superior. 3. Rota do meio. 4. Rota inferior. 5. Selva. 6. Torres externas. 7. Inibidores e suas torres. 8. Nexus e suas torres. 9. Covil do Barão 10. Covil do Dragão

Cada jogador controla um personagem durante a partida e esta escolha é um dos pontos mais importantes do jogo. Esses personagens são oficialmente conhecidos como campeões. Cada um deles possui características, estilos e habilidades específicas. Existem cerca de 140 campeões atualmente no LoL. Esses personagens são verdadeiras criações literárias, pois além das características de batalha, eles possuem uma história, local de origem, motivação e falas, que ajudam a criar uma imersão maior ao jogo.

2.2.1.1 Arena, territórios e bases: A arena onde a partida acontece tem um formato retangular e tem uma configuração majoritariamente simétrica entre os territórios de cada equipe. Três rotas conectam as bases das duas equipes cortando toda arena, cercadas pela selva (área hostil). No território de cada equipe existem diversas construções de proteção. A Figura 2.2 mostra em detalhes a geografia da arena e a localização das estruturas. Os jogadores partem da base da sua equipe e se dividem para ocupar as rotas e a selva.

2.2.1.2 Câmera, visibilidade e minimapa: Os jogadores possuem uma visão em terceira pessoa, aérea e levemente inclinada da arena. A câmera pode ser movida ao longo

Figura 2.3: Minimapa

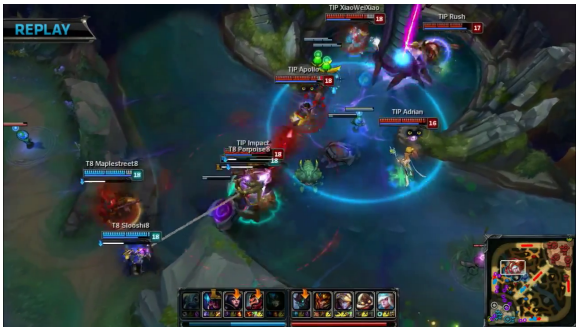


Figura 2.4: Disputa pelo Barão

Figura 2.5: Combate dentro da base

Figura 2.6: Combate na fase de rotas

do mapa, independente da posição do personagem controlado pelo jogador. Quando a câmera está sobre uma área do mapa, o jogador verá o que está acontecendo, caso o seu time possua visão daquela região. Caso contrário, será visto apenas o terreno imerso na escuridão. A visão de uma área é habilitada pela presença do time nas proximidades, seja com um dos personagens, torres, tropas, itens ou habilidades.

Enquanto que a câmera consegue visualizar uma região pequena de toda a arena, na tela do jogador (normalmente no canto direito inferior) existe um minimapa (Figura 2.3) "vivo" com desenho da arena e a localização de maioria das estruturas e personagens (visíveis a equipe do jogador).

2.2.1.3 Experiência, Recursos e combates: Ao longo de uma partida os personagens ganham experiência e evoluem, aumentando suas forças e habilidades. Os jogadores também ganham dinheiro continuamente além de quantias bônus ao abater inimigos, tropas e monstros da selva. O dinheiro é usado para comprar itens durante a partida que irão deixar o personagem ainda mais forte. Enquanto as habilidades de um personagem são as mesmas em todas as partidas, os itens devem ser comprados levando em consideração os outros personagens do jogo, principalmente os adversários.

Os combates são uma das coisas mais importantes do jogo.Quando um jogador abate outro, ele ganha recursos e o adversário morto fica alguns segundos fora do jogo até retornar na fonte da sua base. As Figuras 2.4, 2.5 e 2.6 mostram exemplos de situações de combate durante uma partida.

2.2.2 Cenário competitivo e profissional

Os jogadores de LoL podem jogar partidas no modo casual ou no modo ranqueado onde sua performance lhe concederá pontos que são utilizados em um sistema de divisões que classificam todos os jogadores. Esse sistema de ranqueamento do jogo é uma forma de verificar o nível de habilidade do jogador. Jogadores profissionais normalmente estão dividindo o topo dessa classificação com jogadores amadores, também muito habilidosos. Essa classificação é, de certa forma, reiniciada todo ano, devido a organização em temporadas.

O League of Legends é jogado profissionalmente em várias partes do mundo. Em todas as regiões existem competições a nível nacional ou continental. Estados Unidos, Europa, Coreia e China são as regiões mais desenvolvidas nesse sentido. O principal torneio no Brasil é o Campeonato Brasileiro de League of Legends (CBLOL) que possui duas edições no ano e em 2018 a premiação em cada edição foi de 200 mil reais (TECHTUDO, 2018). Além de participar de competições os jogadores fazem a transmissão das suas partidas no dia-dia para os fãs, postam vídeos com conteúdos personalizados, ensinam outros jogadores, e as equipes vendem produtos personalizados como camisetas.

Muitos torneios oferecem premiações milionárias. O campeonato mundial de 2017 teve premiação total de quase 5 milhões de dólares. Em um outro torneio internacional, o Mid-Season Invitational (MSI) que aconteceu no primeiro semestre de 2018, parte da arrecadação de algum item comprável do jogo foi para a premiação da competição, o valor acrescentado foi de mais de um milhão de dólares (STAFF, 2018).

2.2.3 Classes e Subclasses

Cada personagem do jogo LoL possui uma valoração para um conjunto de atributos como ataque, defesa, agilidade, além de habilidades com os mais diversos efeitos. Apesar das características únicas, ainda é possível identificar padrões e muitas semelhanças entre eles. É possível relacionar os personagens a perfis muito utilizados nos universos fantasiosos e jogos de Role Playing Game (RPG). Existem, por exemplo, os personagens mais resistentes, mais letais, os mágicos e os de suporte. No League of Legends existem duas divisões seguindo essa lógica, que são as classes e as subclasses, com 7 e 13 grupos respectivamente. A segunda divisão é uma especialização da primeira. A Tabela 2.1 mostra a todas as classes e subclasses do jogo.

Classe	Quantidade	Subclasse	Quantidade
Controller	15	Catcher	8
		Enchanter	7
Fighter	31	Diver	18
		Juggernaut	13
Mage	26	Artillery	4
		Battlamage	11
		Burst	11
Marksman	19	Marksman	19
Slayer	18	Assassin	11
		Skirmisher	7
Tank	18	Vanguard	12
		Warden	6
Specialist	14	Specialist	14

Tabela 2.1: Classes e subclasses do jogo League of Legends

- **Controller:** Personagens de dão suporte aos aliados e atrapalham os adversários com suas habilidades de controle.
  - Catcher: Mais foco em controlar e atrapalhar os inimigos
  - Enchanter: Mais foco em fortalecer e recuperar os aliados
- **Figther:** Personagens de combate corpo a corpo que são fortes e ao mesmo tempo resistentes.
  - Diver: Maior mobilidade
  - Juggernaut: Maior resistência
- **Mage:** Personagens que causam dano mágico e efeitos de controle a distância através das habilidades.

  - Artillery: Maior alcance
  - Battlemage: Maior foco em combate corpo a corpo, mais resistentes.
  - Burst: Grande dano em curto espaço de tempo

- **Marksman:** Personagens que atacam a distância causando grande e contínuo dano físico que não depende tanto das habilidades. Muito frágeis.

- **Slayer:** Personagens com grande mobilidade e muito dano em pouco tempo. Normalmente com pouca resistência.

  - Assassin: Maior mobilidade, foco em se infiltrar entre os inimigos para atacar alvos de alta prioridade.
  - Skirmisher: Um pouco menos de mobilidade, focam em alvos mais próximos.

- **Tank:** Personagens muito resistentes e com habilidades de controle. Possuem foco em tomar dano pela equipe e atrapalhar os adversários.

  - Vanguard: Maior foco ofensivo. Possuem habilidades para iniciar ataques
  - Warden: Maior foco defensivo. Possuem habilidade para evitar dano aos aliados.

- **Specialist:** Personagens que são muito diferentes e não se encaixam em nenhuma classe específica.

As definições de classes e subclasses são uma ótima forma dos jogadores entenderem o básico sobre um personagem, e conseguirem saber quais deles possuem jogabilidade parecida. Uma tática de jogo que funciona com um determinado personagem pode funcionar com outros do mesmo tipo que ele e, como nem sempre todos os personagens estão disponíveis aos jogadores (que primeiramente precisam comprá-los com a dinheiro ganho no jogo) devido também a escolha ou banimento prévio naquela partida, saber opções semelhantes é muito importante.

2.2.4 Funções dos jogadores no time

Todo esporte em equipe requer uma forte formação tática e também uma boa divisão de responsabilidades que vão guiar o comportamento e posicionamento dos jogadores. O estabelecimento de papéis é ainda mais crucial em um MOBA devido aos diferentes elementos da arena e à possibilidade de eventos importantes acontecerem ao mesmo tempo em diferentes regiões. As funções estabelecidas em um MOBA envolvem quais jogadores, por exemplo, irão se posicionar em cada rota, terão foco em causar danos aos oponentes, dar suporte aos companheiros, e ter um controle maior sobre o mapa e a visibilidade da equipe. No League of Legends um esquema de 5 funções é o mais utilizado. Para cada uma das funções existem personagens que são considerados muito bons, medianos ou ruins e, normalmente, os jogadores seguem esses padrões. Essa forma de se jogar se tornou tão bem estabelecida que foi incorporada pelo próprio jogo. No momento em

que os jogadores quiserem participar de uma nova partida eles escolhem qual posição gostariam de jogar, e então, os times já são formados com 1 jogador para cada posição. Na lista abaixo estão descritas as 5 funções.

- **Topo:** É comum que apenas 1 jogador do time movimente-se para a rota superior durante a fase de rotas. Esse jogador deve explorar ao máximo os recursos dessa rota e disputar com o jogador da mesma função do time adversário. Comumente a rota do tempo passa a maior parte do tempo isolada do resto dos eventos, sendo então comum a escolha de uma habilidade de teletransporte por parte dos jogadores dessa função para que possam fazer uma mudança de posicionamento rápida quando for necessário.

- **Meio:** Jogar a fase de rota na rota do meio, que também comumente ocupada por apenas 1 jogador de cada equipe. A rota do meio é cercada por selva de ambos os lados e por isso o jogador deve ficar ainda mais atento aos ataques surpresas. A rota do meio é também a mais curta e portanto a mais rápida de se chegar ao ponto de encontro entre os territórios das duas equipes. Muito comum a utilização de personagens mágicos com mais ataque e alguns atiradores (marksman). É preferível usar personagens que possuam uma boa movimentação ou que consigam atacar a distância.

- **Inferior:** Esse função há muito tempo é predominantemente exercida com personagens da classe dos atiradores para focar em causar altíssimo e constante dano físico à distância. Quase sempre os jogadores vão para a rota inferior e batalham com a ajuda de um outro membro do time, o suporte, para lhe defender. Os atiradores têm toda a prioridade para destruir as tropas da rota e ganhar o dinheiro delas. Muitas vezes o jogador dessa função é o maior causador de dano da equipe e o dano físico constante também o faz um ótimo destruidor de torres.

- **Suporte:** O suporte tem como principal função auxiliar a sua equipe, principalmente o atirador. Dessa forma, está quase sempre ao seu lado na rota inferior. Os suportes abrem mão de abater tropas em prol dos outros membros da equipe, e por isso, ganham menos recursos que os outros jogadores. Além disso eles ainda gastam muito comprando itens de visão que é uma responsabilidade secundária dessa função. Os personagens utilizados para suporte normalmente possuem habilidades de controle e utilidade.

- **Caçador:** O caçador é o único jogador da equipe que não ocupa alguma das rotas, ao invés disso, ele se movimenta por toda a selva, principalmente a do seu território, eliminando os monstros neutros para ganhar recursos. Por estar na selva ele passa muito tempo sem ser visto pelo time adversário, sendo assim um dos seus maiores papéis é realizar ataques surpresas aos inimigos nas rotas. Os personagens utilizados nessa rota devem preferivelmente possuir uma boa mobilidade e mecanismos que auxiliem na execução dos ataques às rotas.

Capítulo 3

REVISÃO BIBLIOGRÁFICA

Video games são abordados em trabalhos de pesquisa científica de diversas formas. Duas abordagens muito comuns são: 1. a utilização de jogos como ferramenta educacional; 2. o estudo da correlação entre a forma de se jogar e as características interpessoais e perfil profissional dos jogadores (NUANGJUMNONG; MITOMO, 2012) (REEVES; MALONE, 2007). Um ponto comum dessas abordagens é a utilização do jogo como meio ou ferramenta, não sendo o alvo direto de possíveis contribuições.

O crescimento da cultura competitiva e o surgimento dos esportes eletrônicos trouxeram um incentivo à análise dos quesitos táticos, estratégicos e o desempenho nesses jogos com o objetivo de auxiliar no treinamento dos jogadores e times.

A natureza digital dos e-sports trazem uma vantagem na sua utilização em relação aos esportes tradicionais que é a disponibilidade dos dados. Os ambientes desses jogos são totalmente controlados e monitoráveis, os dados podem ser computados e processados nos mínimos detalhes e são comumente disponibilizados ao público. Existem também diversas iniciativas da comunidade para criar ferramentas que ajudam os fãs a terem acesso a estatísticas atualizadas dos jogos.

Existem trabalhos envolvendo os diferentes gêneros do e-sports. Por exemplo, Hsieh e Sun (2008) utilizam os logs de replay do RTS StarCraft para analisar, aprender e predizer o estilo de jogadores. Similarmente Liu, Ballinger e Louis (2013) utilizam dados das partidas para identificar o jogador pelo seu estilo de jogo, no StarCraft II.

3.1 MOBA

Jogos MOBA possuem grande similaridade com esportes tradicionais em equipe. Em um campo simétrico os jogadores devem atuar estrategicamente para vencer o time adversário. Devido ao nível de detalhes desses jogos, faz-se necessário um alto investimento de tempo para se jogar bem e se destacar competitivamente. Eventos importantes podem ocorrer em várias regiões do mapa simultaneamente, além disso o jogador deve dominar

as mecânicas específicas do seu personagem, conhecer em detalhes os itens que podem ser comprados durante o jogo e trabalhar bem em equipe.

Os trabalhos citados aqui envolvem os jogos DotA, Dota 2 e LoL, que são os maiores títulos do gênero e são muito parecidos. Devido às similaridades, a grande maioria dos experimentos podem ser utilizados em outros títulos do gênero sem muitas adaptações, e muitas vezes até os resultados podem ser aproveitados.

De modo geral, as pesquisas envolvendo o gênero MOBA procuram entender como os diferentes aspectos do jogo influenciam no desempenho dos jogadores e dos times. Pobiedina et al. (2013) fizeram um estudo para identificar os vários fatores que influenciam o resultado no Dota 2 como, por exemplo, a experiência dos jogadores, escolha dos personagens e amizade entre os jogadores. Sapienza, Goyal e Ferrara (2018) trabalharam em entender a influência que um jogador pode ter na performance de outro e em criar um framework para recomendar jogadores para formar um time.

3.1.1 Posicionamento na arena

Em um jogo MOBA, normalmente, as duas equipes possuem o domínio inicial sobre uma região do mapa. Ao decorrer da partida os territórios vão sendo dominados e enfraquecidos. Criar a oportunidade de destruir parte do território inimigo e de iniciar um combate surpresa e com vantagem numérica depende muito da tática de posicionamento da equipe e conhecimento do posicionamento inimigo.

Rioult et al. (2014) selecionaram algumas medidas topológicas em relação ao posicionamento dos jogadores dentro da arena e analisaram a relevância dessas medidas no resultado da partida de DotA. Drachen et al. (2014) analisaram as diferenças em medidas de posicionamento entre quatro diferentes níveis de habilidade e entre os vencedores e perdedores nas partidas de Dota 2. Os autores realizaram, ainda, um agrupamento das séries temporais da distância média entre os jogadores do time, para identificar padrões de comportamento nos quatro níveis.

3.1.2 Combates

O combate direto entre os jogadores adversários são um dos principais meios de se obter vantagem dentro de uma partida de MOBA. Um dos grandes desafios é estabelecer uma forma de representar as ações de combate dos jogadores e suas consequências em termos de estratégia de equipe. Uma abordagem utilizada é a modelagem da interação entre os jogadores como grafos que podem ser divididos em relação ao tempo de partida.

Yang, Harrison e Roberts (2014) modelaram os combates no Dota 2 dividindo-os cronologicamente em poucos grafos que cobrem toda a partida. A partir desses grafos, foram geradas diversas características genéricas. Essas características foram filtradas na construção de uma árvore de decisão. A árvore, então, foi utilizada para gerar sequências de regras que indicam vitória e, dos grafos originais, foram extraídos subgrafos que respeitam essas regras e, dessa forma, representam comportamentos importantes para o bom desempenho da equipe em diferentes momentos de uma partida.

Schubert, Drachen e Mahlmann (2016) montaram um modelo mais complexo denominado *encounter* que representa um combate inteiro sem divisões fixas do tempo da partida. *Encounters* são mais complexos tanto em relação às interações entre os personagens (nós) quanto na relação temporal entre os grafos. Foi feita a análise de métricas que melhor caracterizam o início de um *encounter*, os resultados e o desempenho de cada jogador durante o encounter. Além disso, foi estudado o quanto os *encounters* são capazes de representar os acontecimentos mais importantes da partida.

3.1.3 Áreas hostis (selva)

Áreas hostis nas arenas são um ponto em comum na grande maioria dos jogos MOBA. Essas regiões são, normalmente, chamadas de selva e ocupam os espaços entre as rotas (como se estradas tivessem sido criadas em uma região de selva). A selva é importante por dois principais motivos: 1. possui recursos importantes que devem ser explorados durante toda a partida; 2. necessária para se transitar de uma rota para a outra. É comum nas equipes possuir um jogador (caçador) que tem como principal função estar na selva, explorando os recursos e realizando a maior quantidade de ataques surpresa aos adversários que estão nas rotas. A selva é extensa e os recursos são espalhados, as trajetórias de exploração escolhidas pelos jogadores são muito importantes para maximizar os recursos ganhos, tentar atrapalhar o caçador adversário e ajudar os companheiros nas rotas. Batsford (2015) trabalhou na modelagem da selva do Dota 2 e utilizou rede neural e algoritmo genético para tentar otimizar a ordem em que se deve explorá-la.

3.1.4 Funções dentro da equipe

Gao et al. (2013) e Eggert et al. (2015) realizaram trabalhos sobre entendimento e predição das funções dos jogadores no Dota 2 utilizando dados da partida. Os principais dados utilizados foram sobre o posicionamento dos jogadores durante a partida.

3.1.5 Previsão de vitória

Apesar da imprevisibilidade das partidas, muito é decidido antes do "apito inicial", e os acontecimentos "dentro de campo" tendem a refletir o nível de preparação de cada jogador e equipe. Em um MOBA, muita coisa deve ser definida logo antes da partida, no que pode ser chamado de *draft* (termo comum ao Dota 2) ou *picks and bans* (mais usado no LoL). No *draft* cada jogador normalmente deve decidir, entre outras coisas, a função que irá exercer e o personagem que irá utilizar durante a partida. Nesses dois jogos, por exemplo, existem mais de 100 personagens que podem exercer diferentes papéis.

A escolha da função leva em consideração a experiência dos jogadores. Na maioria das vezes, o jogador tem uma ou duas funções preferidas que ele se sente mais confiante em exercer. A escolha do personagem vai levar em consideração a função escolhida, o conhecimento do jogador e os outros personagens escolhidos pelo seu time e pela equipe

Reference	Model	Training (+ Validation) set size	Validation set size	Test Accuracy
Conley & Perry 2013	Logistic Regression	56691	5669	69,80%
	kNN	50000	6691	67,43%
Agarwal & Pearce 2014	Logistic Regression	40000	4000	62,00%
	PCA	40000	4000	57,00%
Song et al. 2015	Logistic Regression	6000	600	58,00%
Kalyanaraman 2014	Logistic Regression	18500	1500	69,42%
	Genetic Algorithm & Logistic Regression	220		74,10%
Kinkade & Lim 2015	Logistic Regression	62000	5000	72,90%
	Random Forest	62000	5000	67,00%

Figura 3.1: Tabela de resultados de trabalhos em predição de vitória (SEMENOV et al., 2016)

adversária.

Diversas pesquisas visam entender a influência dessas experiências e escolhas prévias nos resultados das partidas. O primeiro trabalho em predição de resultados a partir do *draft* foi desenvolvido por Conley e Perry (2013). Nesse trabalho com Dota 2, foi utilizado apenas os personagens escolhidos por cada time como base de conhecimento. Ainda assim conseguiram uma acurácia entre 65% e 70% na predição dos resultados.

Agarwala e Pearce (2014) tentaram explicitamente levar em consideração a interação entre os personagens e tornar o conjunto de características mais complexo. Para isso, foi somado às escolhas do personagens um conjunto de informações geradas com *Principal Component Analysis* (PCA) sobre um conjunto de estatísticas médias dos personagens escolhidos pelos dois times da partida. No entanto os resultados foram piores que os experimentos com apenas os personagens escolhidos como características. Os modelos treinados com PCA tiveram 57% de acurácia, enquanto que os modelos utilizando todas as informações dos personagens teve acurácia de 62%.

Song, Zhang e Ma (2015) tentaram resolver o problema das interações entre os personagens adicionando manualmente informações sobre a presença de 2 personagens específicos, e fez filtragem das características para selecionar aqueles que mais influenciavam os resultados. Apesar da seleção, foi conseguido apenas 54% de acurácia.

Kalyanaraman (2014) combinou a predição utilizando uma regressão simples, apenas com a lista de personagens escolhidos assim como Conley e Perry (2013), com uma estratégia utilizando algoritmo genético para melhorar os resultados. Obtendo 69% de acurácia apenas com a regressão e 74% com a combinação das duas estratégias.

Kinkade e Lim (2015) representaram a combinação de personagens aliados e oponentes com dois valores, um que indica a diferença de sinergia entre o conjunto de personagem dos dois times e o outro que representa as vantagens dos personagens de uma equipe sobre a outra. Nesse trabalho também foi realizado predição com os dados das partidas, como experiencia, dinheiro adquirido, abates, assistências e mortes de cada jogador das duas equipes. Essa é uma outra linha de pesquisa explorada também por Johansson e Wikström (2015).

A Figura 3.1 mostra um resumo de alguns resultados de predição de vitória pelo *draft* no Dota 2 (SEMENOV et al., 2016). Os resultados mostrados podem parecer baixos, mas eles na verdade só confirmam a imprevisibilidade do decorrer da partida.

Resultados muito próximos ao ótimo indicariam que os acontecimentos da partida não exercem grande influência no seu resultado.

Acredita-se que existe um fator muito importante faltando nesses trabalhos: a experiência dos jogadores. Muitas vezes é melhor escolher um personagem com o qual esteja familiarizado do que um que, supostamente, é um dos melhores no momento ou que combine bem com os outros escolhidos. Mesmo considerando partidas com os jogadores mais experientes e habilidosos esses jogadores não possuirão a mesma habilidade com todos os personagens e funções.

Sugere-se que as características sejam compostas, então, por informações dos personagem escolhidos, estatísticas de cada personagem e da interação entre eles e dados sobre por exemplo o desempenho de cada integrante do time na função a ser desempenhada e com o personagem escolhido. Vale destacar que a coleta de informações sobre a experiência dos jogadores é mais complicada, mas totalmente viável. Alguns fãs já fizeram experimentos de predição levando em consideração a experiência dos jogadores, como Yin (2018).

3.1.6 Jogadores artificiais

Na maioria dos e-sports, existe a possibilidade do jogador enfrentar um adversário artificial em um modo de jogo mais amistoso, podendo-se determinar o nível de habilidade do adversário. Um dos desafios de pesquisa é criar uma inteligência artificial capaz de jogar com alto desempenho, mas de maneira justa. Silva e Chaimowicz (2015) desenvolveram uma arquitetura de agentes inteligentes para MOBA que utiliza mapas de influência e que foi testada no League of Legends. A empresa *OpenAI* iniciou em 2016 um projeto que utiliza Dota 2 como plataforma de teste para estudos de inteligência artificial de propósito geral (OPENAI, 2016). Em 2018 o time artificial do projeto (OpenAI Five) conseguiu vencer jogadores profissionais de Dota 2.

3.1.7 Processamento de imagens

No League of Legends, apesar da API, não existe uma forma fácil de conseguir dados detalhados das ações dos jogadores durante a partida como o posicionamento a cada instante. Farza (2018) utilizou aprendizado profunda para ensinar um modelo a identificar os personagens da partida e seus posicionamentos a partir de imagens do minimapa.

Capítulo 4

MÉTODOS

4.1 ALGORITMOS DE AGRUPAMENTO

O processo de Agrupamento é uma técnica de aprendizado de máquina não supervisionado, que tem como objetivo organizar os dados analisando a similaridade entre os seus atributos (XU; WUNSCHANDERS, 2008). Nesse tipo de aprendizado, normalmente, não existe um conhecimento sobre a classe dos dados. O agrupamento é então uma forma comum para encontrar estruturas presentes nos dados.

Existem várias abordagens para realizar agrupamentos. As quais podem diferir na forma de analisar a similaridade entre os dados. A escolha dos métodos depende do entendimento do problema e do tipo de estrutura desejado.

4.1.1 Agrupamento por densidade

Agrupamento baseado em densidade parte da ideia de formar grupos em áreas com uma maior densidade de objetos (XU; WUNSCHANDERS, 2008). Nesse tipo de agrupamento, áreas de baixa densidade irão separar os grupos. O algoritmo utilizado neste trabalho é o DBSCAN (ESTER et al., 1996). Este algoritmo utiliza um modelo de alcance onde áreas com uma quantidade mínima de objetos são expandidas, conectando novos objetos que estão dentro de um limite máximo de distância. No DBSCAN, normalmente, não se fixa o número de grupos desejados. Os parâmetros principais relacionam a quantidade mínima de objetos para formar um grupo e a distância máxima para considerar 2 objetos vizinhos.

Cada objeto é testado para identificar uma região de alta densidade ao seu redor, dando início a um grupo. Caso exista essa região será expandida ao máximo antes de tentar criar um outro grupo. Dessa forma, esse método busca identificar áreas de baixa densidade para delimitar grupos. O agrupamento por densidade propicia formação de grupos com formatos arbitrários (figura 4.1) por expandi-los através das bordas.

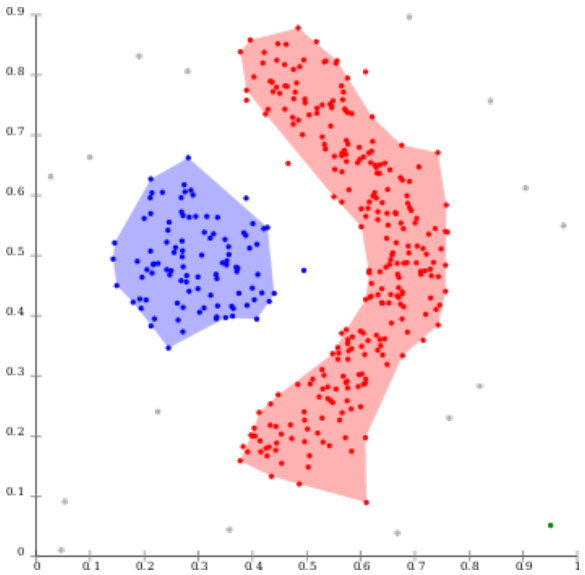


Figura 4.1: Tendência do DBSCAN

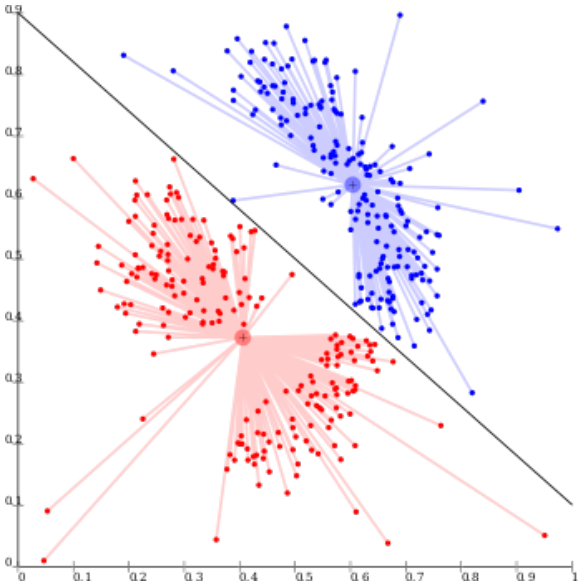


Figura 4.2: Tendência do K-means

4.1.2 Agrupamento Particional

Nessa abordagem, os grupos são representados por um centro que pode ser, por exemplo, um dos dados pertencentes ao grupo ou a média de todos os dados daquele grupo (XU; WUNSCHANDERS, 2008). Ao fixar a quantidade de grupos desejados (k), tem-se um problema de otimização para encontrar uma partição que minimiza a distância quadrática (E) dos objetos ao centro do seu grupo. Esse problema pode ser formalizado da seguinte maneira, considerando c_j como a média entre os dados e centro do grupo C_j:

$$E = \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - c_j\|^2 ,$$

$$c_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i .$$

O algoritmo de particionamento utilizado neste trabalho foi o k-means (MACQUEEN, 1967). A inicialização dos centros é a primeira parte do algoritmo. Nesse caso, são escolhidos aleatoriamente k objetos da base. Então, iterativamente os dados são realocados aos grupos mais próximos e os centros são atualizados com a média dos valores das instâncias do novo grupo. Esse processo ocorre até não haver mais mudanças.

Esse algoritmo tende a gerar grupos de com formatos esféricos devido à minimização da distância dos objetos ao centro (Figura 4.2).

4.1.3 Agrupamento Fuzzy

Uma abordagem diferente para utilização de centros é considerar que um dado não pertence simplesmente a um grupo, mas que possui uma taxa de similaridade relacionada

a todos os grupos. Neste trabalho adotou-se o algoritmo fuzzy c-means (XU; WUNSCHANDERS, 2008).

No fuzzy c-means, pode-se estabelecer uma matriz de pertinência (U), em que U_j é um vetor que indica a pertinência de cada objeto ao grupo j. Iterativamente, os centros são recalculados com a equação:

$$c_j = \frac{\sum_{i=1}^{n}[U_j(x_i)]^m x_i}{\sum_{i=1}^{n}[U_j(x_i)]^m}.$$

Então, os valores de pertinência são atualizados seguindo:

$$U_j(x_i) = [\sum_{k=1}^{c}(\frac{\|x_i - c_j\|^2}{\|x_i - c_k\|^2})^{\frac{1}{m-1}}]^{-1}.$$

Mantendo-se a restrição:

$$\sum_{j=1}^{c} U_j(x_i) = 1.$$

Esse processo é repetido até que não existam mudanças significativas nas pertinências. A execução é iniciada com uma aleatorização de U. No cálculo dos centros, é levado em consideração um argumento m que representa o quão fuzzy será o agrupamento. Por fim, deseja-se minimizar a seguinte função objetivo:

$$\arg\min_{C} \sum_{i=1}^{n}\sum_{j=1}^{c} U_j(x_i) \|x_i - c_j\|^2.$$

Nos experimentos realizados, ao fim da execução do algoritmo cada objeto foi associado ao grupo ao qual possui o maior grau de pertinência.

4.1.4 Agrupamento Hierárquico

Esse tipo de agrupamento baseia-se em criar uma hierarquia de grupos (XU; WUNSCHANDERS, 2008). Duas estratégias opostas podem ser utilizadas. Na aglomerativa, cada instância pode ser considerada, inicialmente, como um grupo e, a cada nível, dois grupos são unidos. Na divisiva, todos os dados formam um único grupo que irá se dividir a cada passo do algoritmo. É comum a utilização de dendrogramas para visualizar a hierarquia dos grupos.

Neste trabalho, foi utilizada a abordagem aglomerativa. Além da escolha da medida de distância, como Euclidiana ou de Manhattan, ao utilizar essa técnica existe um ponto muito importante que é o critério para medir a proximidade entre dois grupos. A cada passo, os dois grupos mais próximos seguindo esse critério serão unidos. Os critérios aplicados nos experimentos deste trabalho são:

- **Single-link:** A distância entre dois grupos é a menor distância entre elementos dos dois grupos. Formalmente:

$$d(C_1, C_2) = \min_{x_i \in C_1, x_j \in C_2,} \|x_i - x_j\| .$$

Permite a formação de grupos com formatos mais arbitrários.

- **Complete-link:** A distância entre dois grupos é a maior distância entre elementos dos dois grupos. Formalmente:

$$d(C_1, C_2) = \max_{x_i \in C_1, x_j \in C_2,} \|x_i - x_j\| .$$

Muito sensível a ruídos (objetos muito destoantes).

- **Average-link:** A distância entre dois grupos é distância média entre elementos dos dois grupos. Formalmente:

$$d(C_1, C_2) = \frac{1}{|C_1|\,|C_2|} \sum_{x_i \in C_1, x_j \in C_2,} \|x_i - x_j\| .$$

- **Método Ward:** A cada iteração serão unidos o par de grupos que resulte numa partição que minimize o valor de uma função objetivo. A função objetivo no caso é a soma dos erros quadráticos, ou seja a soma das distâncias entre todos os pares de objetos de um mesmo grupo. Esse critério é também chamado de variância mínima de Ward ou variância intragrupo. Esse método tende a formar grupos compactos esféricos e com tamanhos similares, como os grupos obtidos pelo k-means.

Os algoritmos k-means, fuzzy c-means e o DBSCAN, por possuírem fatores de aleatoriedade, foram sempre executados múltiplas vezes nos experimentos. A quantidade de repetições mais utilizada foi 10.

4.2 VALIDAÇÃO DE AGRUPAMENTO

A análise dos resultados dos experimentos é um dos passos mais importantes da pesquisa. Existem diversas técnicas para validação de agrupamentos, que dependem também do objetivo da análise. Nas próximas seções, são apresentadas as abordagens utilizadas na validação dos resultados obtidos neste trabalho.

4.2.1 Critério externo

Neste trabalho, foram utilizadas avaliações externas para comparar os agrupamentos dos personagens com as classes e subclasses estabelecidas pelo jogo. Os índices Rand (R), Jaccard (J), Fowlkes Mallows (FM) e Hubert normalizado (Γ) (XU; WUNSCHANDERS, 2008) foram utilizados para esse tipo de avaliação. Todos os índices são baseados nos contadores descritos na Tabela 4.1, considerando o agrupamento do experimento α e o agrupamento do jogo β.

Tabela 4.1: Contatores utilizados nos índices para avaliação de critério externo

Variável	Descrição
VP	Número de pares de objetos que pertencem ao mesmo grupo em α e em β (verdadeiros positivos)
FP	Número de pares de objetos que pertencem ao mesmo grupo em α e a grupos distintos em β (falsos positivos)
FN	Número de pares de objetos que pertencem a grupos distintos em α e a ao mesmo grupo em β (falsos negativos)
VN	Número de pares de objetos que pertencem a grupos distintos em α e em β (verdadeiros negativos)

- **Rand (R):** Mede a probabilidade de dois objetos pertencerem ao mesmo grupo ou grupos diferentes nas duas partições, com valores entre 0 e 1, segundo a equação:

$$R(\alpha, \beta) = \frac{VP + VN}{VP + FP + FN + VN}.$$

- **Jaccard (J):** Avalia o agrupamento considerando objetos que estão nos mesmos grupos em ambas partições, com valores entre 0 e 1, segundo a equação:

$$J(\alpha, \beta) = \frac{VP}{VP + FP + FN}.$$

- **Fowlkes e Mallows (FM):** Mede a semelhança entre duas partições, com valores entre 0 e 1, segundo a equação:

$$FM(\alpha, \beta) = \frac{VP}{\sqrt{(VP + FP)(VP + FN)}}.$$

- **Hubert normalizado (Γ):** Calcula a correspondência linear entre duas partições, com valores entre -1 e 1, segundo a equação:

$$\Gamma(\alpha, \beta) = \frac{(VP + FP + FN + VN)VP - (VP + FP)(VP + FN)}{\sqrt{(VP + FP)(VP + FN)(FN + VN)(FP + VN)}}.$$

4.2.2 Critério relativo

A avaliação de critério relativo tem o objetivo de analisar a qualidade do agrupamento para, por exemplo, comparação entre métodos e ajuste de parâmetros. Analisa-se, por exemplo, a similaridade *intra grupos* e o distanciamento *inter grupos* (XU; WUNSCHANDERS, 2008).

Foi utilizado neste trabalho, o índice silhueta (s) que mede a similaridade dos dados de um mesmo grupo em contraste com a similaridade com dados de outros grupos. O índice é calculado a partir da equação:

$$s(x_i) = \begin{cases} 1 - \frac{a(x_i,C_i)}{b(x_i,C_i)} & \text{if } a(x_i,C_i) < b(x_i,C_i) \\ 0 & \text{if } a(x_i,C_i) = b(x_i,C_i) \\ \frac{b(x_i,C_i)}{a(x_i,C_i)} - 1 & \text{if } a(x_i,C_i) > b(x_i,C_i) \end{cases}.$$

Sendo que:

$$a(x_i, C_i) = \frac{1}{|C_i|} \sum_{x_i, x_j \in C_i, x_i \neq x_j} d(x_i, x_j),$$

e

$$b(x_i, C_i) = \min_{C_j \in C, C_j \neq C_i} a(x_i, C_j).$$

4.3 APRENDIZADO DE MÁQUINA SUPERVISIONADO

O aprendizado supervisionado consiste no aprendizado utilizando exemplos com a saída (rótulo) conhecidos (MITCHELL, 1997). Deseja-se obter uma função de mapeamento que consiga realizar a tarefa de responder a saída correta dada uma entrada desconhecida. Para isso o modelo deve aprender a generalizar a tarefa a partir dos dados de treinamento.

O sobreajuste ocorre quando o modelo aprende demais com o conjunto de treinamento (sem generalização) (MITCHELL, 1997). Nesse caso o modelo consegue mapear os exemplos utilizados no treinamento muito bem, mas erra bastante na classificação de novos exemplos. Uma ótima forma de detectar e evitar o sobreajuste na hora de treinar um modelo é separar parte dos dados com rótulos conhecidos, sem utilizá-los no treinamento, e aplicar esse subconjunto para teste e validação do desempenho do modelo com dados desconhecidos.

Os experimentos que envolvem aprendizado supervisionado neste trabalho tem o objetivo de predizer o time ganhador de uma partida de League of Legends. Foram utilizados os algoritmos Random Forest e Redes Neurais, com o auxilio do *framework TensorFlow* (ABADI et al., 2015). As próximas seções discutem um pouco os dois algoritmos de aprendizado.

4.3.1 Random Forest

Um dos algoritmos utilizados no problema da predição do resultado das partidas foi o Random Forest (Floresta aleatória) (BREIMAN, 2001). Random Forest é um algoritmo para regressão e predição formado por um conjunto de árvores de decisão (*ensemble learning*). O valor de saída do modelo é achado a partir da união das respostas de todas as árvores, que pode ser, por exemplo, através da moda ou da média dessas respostas.

Árvore de decisão é um dos algoritmos mais conhecidos e utilizados para aprendizado de máquina (MITCHELL, 1997). Em geral, uma característica negativa das árvore de decisões é o fácil sobreajuste do modelo em relação aos dados de treinamento.

A estratégia do Random Forest é utilizar a potencialidade das árvores de decisão evitando o sobreajuste. Para isso diversas árvores são treinadas com diferentes subconjuntos dos dados de treinamento (bootstrap aggregating). Além disso, no treinamento de cada árvores, podem ser escolhidos subconjuntos das características como candidatas para expansão, momento em que se utiliza, por exemplo a medida *information gain*. Esse método induz uma maior variabilidade entre as árvores, que caso contrário seriam muito correlacionadas ou até iguais. O principal parâmetro a ser ajustado na random forest é a quantidade de árvores, além de parâmetros relacionados ao treinamento de cada árvore de decisão.

4.3.2 Redes Neurais Artificiais

Rede neural artificial é um estrutura de aprendizado formada por camadas de nós que representam "neurônios" conectados entre si para passar e transformar dados (MITCHELL, 1997). Em uma rede neural *feedfoward*, a primeira camada é representada pela entrada dos dados, que serão passados adiante pelas camadas escondidas até chegar na última que deverá dar como saída uma resposta no formato esperado para a tarefa objetivo. As camadas formam uma série de funções que compõem a função total do modelo (GOODFELLOW; BENGIO; COURVILLE, 2016). A quantidade de camadas de uma rede neural indica a sua profundidade e a partir da utilização de redes mais profundas surgiu o termo aprendizado profundo (deep learning) (GOODFELLOW; BENGIO; COURVILLE, 2016).

Cada nó na rede neural irá receber dados da camada anterior ponderados por pesos. Uma rede Multilayer Perceptron (MLP) é formadas por camadas totalmente conectadas, de forma que um nó recebe como entrada dados vindo de todos os nós da camada anterior. Nessas unidades, os dados são compilados em apenas uma saída a partir de uma função de ativação, por exemplo, *sigmoid* ou *softmax*. Neste trabalho são utilizadas redes MLP.

A *feedfoward* é base para a formação de outras arquiteturas. Quando a saída pode ser usada para realimentar a rede, temos uma rede neural de recorrência que é muito utilizada para reconhecimento de linguagem natural. Para processamento de imagens são muito utilizadas as redes de convolução (Convolutional Neural Network (CNN)), que possuem camadas de convolução.

4.3.2.1 Aprendizado da rede neural Para que a inferência feita pela rede seja otimizada em relação ao objetivo, a rede precisa ser treinada, o que significa ajustar os pesos das conexões entre as camadas utilizando os dados de treinamento. Esse ajuste dos pesos pode ser feito levando em consideração alguma função de erro (a diferença entre a predição e os resultados esperados). O método de aprendizado com função de erro comumente usado é o *Gradient Descent*. Os gradientes podem ser calculados utilizando o algoritmo de *back-propagation* (GOODFELLOW; BENGIO; COURVILLE, 2016).

Existem diversas técnicas de otimização para o aprendizado de uma rede como: *Mo-*

mentum RMSProp, *AdaGrad* e *Adam* (THUSHV, 2017). Elas influenciam diretamente a atualização dos pesos de uma rede, entre os passos de seu treinamento, a fim de acelerar a procura de soluções otimizadas. Essas técnicas modificam por exemplo, a utilização da taxa de aprendizado, que define quanto os pesos vão "aprender" a partir do gradiente em uma iteração do treinamento.

Um grande desafio em redes neurais é evitar o sobreajuste e assim conseguir bons resultados, não só com o dados de treino, mas também com exemplos desconhecidos. Para isso, existem os métodos de regularização como, *dropout*, *batch normalization* e *weight decay*, que ajudam a manter a generalização no aprendizado da rede (GOODFELLOW; BENGIO; COURVILLE, 2016). O *batch normalization* ajuda a estabilizar a rede, ao normalizar a saída de uma camada para depois usá-la como entrada para a próxima (DOUKKALI, 2017). Normalização é comumente usada nos dados antes de serem utilizados em um modelos de aprendizado, o *batch normalization* aplica essa ideia entre as camadas da rede neural. Essa técnica tem efeito de regularização, por aplicar ruído às ativações das camadas.

O *dropout* é usado também para gerar ruído e penalidade para a função de *loss* e, portanto, também ajuda a evitar o sobreajuste. Nessa técnica a cada passo do treinamento de uma rede cada nó da camada aplicada pode ser descartado com uma certa probabilidade. *Weight decay* adiciona um decaimento na atualização dos pesos de uma rede. Essa técnica força os pesos a se aproximarem de 0 (zero), e assim evitar a influência de pesos muito grandes.

Técnicas de otimização e regularização são comparadas nos experimentos do Capítulo 6.

4.3.3 Dados Categóricos

A maioria dos métodos de aprendizado de máquina necessitam ou preferem dados numéricos. Dessa forma, quando se deseja utilizar um dado não numérico, é preciso fazer algum tipo de transformação. Esses dados são chamados de categóricos, e um exemplo deles é o personagem escolhido por cada jogador.

Uma estratégia para representação de dados categóricos é chamada *one hot enconder*. O campo categórico é transformado em um vetor, no qual cada posição representa um dos possíveis valores originais e terá o valor 1 (um) se for o corresponde ao valor do exemplo, e 0 caso contrário. Esse método aumenta a dimensionalidade dos objetos com a quantidade de valores possíveis para aquele campo, o que pode ser muito, como no caso dos campeões que são mais de 140 (cento e quarenta).

Para evitar esse problema foi utilizada a técnica de *embedding* (TENSORFLOW, 2018), que também mapeia o campo para um vetor, mas com um tamanho escolhido e possivelmente com valores diferentes de 0 ou 1. Esses valores serão treinados juntamente com o resto do modelo para melhor representar a informação original. Um padrão para o valor escolhido é a raiz quarta da quantidade de valores possíveis para o dado.

Capítulo 5

EXPERIMENTOS DE AGRUPAMENTO

Neste capítulo são discutidos diversos experimentos e análises em cima de dados do estáticos e de partidas reais do LoL. O foco foi a utilização de diversas técnicas de agrupamento, com a análise da qualidade desses grupos (avaliação relativa) e comparação com rótulos pré estabelecidos (avaliação externa).

5.1 DADOS DO JOGO

As classes e subclasses, que foram explicadas com mais detalhes no Capítulo 2, possuem muita relação com os atributos e as habilidades dos personagens. Esta seção descreve os experimentos de agrupamento dos personagens, levando em consideração esses dois elementos e a comparação desses grupos com as classes e subclasses pré-estabelecidas.

A base para os experimentos desta etapa foi formada de um registro para cada campeão, totalizando 141 (cento e quarenta e um) objetos na base, e 127 (cento e vinte e sete) sem os especialistas. Os especialistas foram utilizados no agrupamento, mas deixados de fora de análises, como a avaliação externa, por não formarem um grupo de personagens semelhantes. Cada objeto foi formado inicialmente pelos atributos básicos do personagem correspondente. Esses dados foram coletados da API do League of Legends. Inicialmente eram 20 (vinte) valores de atributos. A partir de processamento e remoção de dados, a quantidade de atributos na base passou a ser 9 (nove).

As habilidades são o componente principal na caracterização de um personagem. Os dados até então na base não dão nenhuma informação sobre as habilidades e uma forma encontrada para leva-las em consideração foi de se utilizar alguns dados estimados fornecidos na própria API e em uma wiki do jogo (FANDOM, 2018). Esses dados são, por exemplo: força, resistência e agilidade do personagem que levam em consideração os atributos básicos e efeitos das habilidades. Com essas adições, o número total de características na base passou a ser 19 (dezenove).

	Classes				Subclasses				
Método	R	J	FM	Γ	R	J	FM	Γ	Base
K-means	0.753	0.275	0.442	0.293	0.829	0.167	0.296	0.204	**9**
	0.871	0.482	0.652	0.574	0.872	0.266	0.428	0.359	**19**
Fuzzy 1.2	0.794	0.271	0.426	0.301	0.833	0.158	0.279	0.189	**9**
	0.852	0.459	0.636	0.548	0.907	0.335	0.503	0.452	**19**
Single	0.204	0.171	0.407	0.011	0.610	0.163	0.375	0.244	**9**
	0.204	0.171	0.407	0.011	0.191	0.090	0.283	0.004	**19**
Complete	0.647	0.287	0.503	0.331	0.725	0.152	0.308	0.183	**9**
	0.731	0.298	0.483	0.330	0.880	0.334	0.517	0.456	**19**
Average	0.209	0.174	0.412	0.040	0.613	0.160	0.367	0.234	**9**
	0.204	0.171	0.407	0.011	0.803	0.283	0.505	0.429	**19**
Ward	0.737	0.262	0.428	0.271	0.840	0.169	0.295	0.208	**9**
	0.872	0.497	0.667	0.590	0.896	0.366	0.546	0.492	**19**

Tabela 5.1: Resultados da avaliação externa

5.1.1 Avaliação externa com classes e subclasses

Os experimentos utilizando os dados estáticos foram realizados com *K-means*, *Fuzzy C-means* e com agrupamento hierárquico. Os resultados mostram que, tanto na comparação de classes e subclasses, a base com a adição dos dados estimados consegue representar melhor o que foi estabelecido pelo jogo (Tabela 5.1).

Mesmo com todo o conjunto de características os agrupamentos não obtiveram uma aproximação tão alta das classes e subclasses originais. O índice *Rand* obteve valores elevados em quase todos os experimentos, pois ele leva em consideração as duplas que foram corretamente postas em diferentes grupos (verdadeiros negativos) que podem ser muitos mesmo com agrupamentos muito diferentes.

Os resultados mostram o quanto as separações das classes, que são grupos mais genéricos, são mais claras do que das subclasses. Na comparação entre os algoritmos, o *K-means*, o *Fuzzy C-means* e o método Ward para agrupamento hierárquico tiveram, em geral, os melhores resultados em relação à avaliação externa.

Com agrupamento hierárquico o método de similaridade Ward obteve a maior aproximação. O single-link obteve os piores resultados, o método gerou grupos com tamanhos muito destoantes (poucos grupos com quase todos os personagens). Pela característica do single-link isso indica que os dados são muito próximos e sobrepostos mesmo entre diferentes classes e subclasses.

5.1.2 Avaliação relativa

Além da avaliação com índices externos, utilizou-se a silhueta como critério relativo. Testes foram feitos fixando diferentes quantidades de grupos, entre a quantidade de classes e subclasses. No geral, os resultados foram baixos mostrando não existir estruturas de

Método	Classes	Subclasses
K-means	**0.215**	0.145
Fuzzy 1.2	0.177	0.129
Single	0.112	-0.143
Complete	0.116	0.172
Average	0.112	**0.178**
Ward	0.206	0.172

Tabela 5.2: Resultados da avaliação relativa - índice silhueta

similaridade muito distinguíveis nos dados. A Tabela 5.8 mostra os resultados da silhueta para 6 (seis) grupos (classes) e 12 (doze) grupos (subclasses) para todos os algoritmos.

Não houve grandes variações, utilizando diferentes números de grupos, para os experimentos mais bem sucedidos. O valor da silhueta tendeu a diminuir com o aumento no número de grupos. O Gráfico 5.1 mostra os valores do índice silhueta para os vários números de grupos.

As subclasses do League of Legends são especializações das classes, trazendo uma relação hierárquica entre os dois níveis. Intuitivamente um algoritmo de agrupamento hierárquico se encaixa muito bem para a análise desses dados. A Tabela 5.3 mostra uma descrição do agrupamento hierárquico com o método Ward com 6 (seis) grupos que obteve um dos melhores índice silhueta para essa quantidade de grupos entre todos os algoritmos, além de ter resultado em grupos com tamanhos mais balanceados. A Figura 5.2 mostra o agrupamento com os dois principais componentes gerados pelo método PCA.

O agrupamento discutido indica quais características e grupos de personagens são mais distinguíveis. A classes de personagens muito fortes porém com baixa resistência foram melhores separadas do resto, principalmente, a classe *Marksman* que é predominantemente de dano físico. As classes de personagens de combate a curta distância parecem ter mais sobreposição, principalmente a classe *Fighter* e *Slayer* que possuem mais força. A maioria dos personagens utilizados para suporte dentro do jogo são das classes *Tank* e *Controller*, que possuem respectivamente mais resistência e mais controle e utilidade. Essas classes foram bem separadas apesar de algumas sobreposições.

5.1.3 Personagens especialistas

Os personagens especialistas supostamente não se encaixam nas classes e subclasses existentes. Foram feitos alguns testes para analisar essa característica, em relação aos dados coletados sobre os personagens. A primeira avaliação foi feita com o algoritmo *Fuzzy C-means*, a hipótese testada foi de que os especialistas não eram tão similares a nenhum grupo quanto os outros personagens. O Gráfico 5.3a mostra uma comparação entre a média do maior grau de pertinência a um grupo por parte dos especialistas e dos outros personagens. Para os especialistas os valores foram predominantemente menores, apoiando a avaliação de que eles não se encaixam tanto a um grupo quanto os demais. Este gráfico também oferece uma boa visualização do efeito do parâmetro *fuzzy m* do *Fuzzy*

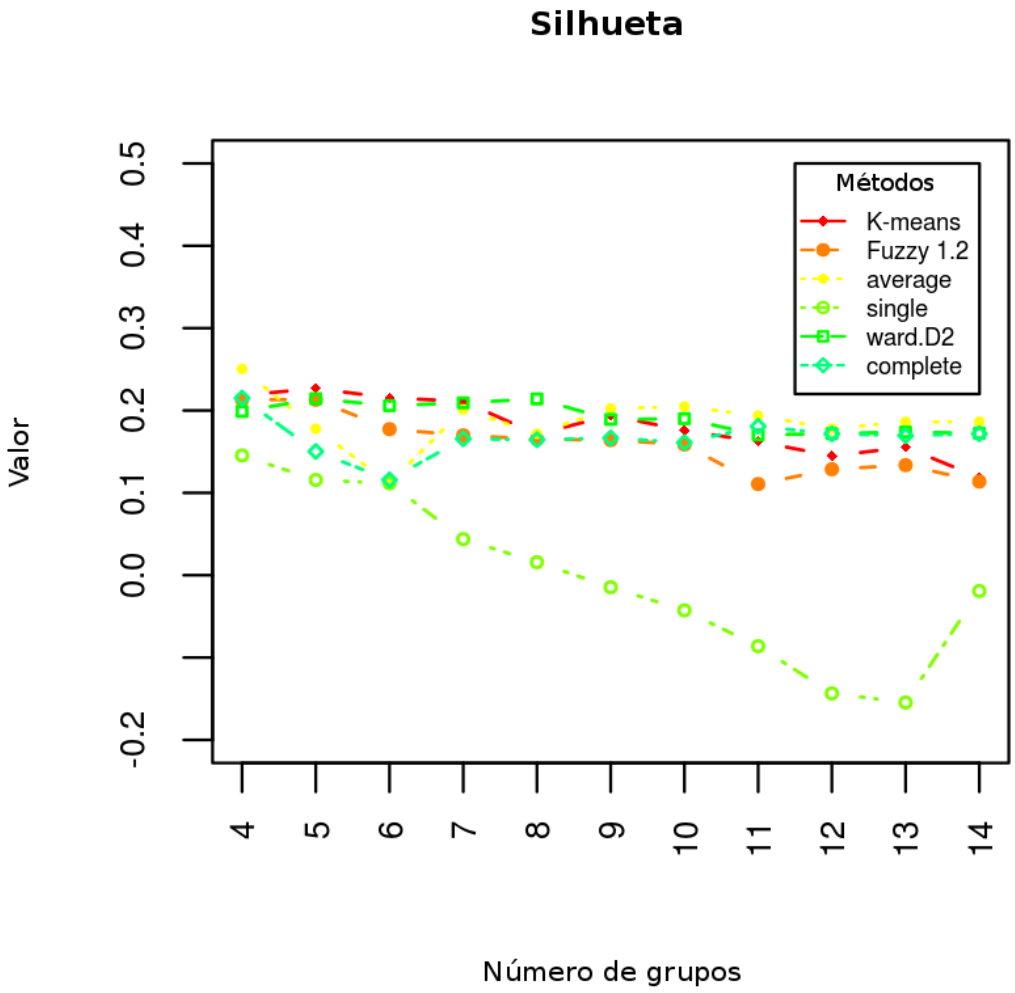


Figura 5.1: Silhueta do agrupamento de dados estáticos dos personagens

Tamanho	Características	Classes frequentes	Personagens
44	Personagens com força elevada. Resistência, controle e mobilidade acima da média. Utilidade fraca. Predominantemente ataque físico, corpo a corpo. Utiliza tanto ataques básicos quanto habilidades.	Fighter e Slayer	Renekton, Trundle, Zed
27	Personagens de dano mágico, totalmente dependentes das habilidades e de combate a distância. Força muito elevada. Controle elevado. Resistência, mobilidade e utilidade fracas.	Mage	Ziggs, Malzahar, Lux
10	Personagens predominantemente mágicos, muito dependentes das habilidades e de combate corpo a corpo. Muita força. Mobilidade elevada, resistência e controle moderados. Pouca utilidade.	Slayer, Tank	Akali, Ekko, Amumu
16	Grupo com a presença grande de personagens usados comumente para suporte, mas aqueles que são mais resistentes. Majoritariamente mágicos, muito dependentes das habilidades e combate corpo a corpo. Resistência e controle muito elevados. Pouca força, mobilidade e utilidade.	Tank, Controller	Braum, Nautilus, Taric
19	Grupo com todos e apenas os personagens da classe marksman (atirador). Personagens de dano predominantemente físico, pouco dependentes das habilidades para causar dano (uso de ataques básicos) e de combate a distância. Força muito alta. Mobilidade moderada. Controle e utilidade fracos. Resistência muito baixa.	Marksman	Draven, Ashe, Taric
11	Grupo com a presença predominante de personagens de suporte, aqueles com foco em utilidade. Personagens com foco em dano totalmente mágico, muito dependentes das habilidades e majoritariamente de combate a distância. Força e resistências baixas. mobilidade moderada. Controle elevado. Muita Utilidade.	Controller	Bardo, Nami, Soraka

Tabela 5.3: Descrição de agrupamento dos personagens

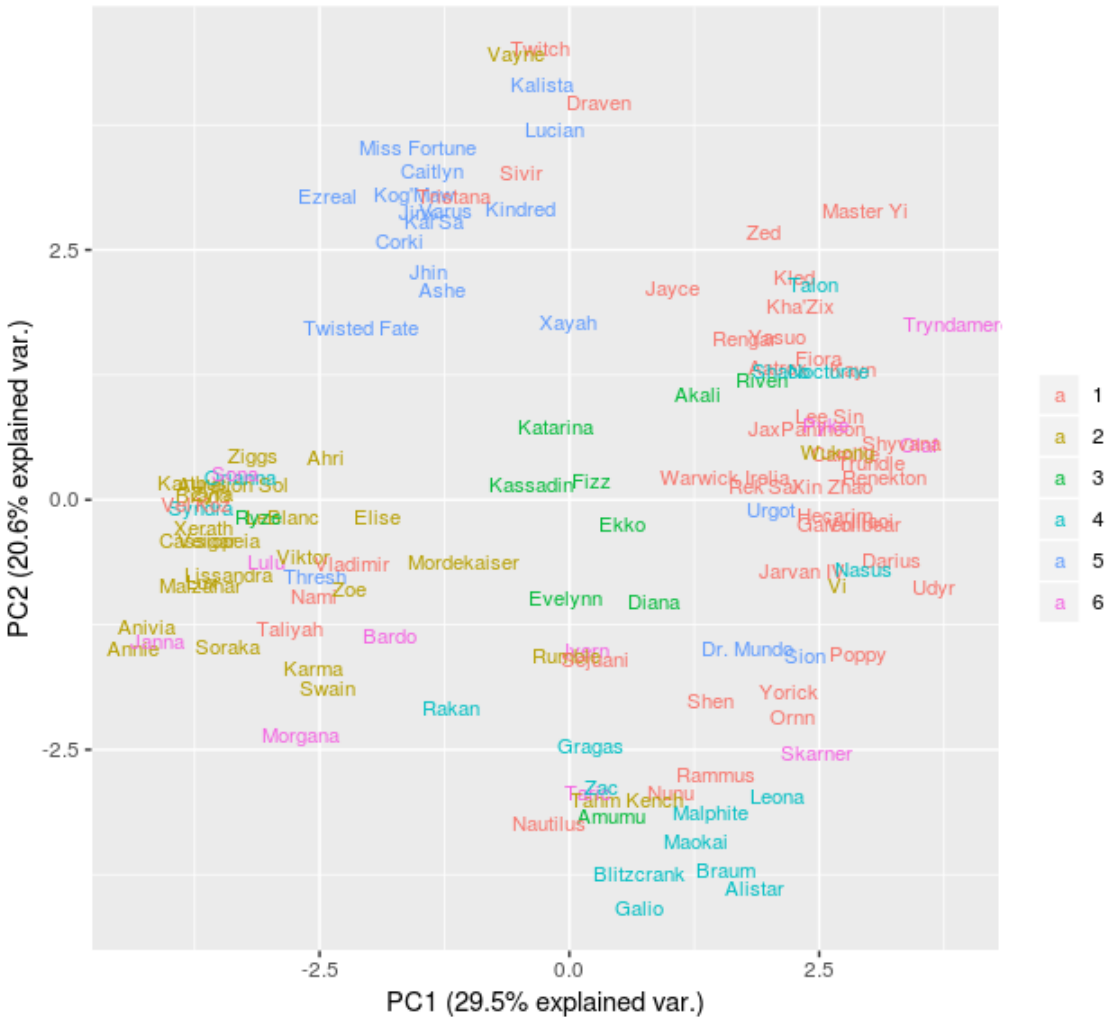
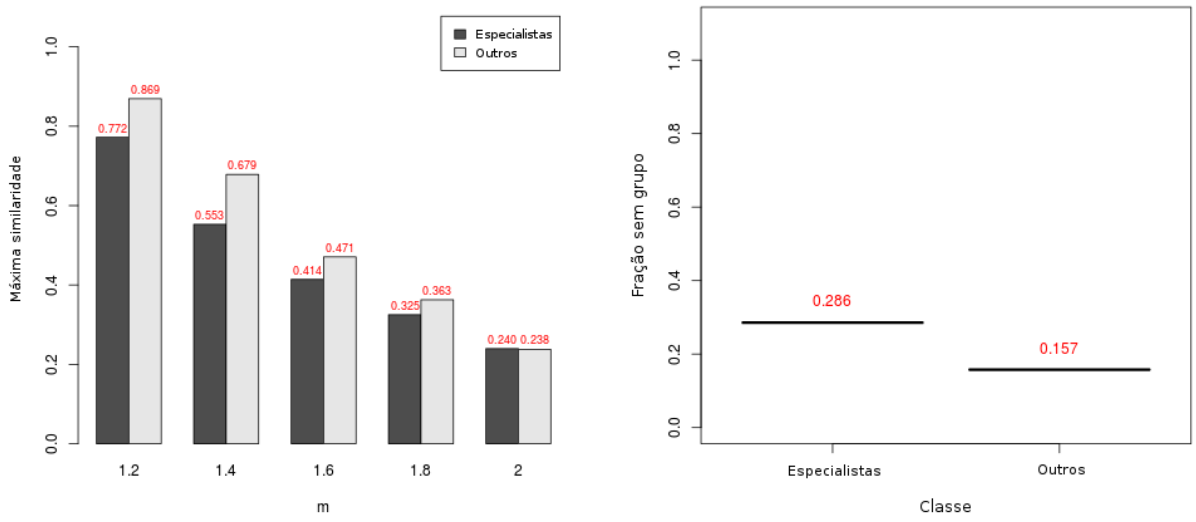


Figura 5.2: Visualização de agrupamento dos personagens com PCA



(a) Fuzzy C-means (b) DBSCAN

Figura 5.3: Comparação dos especialistas com os outros personagens

C-means.

Uma segunda análise foi feita com o algoritmo de agrupamento por densidade o DBSCAN. Pela sua arquitetura, existe a possibilidade de um objeto não ser associado a nenhum grupo. O Gráfico 5.3b mostra um exemplo em que a fração dos personagens especialistas sem grupo tende a ser maior que a dos outros personagens. Esse comportamento ocorreu em quase todos os experimentos, mas vários tiveram resultados muito ruins, com poucos grupos e grande quantidade de personagens sem grupos. O exemplo da figura foi uma configuração que conseguiu formar 5 (cinco) grupos e não deixou tantos personagens sem grupo.

5.2 DADOS DE PARTIDAS

Os criadores do League of Legends decidem os atributos dos personagens e suas habilidades, mas, basicamente tudo, depende do comportamento dos jogadores, e isso inclui, por exemplo: os itens comprados, quem atacar, onde se posicionar. Nesta etapa foram realizados experimentos de agrupamento com estatísticas de jogadores em partidas reais. Os resultados são novamente comparados com as classes e subclasses, e também com as funções dos membros do time na partida.

Para estes experimentos, foram coletados dados de partidas passadas e cada objeto da base representa as estatísticas de um jogador em uma partida. A base foi formada por 28 (vinte e oito) características como: quantidade de abates, mortes, assistências, dano causado, dano obtido e recursos obtidos. Esses dados conseguem cobrir a grande maioria dos elementos comuns a uma partida de LoL como: combates, destruição de estruturas,

	Classes				Subclasses			
Método	R	J	FM	Γ	R	J	FM	Γ
K-means	0.826	0.364	0.535	0.430	0.859	0.196	0.330	0.253
Fuzzy 1.2 / 1.4	0.849	0.393	0.565	0.473	0.890	0.231	0.375	0.315
Single	0.224	0.169	0.398	0.000	0.222	0.092	0.284	0.017
Complete	0.706	0.269	0.448	0.279	0.827	0.196	0.342	0.252
Average	0.250	0.165	0.383	-0.015	0.643	0.148	0.328	0.188
Ward.D2	0.792	0.335	0.509	0.385	0.844	0.185	0.319	0.235

Tabela 5.4: Avaliação externa, agrupamento com estatísticas médias por personagem

suporte, controle e visão do mapa.

Cada personagem foi representado com 12 (doze) registros na base, 6 em que o jogador estava no time ganhador e 6 (seis) no time perdedor. Quando os experimentos foram realizados existiam 139 (cento e trinta e nove) personagens (2 (dois) a menos que nos experimentos da última seção). Totalizando 1668 (um mil seiscentos e sessenta e oito) objetos na base, 1488 (um mil quatrocentos e oitenta e oito) sem os personagens especialistas. A grande maioria das partidas foram do Brasil, e o resto da região da América do Norte.

5.2.1 Classes e Subclasses:

Foi realizada uma seleção das características disponíveis para a análise em relação às classes e subclasses, além disso, foram desconsiderados dados que intuitivamente dizem mais respeito a função do jogador na partida do que ao tipo de personagem utilizado. foram removidos dessa etapa 11 (onze) características, restando 17 (dezessete) para análise.

Na primeira abordagem para a utilização dos dados das partidas foi utilizado um objeto por personagem, 139 (cento e trinta e nov) no total, com os valores médios das estatísticas daquele personagem nas partidas coletadas. Na Tabela 5.4 estão os resultados da avaliação externa desta abordagem. O agrupamento *fuzzy*, *k-means* e o hierárquico *ward* tiveram os melhores resultados. O *k-means* criou grupos com tamanhos mais similares. A utilização das médias faz com que os experimentos se tornem muito menos custosos, mas muitas características dos dados podem ser perdidas.

Inspirado na etapa do agrupamento *fuzzy* de associar os objetos ao grupo com maior grau de pertinência, após o agrupamento com toda a base de partidas, uma avaliação externa foi realizada com a associação dos personagens ao grupo com mais registros deles. Os resultados estão na tabela 5.5. Essa estratégia porém também tem perda de informações para análise.

Por último, a avaliação externa foi feita diretamente no agrupamento dos dados das partidas, ou seja na base completa. O valor de comparação para cada registro era a classe e subclasse do campeão utilizado. Os resultados dessa abordagem podem ser vistos na tabela 5.6. Foi constatado que valores maiores para o parâmetro m do algoritmo *fuzzy* favoreciam distribuições mais desiguais entre o tamanho dos grupos. O *Fuzzy C-means*

	Classes				Subclasses			
Método	R	J	FM	Γ	R	J	FM	Γ
K-means	0.808	0.330	0.498	0.382	0.847	0.174	0.300	0.217
Fuzzy 1.2	0.822	0.345	0.514	0.406	0.856	0.160	0.278	0.198
Single	0.185	0.173	0.414	0.022	0.106	0.092	0.302	0.019
Complete	0.399	0.186	0.395	0.095	0.634	0.104	0.235	0.071
Average	0.185	0.173	0.414	0.022	0.130	0.091	0.294	0.000
Ward.D2	0.715	0.240	0.401	0.230	0.832	0.131	0.235	0.143

Tabela 5.5: Avaliação externa, grupo com mais objetos do personagem

	Classes				Subclasses			
Método	R	J	FM	Γ	R	J	FM	Γ
K-means	0.776	0.264	0.419	0.281	0.842	0.148	0.258	0.171
Fuzzy 1.2/1.4	0.795	0.268	0.422	0.298	0.858	0.150	0.260	0.182
Single	0.193	0.178	0.420	0.019	0.121	0.098	0.310	0.013
Complete	0.434	0.180	0.370	0.061	0.647	0.103	0.224	0.056
Average	0.196	0.178	0.418	0.011	0.142	0.098	0.305	0.005
Ward.D2	0.728	0.224	0.371	0.203	0.829	0.134	0.238	0.142

Tabela 5.6: Avaliação externa, agrupamento completo dos jogadores em partidas

obteve melhores resultados em semelhança com as classes e subclasses. Os resultados, de modo geral, foram muito baixos, eles mostram que com um mesmo personagem os jogadores agem de diversas maneiras. A presença de dados de jogadores vitoriosos e derrotados já forma uma grande diferença entre os dados e isso pode ter tido uma grande influência nos resultados.

5.2.1.1 Avaliação relativa: O Gráfico 5.4 lista os resultados da avaliação relativa com o índice silhueta para os agrupamentos com dados das partidas. Os valores são consideravelmente baixos. O agrupamento hierárquico com *single* e *average link* obtiveram valores altos, principalmente com poucos grupos, devido a presença de apenas um grupo com quase todos os objetos.

5.2.2 Funções dos jogadores na partida:

Como explicado no Capítulo 2, cada membro do time costuma desempenhar um papel diferente durante uma partida. As responsabilidades e objetivos de cada uma dessas funções influenciam completamente o comportamento dos jogadores. Nesta seção são explicados experimentos e análises dos dados das partidas em relação às funções escolhidas.

Para realizar a análise, as 5 (cinco) funções tradicionais foram estimadas manualmente levando em consideração 2 (dois) campos disponíveis na API do League of Legends sobre

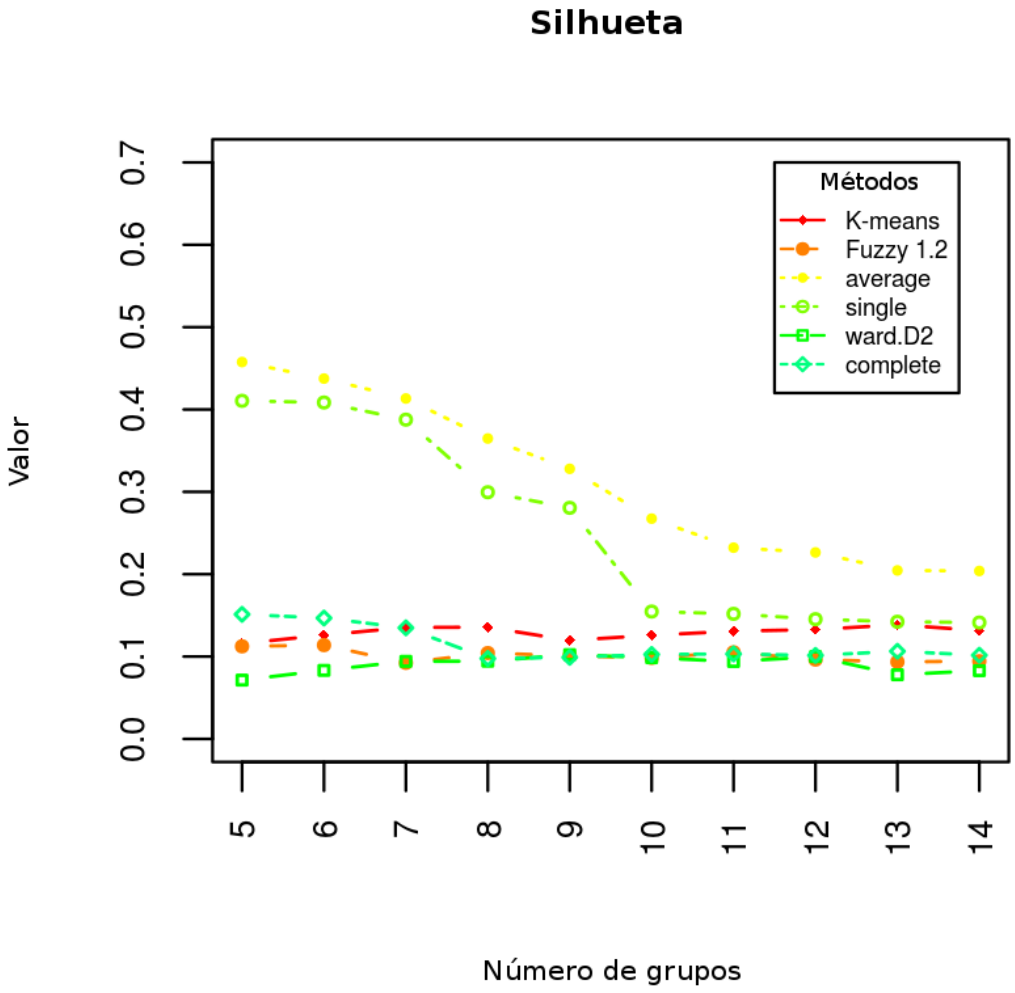


Figura 5.4: Resultado do índice silhueta para o agrupamento de dados de partidas (classes e subclasses)

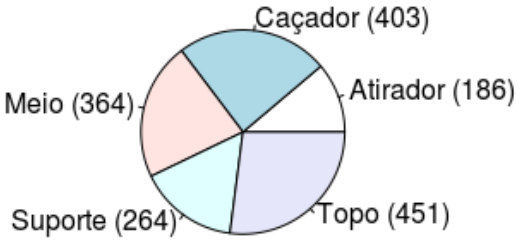


Figura 5.5: Distribuição das funções na base

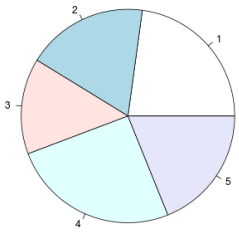
Método	R	J	FM	Γ
K-means	0.797	0.351	0.520	0.392
Fuzzy 1.4	0.802	0.361	0.531	0.406
Single	0.225	0.215	0.460	-0.007
Complete	0.292	0.212	0.439	0.015
Average	0.233	0.214	0.456	-0.008
Ward	0.793	0.397	0.572	0.438

Tabela 5.7: Avaliação externa, função dos jogadores nas partidas

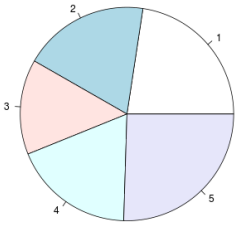
esta informação e o personagem utilizado. O Gráfico 5.5 apresenta a distribuição da base em relação às funções estimadas.

Nesta série de experimentos, apenas 3 (três) das 28 (vinte e oito) características foram desconsideradas e os personagens especialistas foram incluídos na avaliação externa. Na Tabela 5.7, estão os resultados da avaliação externa. Os piores resultados para o índice Rand (R) são dos agrupamentos com tamanhos desiguais com poucos grupos com quase todos os objetos, havendo, por isso, uma grande redução dos verdadeiros negativos. Os gráficos da Figura 5.6 fornecem uma visualização dos tamanhos dos agrupamentos para os diferentes métodos.

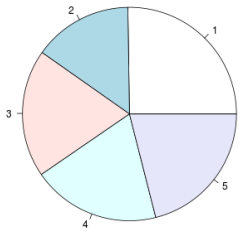
Existe uma padronização por parte da comunidade sobre quais personagens se encaixam melhor em cada função. Muitas vezes, um mesmo personagem é considerado bom para mais de uma delas, como, por exemplo, para suportes e para caçadores. E em vários desses casos, a forma "correta" de se usar o personagem muda drasticamente. A Figura 5.7 mostra exemplos de histogramas da quantidade de personagens em relação ao número de grupos que possuem, pelo menos, um registro daquele personagem.



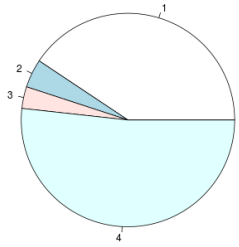
(a) K-means



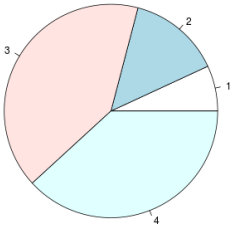
(b) Fuzzy C-Means - m = 1.2



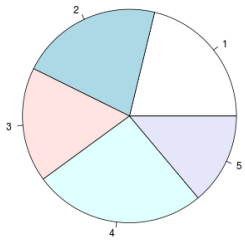
(c) Fuzzy C-Means - m = 1.4



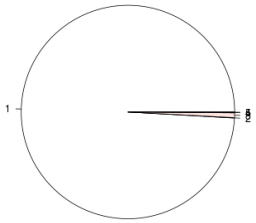
(d) Fuzzy C-Means - m = 1.8



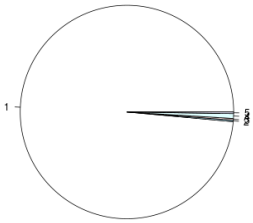
(e) Fuzzy C-Means - m = 2



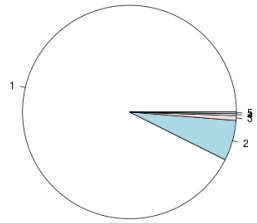
(f) Hierárquico - ward.D2



(g) Hierárquico - single link

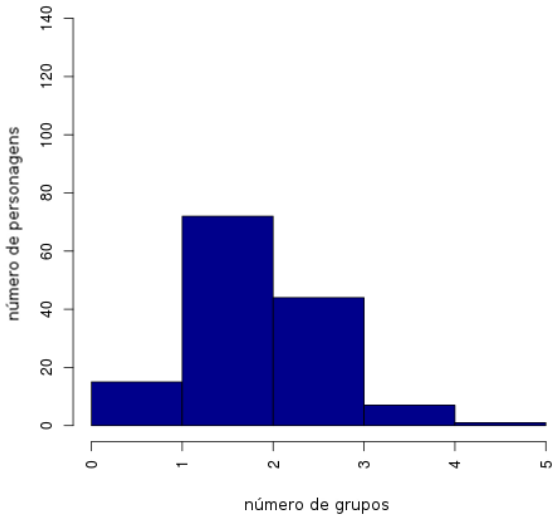


(h) Hierárquico - average link

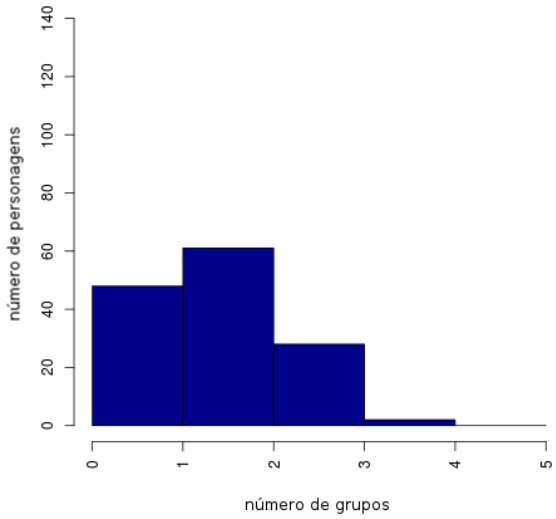


(i) Hierárquico - complete link

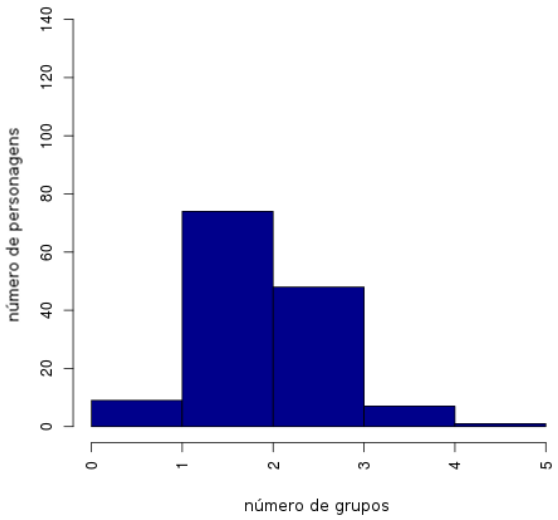
Figura 5.6: Tamanhos dos grupos, em agrupamentos de tamanho 5, com todos os métodos



(a) K-means



(b) Hierárquico - ward



(c) Fuzzy C-means - $m = 1.2$

Figura 5.7: Histograma do número de grupos por personagem. Agrupamentos de tamanho 5

Método	Todos	Vitórias
K-means	0.133	0.177
Fuzzy 1.2	0.133	0.123
Single	**0.418**	0.369
Complete	0.205	0.189
Average	0.353	**0.381**
Ward	0.143	0.164

Tabela 5.8: Resultados da avaliação relativa - índice silhueta

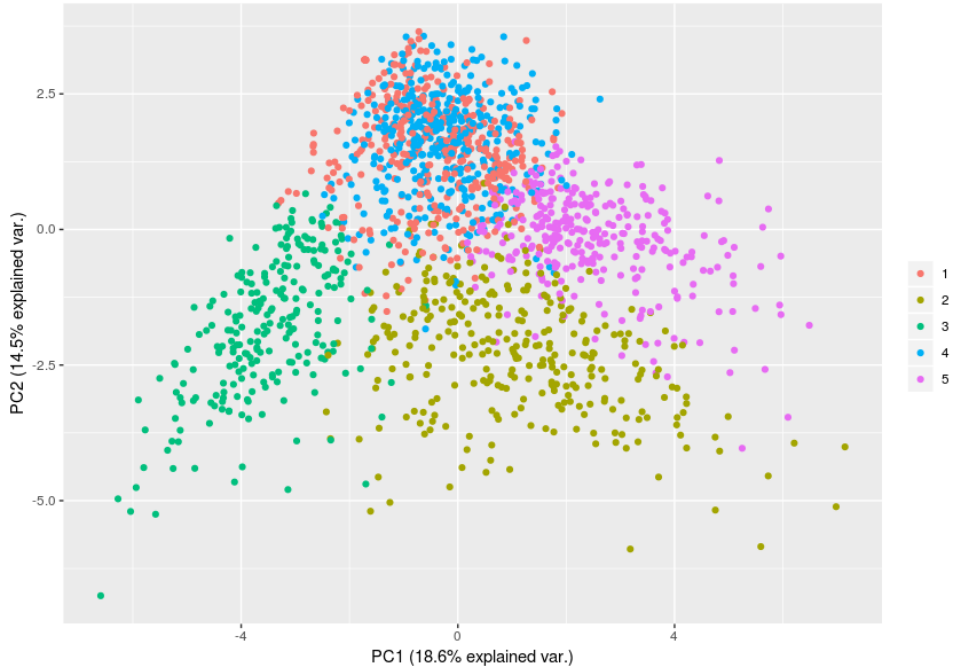


Figura 5.8: Visualização de agrupamento dos dados das partidas com PCA

O agrupamento hierárquico com o método *K-means* teve um dos melhores resultados em todos os aspectos discutidos, avaliação externa, silhueta e distribuição mais homogênea do tamanho dos grupos. A tabela 5.9 descreve os grupos de uma execução deste método (a partir dos dados médios dos grupos e não das características das funções). Já na figura 5.8 é possível ver a distribuição dos dados com os dois principais componentes do PCA. Esses dois componentes explicam somente cerca de 33% da variação dos dados. Observa-se uma grande sobreposição entre os grupos 1 (um) e 4 (quatro).

As características do agrupamento e as frequências das funções nos grupos mostram que as funções de caçador e de suporte são mas mais distinguíveis. Caçador, devido ao alto dano a monstros da própria selva. Para os suportes o mais distinguível foi a grande quantidade de assistências ao invés de abates, e a grande contribuição para a visão do mapa. As 3 (três) funções de rota não foram tão bem separadas pelo fato delas serem

Tamanho	Características	Funções frequentes	Personagens
381	Jogadores com várias estatísticas baixas como: assistências, dano a objetivos e torres. Contribuem menos com a visão e são abatidos mais vezes. Exercem mais dano físico que mágico, mas a diferença é equilibrada.	Topo (167), Meio (67), Inferior (65)	Garen, Kled, Renekton
307	Jogadores com estatísticas elevadas em vários aspectos. Maior foco no combate aos monstros da selva. A grande quantidade de dano recebido demonstra que ao contrario das rotas os monstros da selva estão sempre contra atacando o jogador. Alta contribuição à visão do mapa.	Caçador (301)	Zac, Master Yi, Rek'Sai
242	Exercem pouco dano em geral. Altíssima quantidade de assistências. Foco em melhorar as condições dos companheiros (utilidade) e em atrapalhar os adversários (controle). Normalmente são os que mais contribuem para a visão do mapa.	Suporte (219), Caçador (20)	
423	Foco em dano mágico. Muitos abates. Matam muitas tropas das rota que da selva. Recebem também mais dano mágico (já que o adversário da função direto na rota tende a ser do mesmo grupo).	Meio (272), Topo (119)	Azir, Heimerdinger, Malzahar
315	Foco em dano físico. Maior número de abates. Matam muitas tropas de rota. Maior quantidade de recursos coletados (moeda do jogo).	Topo (160), Inferior (115)	Draven, Gangplank, Illaoi

Tabela 5.9: Descrição de agrupamento dos dados das partidas

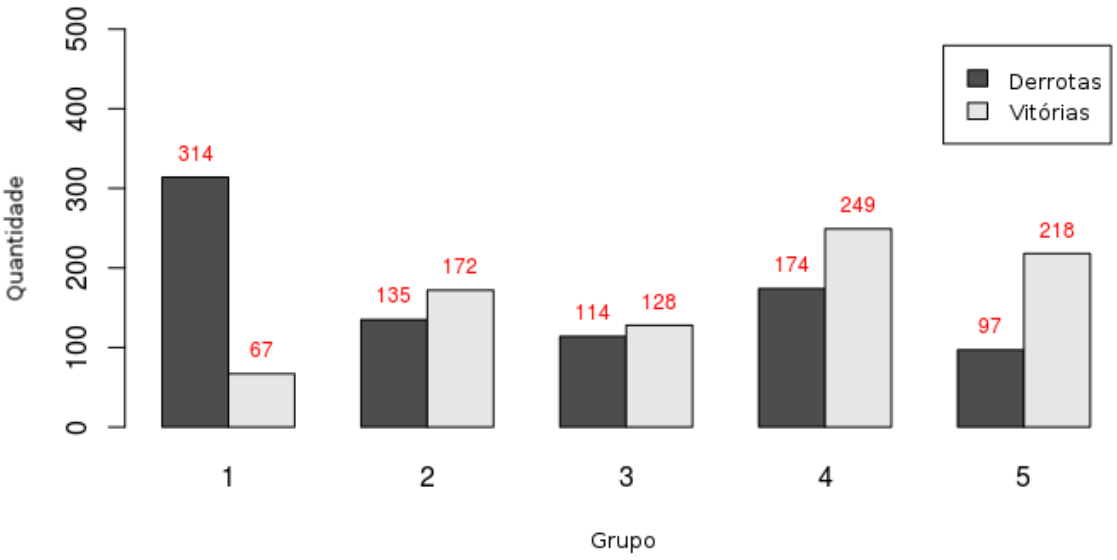


Figura 5.9: Distribuição de registros de vitória e derrota por grupo

muito mais parecidas em relação aos dados disponíveis. Pode se considerar que a maior diferença entre elas seja o posicionamento, que não é um dado disponível. Apesar de uma maior colisão entre esses conjuntos ainda foi possível identificar um grupo com a presença maior de cada uma dessas funções. Isso foi possível pelas diferenças como: a utilização de personagens de dano mágico mais na rota do meio e de dano físico na rota inferior, um maior isolamento da rota superior, percebido pela menor quantidade de assistências e dano a objetivos. Além da utilização muitas vezes de personagens mais focados em resistência, na rota superior.

O Grupo 1 (um) teve uma formação curiosa, apesar da maior quantidade de registros da função da rota superior, houve uma boa quantidade de todas as funções, mas com uma concentração de registros com desempenho inferior (jogadores que perderam). O gráfico 5.9 mostra, por grupo, a distribuição entre registros de ganhadores e de perdedores das partidas.

Como já comentado anteriormente, a presença de registros vitoriosos e perdedores já exerce uma grande influência sobre a distribuição dos dados. Foram feitos, então, testes apenas com a metade da base que venceu a partida jogada. Na Tabela 5.10 estão os resultados da avaliação externa em comparação às funções. Os resultados aumentaram consideravelmente em relação à utilização de toda a base. A Figura 5.10 mostra uma visualização dos grupos deste experimento com o método Fuzzy C-means que produziu grupos com maior semelhança em tamanho.

Método	R	J	FM	Γ
K-means	0.833	0.475	0.647	0.541
Fuzzy 1.4	0.846	0.461	0.631	0.535
Single	0.222	0.208	0.451	-0.013
Complete	0.307	0.209	0.434	0.033
Average	0.221	0.208	0.452	-0.012
Ward	0.816	0.440	0.615	0.498

Tabela 5.10: Avaliação externa, função dos jogadores nas partidas (apenas vitórias)

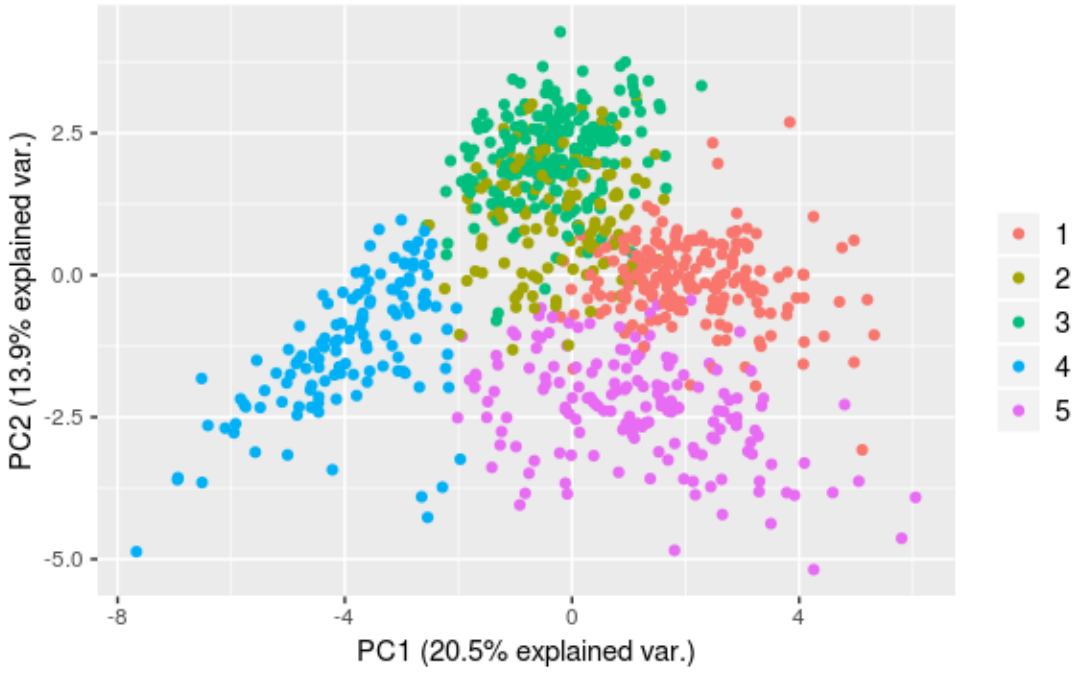


Figura 5.10: Visualização de agrupamento dos dados das partidas com PCA (apenas vitórias)

Capítulo 6

PREDIÇÃO DE VITÓRIA

Neste capítulo, são discutidos os experimentos, para a tarefa de predição do time vencedor de partidas de League of Legends. Deseja-se analisar a viabilidade de predição do resultado de uma partida antes dela começar. Antes do início de uma partida, existe a fase de *draft* quando, após a formação automática dos times, os jogadores devem escolher os personagens que gostariam de utilizar ou banir para que os adversários não possam escolhê-los. Os detalhes da partida definidos durante o *draft* são muito importantes, a ponto de especular-se o resultado apenas com esses dados. Isso acontece porque, os personagens possuem características muito diversificadas, e podem possuir vantagens sobre outros ou podem combinar bem em um mesmo time.

Neste trabalho, porém, supõe-se que a lista de personagens escolhidos deixa de fora algo essencial, que é a habilidade dos jogadores. Dessa forma, foram coletados dados que possam representar a habilidade dos jogadores dos dois times. League of Legends escolhe os jogadores de uma partida, de forma que, as duas equipes fiquem balanceadas, e assim, tenham chances parecidas de vitória. Apesar do balanceamento, ainda existem diferenças, que ficam mais evidentes após as escolhas das funções e dos personagens. Por exemplo, um jogador pode ter vencido 53% de suas partidas, mas com um personagem ter vencido 61%, e com outro, apenas 45%. Esse trabalho utiliza dados gerais da experiência dos jogadores e também considerando as escolhas do *draft*, tirando proveito das combinações entre aliados e adversários, para inferir os resultados.

Estudos que envolvam a influência de experiências e escolhas do *draft* nas partidas podem contribuir bastante, para auxiliar e educar os jogadores a, por exemplo, escolher um personagem que combine melhor com o seu time e seja mais forte contra os adversários. Vale a pena explicar que, no *draft* de partidas de LoL, os jogadores não sabem quem está jogando no time adversário, apenas os personagens escolhidos.

Uma aplicação direta para a predição de resultados é o mercado de apostas, o que só existe para campeonatos. Outra aplicação é ajudar os jogadores a decidirem cancelar uma partida se as chances de vitória de sua equipe são baixas. No entanto, este é um

comportamento considerado inadequado, pois o cancelamento de partidas, mesmo que ainda no draft, atrapalham bastante a experiência dos usuários que terão que repetir aquela etapa. Por isso os jogadores são punidos quando abandonam uma partida durante o *draft*.

6.1 DADOS

Cada instância da base de dados, criada para este trabalho, é baseada em uma partida de League of Legends, de uma das mais de 10 (dez) regiões de servidores, aos quais as contas dos jogadores estão associadas. Uma instância é formada pelos dados básicos da partida (que inclui o rótulo de qual das duas equipes foi a vencedora), pelas escolhas do *draft*, pelas estatísticas dessas escolhas e os dados de experiência dos jogadores. Todas as partidas utilizadas como instância são do modo ranqueado, pois existe uma maior confiança de que os jogadores terão mais seriedade. Para compor os dados sobre a experiência do jogador, no entanto, foram utilizadas todas partidas do mapa 5 (cinco) contra 5 (cinco), mesmo as não ranqueadas.

Quase todos os dados utilizados possuem um valor por jogador. Eles foram coletados separadamente e unidos para formar um exemplo. Essa união foi feita considerando o lado dos times no mapa do League of Legends. Primeiro o time azul, que tem sua base no canto inferior esquerdo da arena, e segundo, o vermelho, do canto superior direito. Existe uma discussão dentro da comunidade do jogo sobre as vantagens desses dois lados, que envolvem posicionamento dos menus do jogo, localização de monstros em relação aos personagens mais comuns em cada rota e até movimento realizado pelo pulso do jogador com o mouse. A base coletada é bem balanceada em relação a quantidade de vitórias para cada lado. Cerca de 49% para o azul e 51% para o vermelho.

Em cada time, os dados dos jogadores foram ordenados pela função que eles, teoricamente, desempenharam no jogo. As funções foram explicadas no Capítulo 2, e assim como feito para os experimentos do Capítulo 5, os valores foram estimados a partir de dois campos disponíveis nos dados das partidas na API do LoL.

O dados que compõem cada instância da base podem ser divididos nos seguintes grupos:

- **Dados básicos do *draft***. As informações coletadas diretamente do *draft* são as escolhas dos personagens e as funções dos jogadores, que foram traduzidos na ordenação dos jogadores.

- **Estatísticas dos personagens escolhidos**. Sempre existe algum desbalanceamento entre as forças dos personagens do jogo, e até por isso os desenvolvedores estão sempre fazendo ajustando-os. Neste trabalho são utilizadas as médias de vitória dos personagens escolhidos. Essas informações foram do site de estatísticas (GRAPH, 2018). Foram coletadas, também, a taxa de vitória dos confrontos diretos em cada rota ou função, e entre o suporte e atirador.

- **Estatísticas sobre os jogadores**. Esses dados tentam representar a qualidade

dos jogadores considerando, por exemplo, o personagem escolhido e a função desempenhada. Eles foram coletados da API e alguns computados "manualmente" analisando todas as partidas jogadas até 1 semana antes da partida alvo.

No total, a base possui 5178 (cinco mil cento e setenta e oito) instâncias e, sem nenhuma seleção, foram coletadas mais de 100 características por jogador. 80% da base (4143 (quatro mil cento e quarenta e três) exemplos) foi utilizada para treinamento e 20% (1035 (um mil e trinta e cinco)) para validação dos modelos.

6.2 EXPERIMENTOS E RESULTADOS

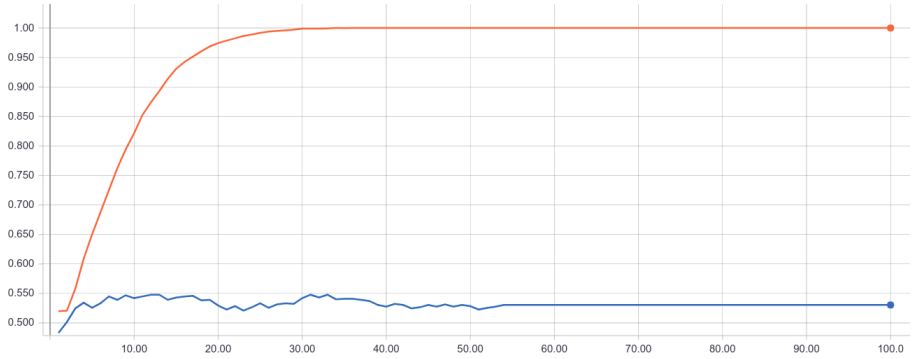
Nesta seção, são explicados e discutidos os experimentos utilizando *Random Forest* e Rede Neural MLP. Foram feitas análises do comportamento do aprendizado a partir da mudança nos parâmetros dos algoritmos e nos dados. No geral o resultados mostraram a dificuldade da tarefa e por isso uma necessidade de mais dados.

Os Gráficos 6.1, mostram a evolução da acurácia e da função de *loss* dos experimentos utilizando a base com os dados do *draft*, de estatísticas e da experiência dos jogadores. Esses são resultados de 2 (dois) modelos, o primeiro utilizando *Random Forest* com 50 (cinquenta) árvores e o outro Rede Neural com 1 (uma) camada interna com 64 (sessenta e quatro) unidades. Essas e outras configurações foram definidas a partir de argumentos teóricos e análises da influência desses elementos no treinamento e performance dos modelos, detalhadas nas próximas seções.

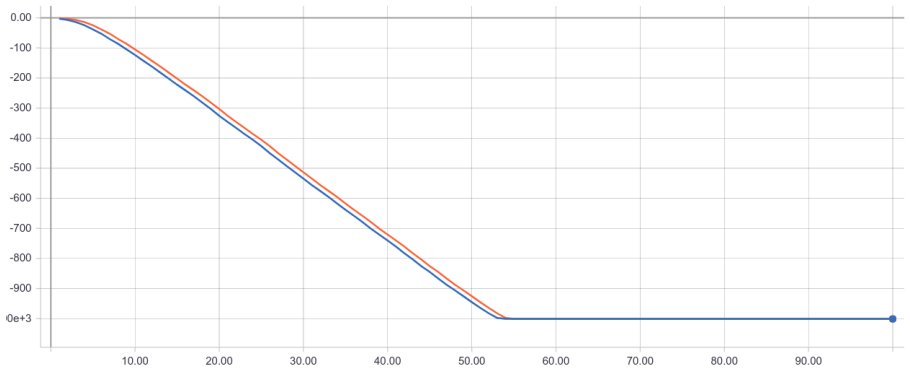
Os resultados mostram que os modelos foram capazes de aprender a partir dos dados de treinamento. No entanto, não houve uma boa generalização e aprendizado da tarefa, como indicado pela baixa evolução da função de *loss* e da acurácia para os exemplos de teste. Os melhores resultados, em termos de predição, foram de aproximadamente 56% de acurácia.

A predição do resultado de partidas antes do seu início possui uma limitação de performance, principalmente, pelo nivelamento das equipes e que, portanto, é da natureza do problema e independente do quão otimizados sejam os dados e os modelos. Avalia-se que a taxa de predição máxima real para esta tarefa, seja entre 60 e 70%. Esse intervalo é evidenciado, por exemplo, pelos resultados alcançados em outros trabalhos e pela pequena diferença entre a quantidade de vitórias e derrotas dos jogadores em partidas ranqueadas (OP.GG, 2018), devido ao nivelamento.

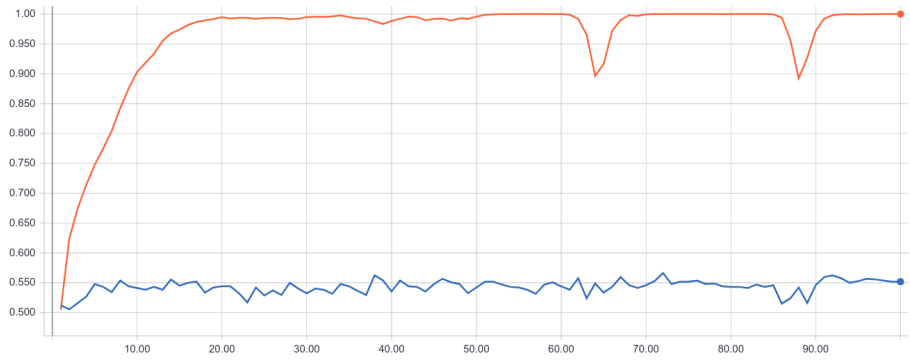
Apesar das limitações da tarefa, os resultados alcançados neste trabalho mostram que ainda existe bastante espaço para melhorias. A quantidade de exemplos coletados não foi a ideal. Para uma tarefa tão complicada, supõe-se a necessidade de muitos dados para treinar modelos capazes de obter bons resultados (BROWNLEE, 2017). A necessidade de uma base grande é ainda maior para o treinamento adequado de redes neurais. No entanto, considerando o custo para a coleta dos tipos de dados, o tamanho da base representa um avanço, se comparado aos poucos trabalhos já realizados que utilizaram esse mesmo tipo de informação do League of Legends, como (YIN, 2018) e (HALL, 2017).



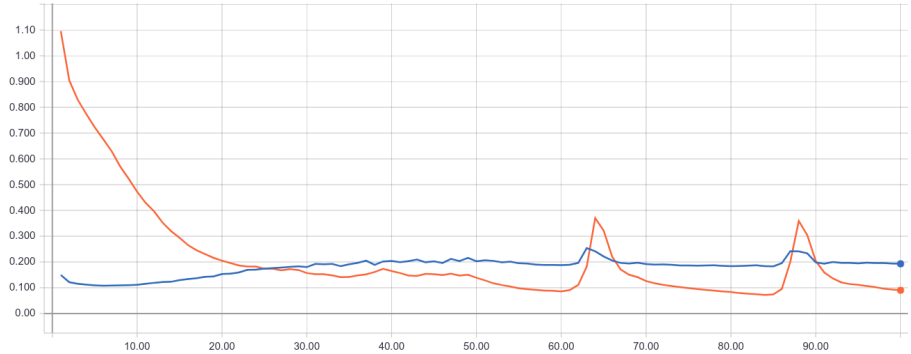
(a) Acurácia por passo - Random Forest.



(b) *Loss* por passo - Random Forest



(c) Acurácia por época - Rede Neural



(d) *Loss* por época - Rede Neural

Figura 6.1: Acurácia e *loss* para todos os métodos: Laranja - Treino; Azul - Teste.

6.2.1 Arquitetura e parâmetros da rede neural

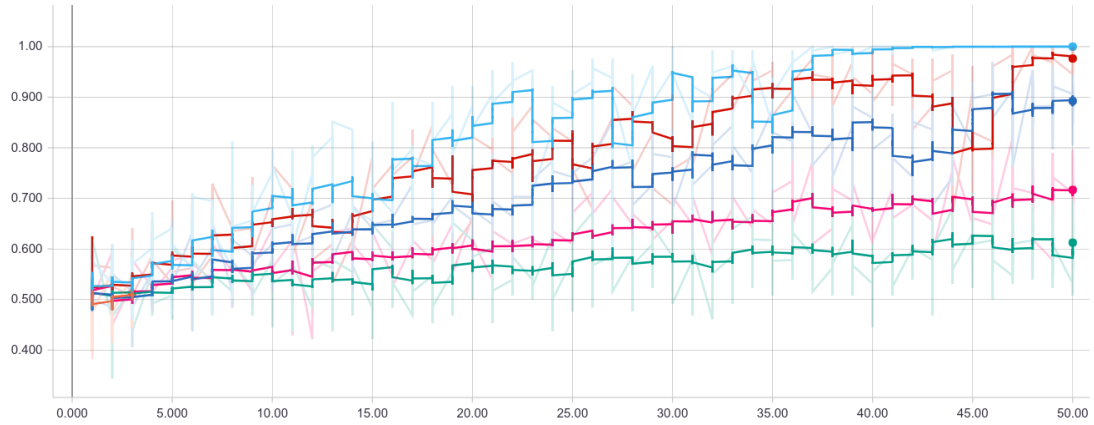
Existem diversas formas de modificar-se a arquitetura e os parâmetros de uma rede neural. Nesta seção são discutidas algumas das principais configurações e as diferenças que causam no aprendizado. Para a função de *loss* foi utilizado o *cross entropy* (DAHAL, 2018) aplicando *sigmoid* como ativação da última camada da rede, ao invés do erro quadrático médio. A função de ativação utilizada nas camadas internas foi a *ReLU*.

6.2.1.1 Largura da camada: Os Gráficos 6.2 mostram a diferença no aprendizado, em uma rede com apenas uma camada interna, com diferentes quantidades de unidades nesta camada. Os testes com menos unidades obtiveram um ajuste mais rápido (com menos épocas), em relação a base de treino. Em relação a capacidade de predizer novos exemplos, de modo geral, redes com quantidades menores (64 e 128) obtiveram melhores resultados. A quantidade ideal de unidades na camada depende, também, da quantidade de dados e do número de características.

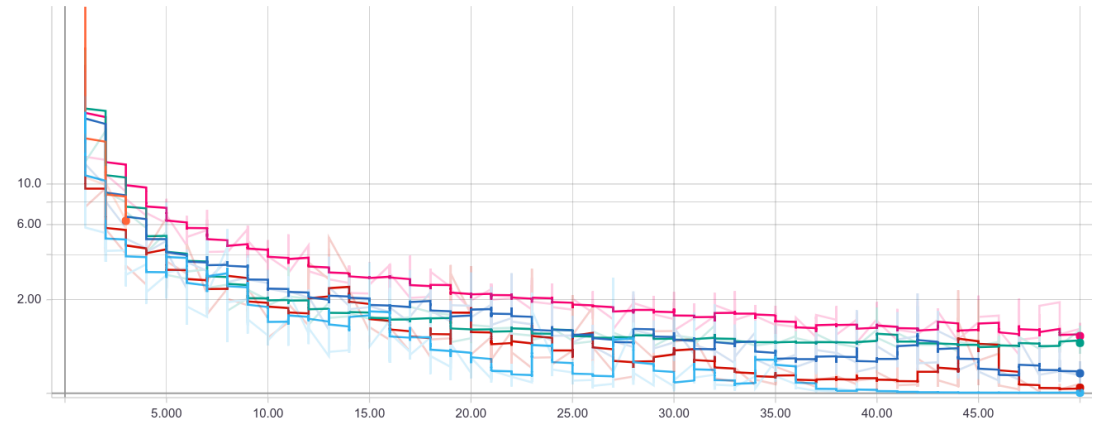
6.2.1.2 Profundidade da rede: Foi analisado, também, o comportamento do aprendizado ao aumentar a profundidade da rede neural. Essa análise é mostrada no Gráfico 6.3. Todas as camadas nesta análise tinham 64 (sessenta e quatro) unidades. O aumento na profundidade acelerou a convergência do treinamento em número de épocas. Especificamente, a rede sem camadas internas possuiu uma convergência muito mais lenta. O loss, nos testes, começa a subir em menos épocas, com a utilização de mais camadas, indicando que o aumento na capacidade da rede resultou em um sobreajuste mais rápido.

6.2.1.3 Otimização: Existem algumas técnicas de otimização quem modificam a forma com que os pesos da rede neural são atualizados através do *Gradient Descent*. Os Gráficos 6.4 demonstram uma comparação dos resultados aplicando as técnicas: *Momentum*, *RMSProp*, *AdaGrad* e *Adam*. Os resultados mostram uma superioridade dos *RMSProp* e o *Adam*. Essas duas técnicas trabalham com individualização das taxas de aprendizado para cada parâmetro (peso entre as camadas) da rede. O Adam utiliza a mesma estratégia de individualização para a aplicação de *momentum* (THUSHV, 2017).

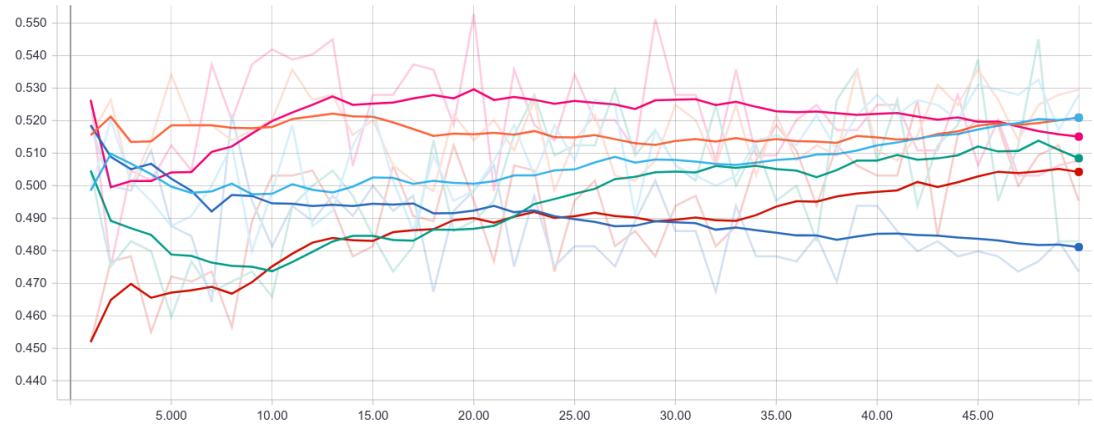
6.2.1.4 Regularização: Entre outras otimizações, o aumento da profundidade de redes neurais promove um grande aumento da sua capacidade. No entanto, também aumenta a facilidade com que o modelo realiza sobreajuste. O sobreajuste pode ocorrer ainda mais rápido devido ao tamanho da base. Para tentar diminuir esse problema com os mesmos dados, existem técnicas de regularização da rede. Nesse trabalho foram analisados os efeitos do dropout, *batch normalization* e *weight decay*. Os Gráficos 6.5 mostram a diferença nos resultados. A aplicação do batch normalization e *weight decay* geraram melhores resultados, enquanto que o *dropout* dificultou bastante o aprendizado.



(a) Acurácia por época - Treino.

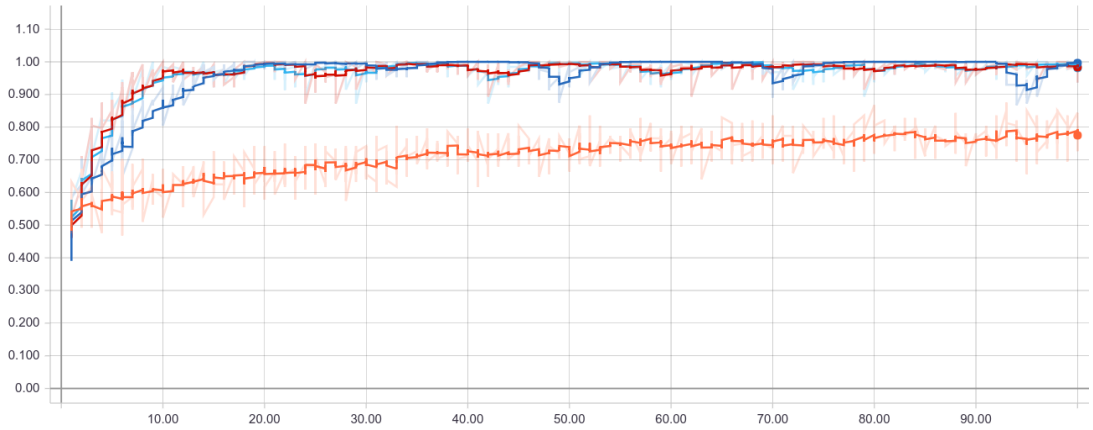


(b) *Loss* por época - Treino.

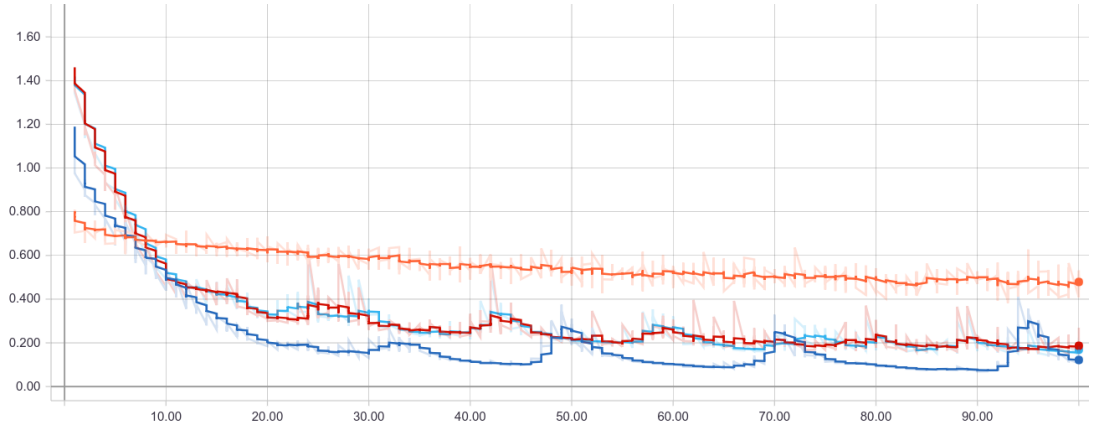


(c) Acurácia por época - Teste.

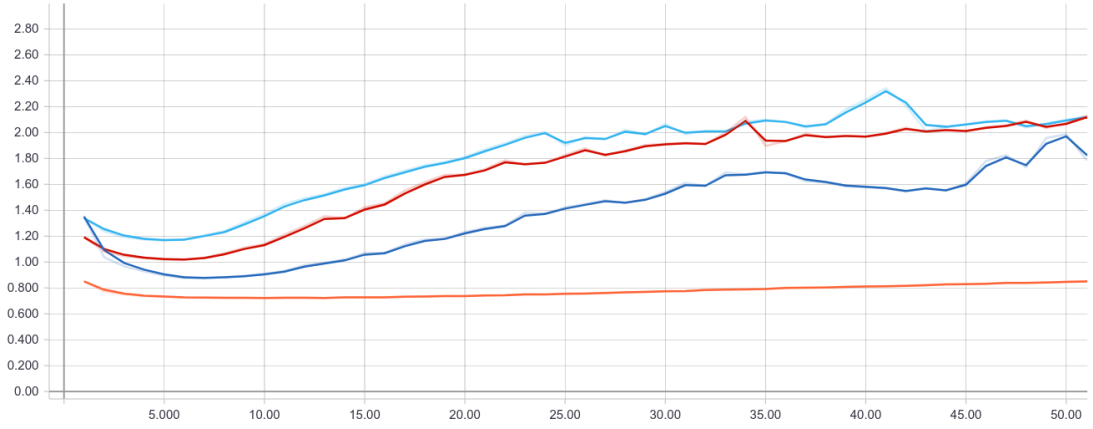
Figura 6.2: Comparação da quantidade de nós na rede com 1 camada interna: Verde - 32; Rosa - 64; Laranja - 128; Azul - 256; Vermelho - 512; Azul claro - 1024.



(a) Acurácia por época - Treino.

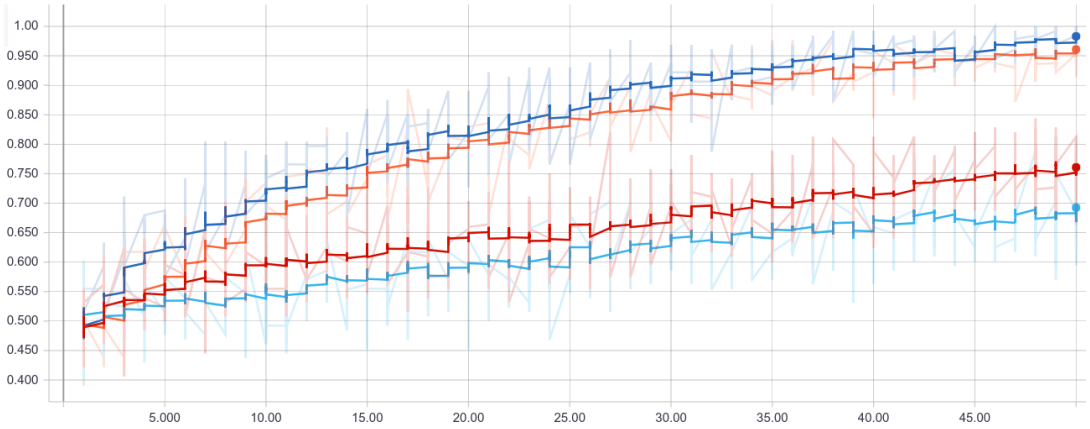


(b) *Loss* por época - Treino.

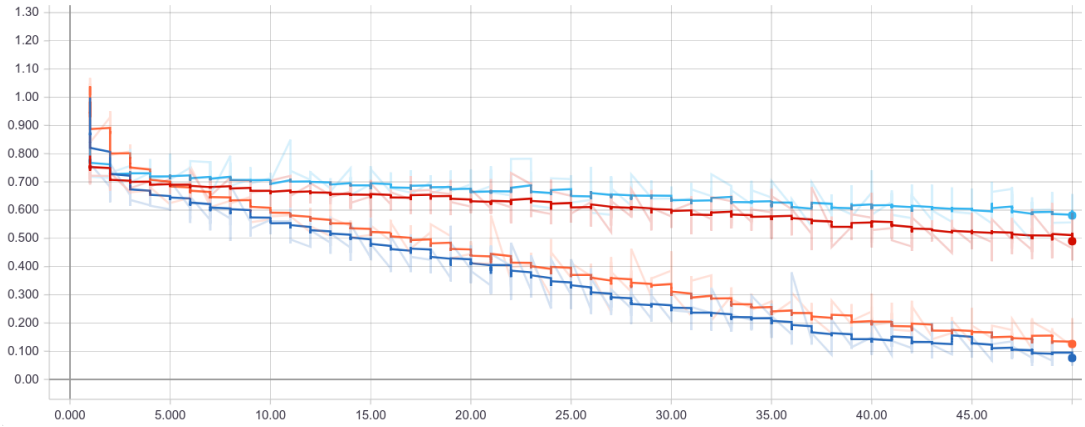


(c) *Loss* por época - Teste.

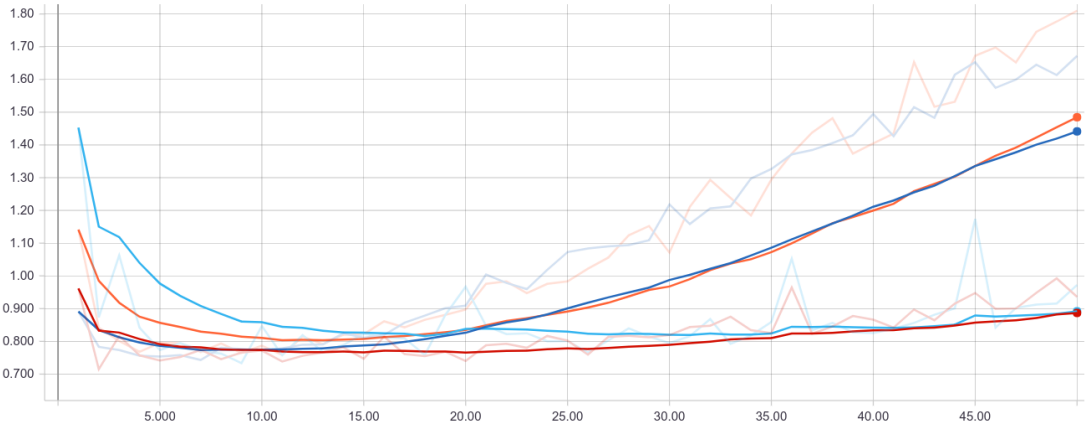
Figura 6.3: Análise da profundidade da rede neural (número de camadas internas): Laranja - 0; Azul - 1; Vermelho - 2; Azul claro - 3.



(a) Acurácia por época - Treino.

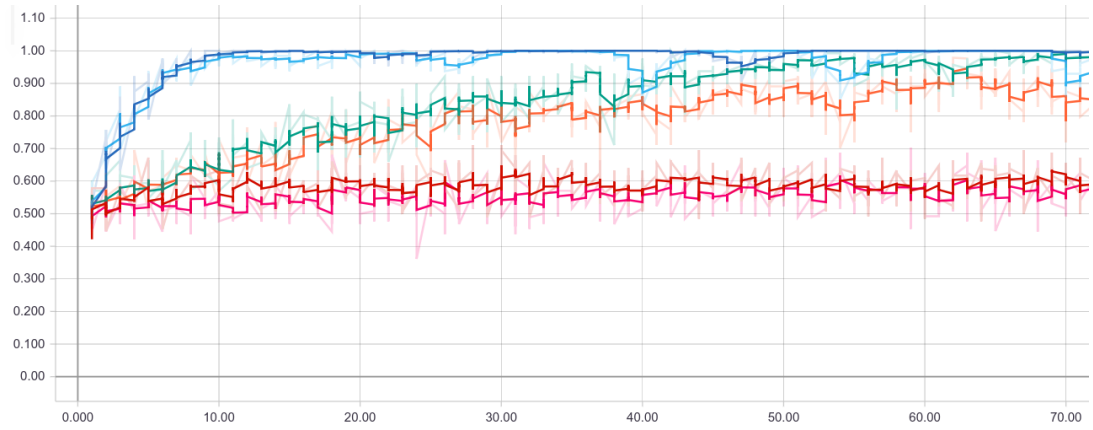


(b) *Loss* por época - Treino.

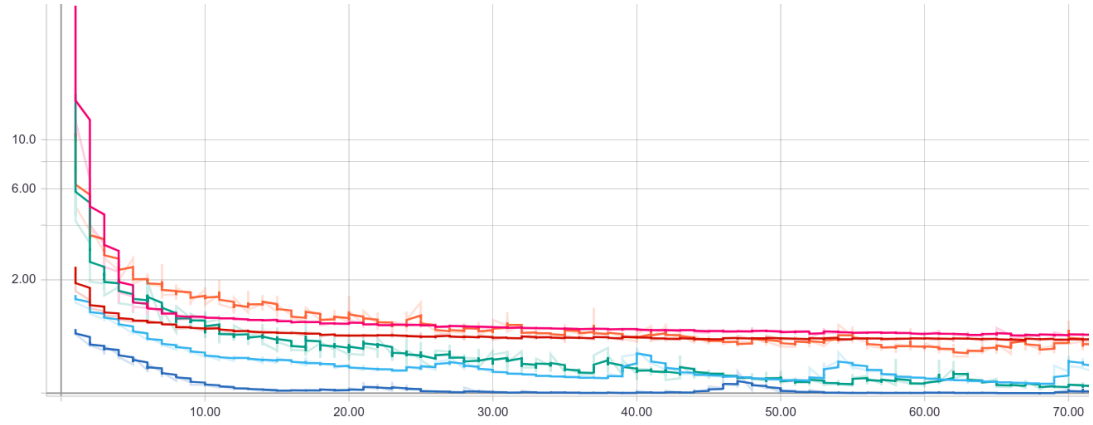


(c) *Loss* por época - Teste.

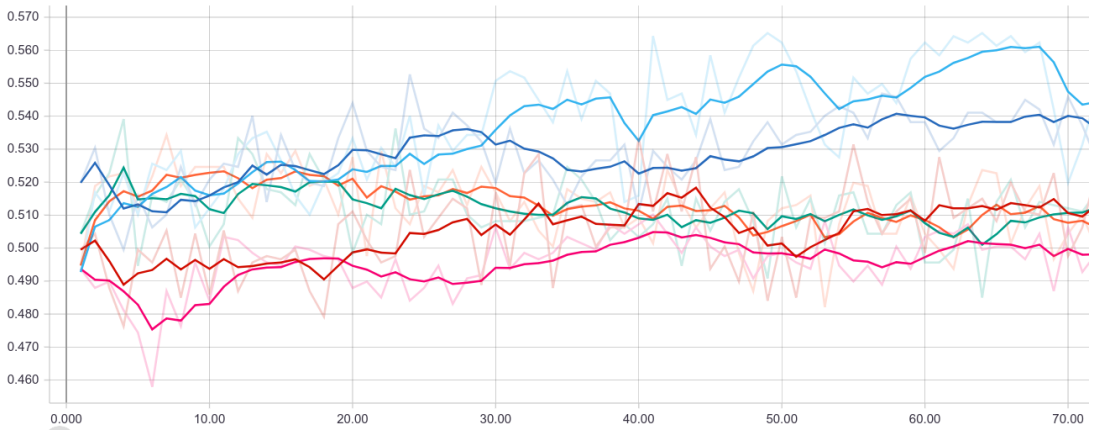
Figura 6.4: Análise das técnicas de otimização: Azul claro - *Momentum*; Vermelho - *AdaGrad*; Laranja - *RMSProp*; Azul - *Adam*.



(a) Acurácia por época - Treino.



(b) *Loss* por época - Treino.



(c) Acurácia por época - Teste.

Figura 6.5: Análise das técnicas de regularização: Verde - Sem regularização; Azul - *Batch Normalization*; Rosa - *Dropout*; Laranja - *Weight Decay*; Azul claro - *Weight Decay/Batch Normalization*; Azul claro - *Weight Decay/Batch Normalization/Dropout*.

6.2.2 Arquitetura e parâmetros da Random Forest

Para o *Random Forest* foi avaliado a quantidade de árvores treinadas no modelo. Os Gráficos 6.6 contém os resultados com os diferentes quantidades. Os resultados indicam a necessidade de mais passos do treinamento no aprendizado com poucas de árvores. Como no *Random Forest* um subconjunto dos exemplos é usado em cada árvore, uma quantidade muito pequena de árvores, em comparação com o tamanho da base, pode deixar de fora um exemplo ou usá-lo muito pouco (DEVINIAK, 2012).

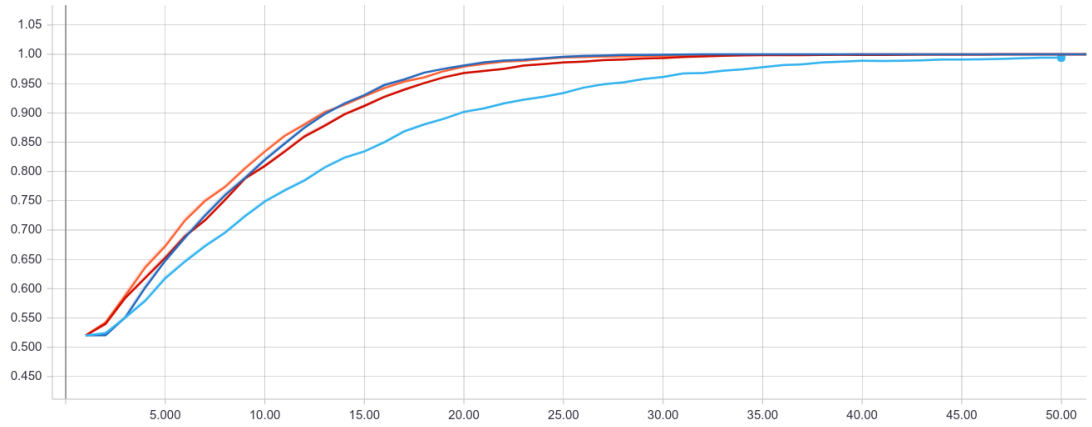
6.2.3 Manipulação e seleção de características

Um dos principais objetivos deste trabalho é ter uma melhor inferência dos resultados com a ajuda do conhecimento da habilidade dos jogadores. Por isso, foi feita uma avaliação comparando o treinamento com e sem a utilização dos dados de sobre a experiência. A quantidade de registros para treino e validação foram iguais nos dois experimentos. A diferenças utilizando *Random Forest* e Rede Neural com 1 camada interna são mostradas nos Gráficos 6.7 e 6.8. A acurácia na predição de exemplos desconhecidos é um pouco maior com a utilização dos dados das experiências dos jogadores.

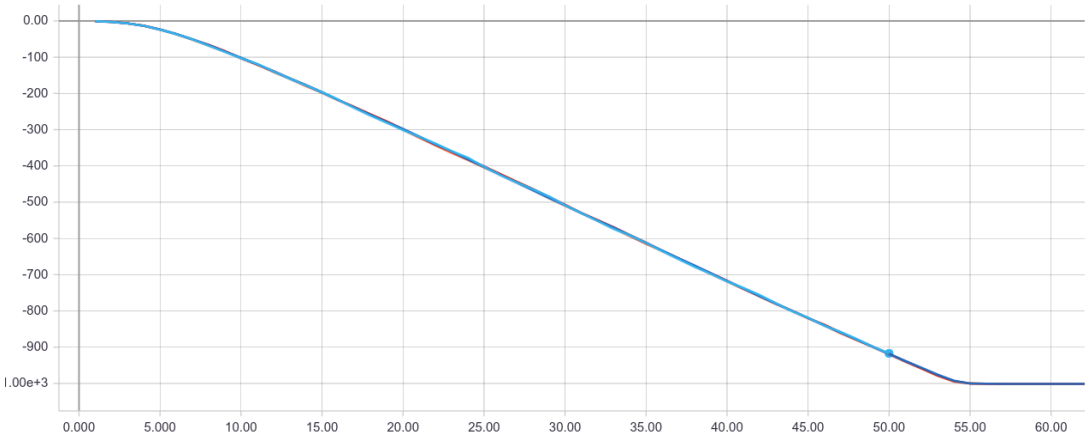
6.2.4 Validação com tarefa simplificada

Eventuais erros nos códigos desenvolvidos para os experimentos ou falhas na construção dos modelos poderiam, também, estar ocasionando um baixo desempenho no aprendizado da tarefa. Para avaliar esta possibilidade foram utilizados os mesmos códigos em uma tarefa mais simples e na qual fosse esperado um bom aprendizado para a inferência de exemplos desconhecidos. Para não fugir do escopo do trabalho, foram realizados experimentos para inferir o resultado de partidas de LoL, mas utilizando as estatísticas da própria partida. Levando em consideração que, normalmente, o time vencedor tenha estatísticas positivas, como mais abates e menos mortes, era esperado que os modelos conseguissem aprender bem a tarefa.

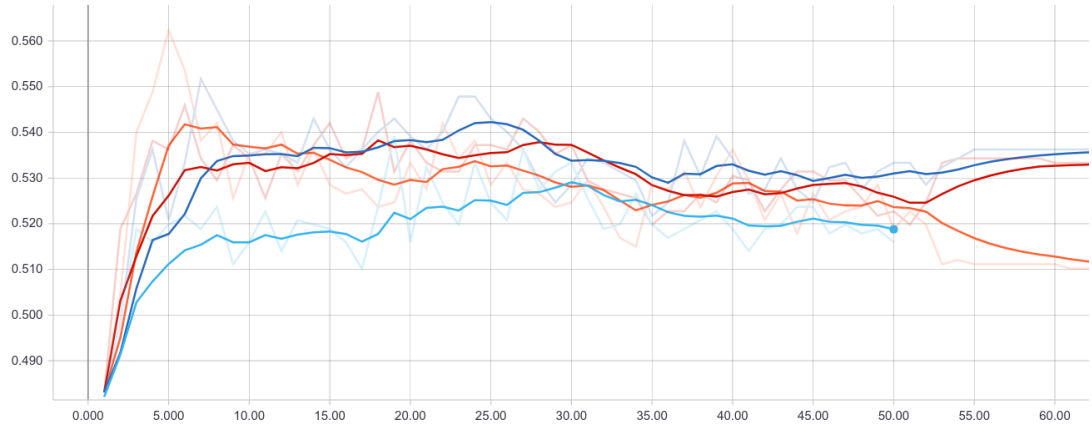
Os dados para este problema também foram coletados utilizando a API do League of Legends. Foram utilizadas 1067 partidas para treinamento e 1066 para testes. Os gráficos 6.9 mostram os resultados utilizando *Random Forest* e Redes Neural com 0, 1 e 3 camadas internas. Os modelos foram capazes de aprender bem a tarefa, alcançando até 95% de acurácia para os testes desconhecidos. O aprendizado bem sucedido para esta tarefa, utilizando os mesmos códigos que o problema original, aumentam a confiança de que não haja erros graves na manipulação dos dados, dos modelos ou na utilização do *TensorFlow*.



(a) Acurácia por passo - Treino.

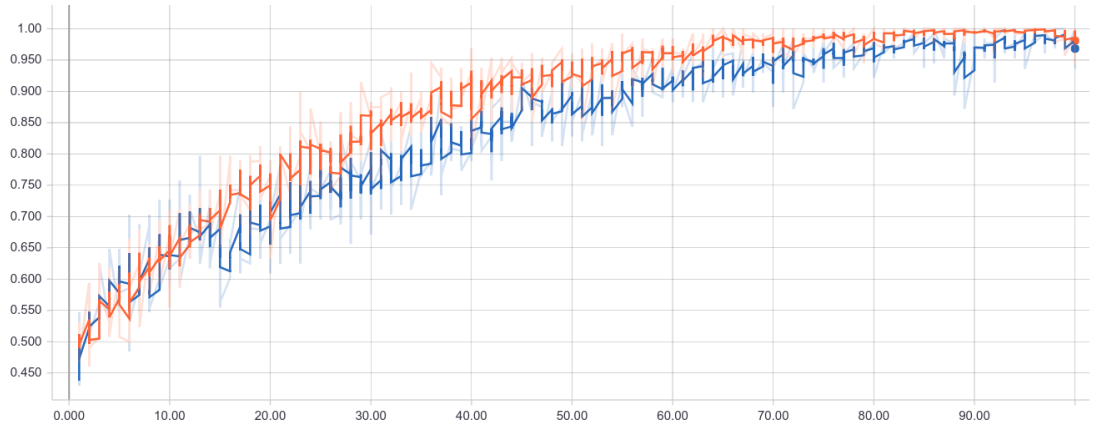


(b) *Loss* por passo - Treino.

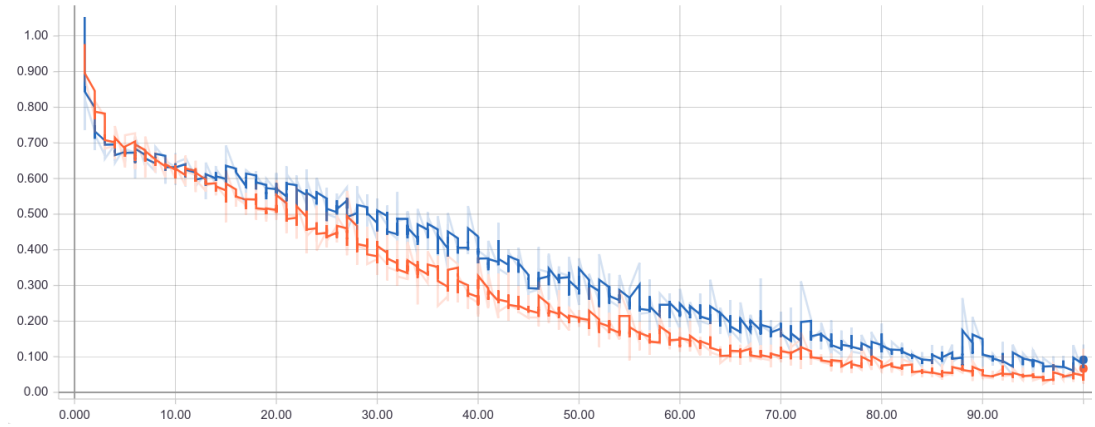


(c) Acurácia por passo - Teste.

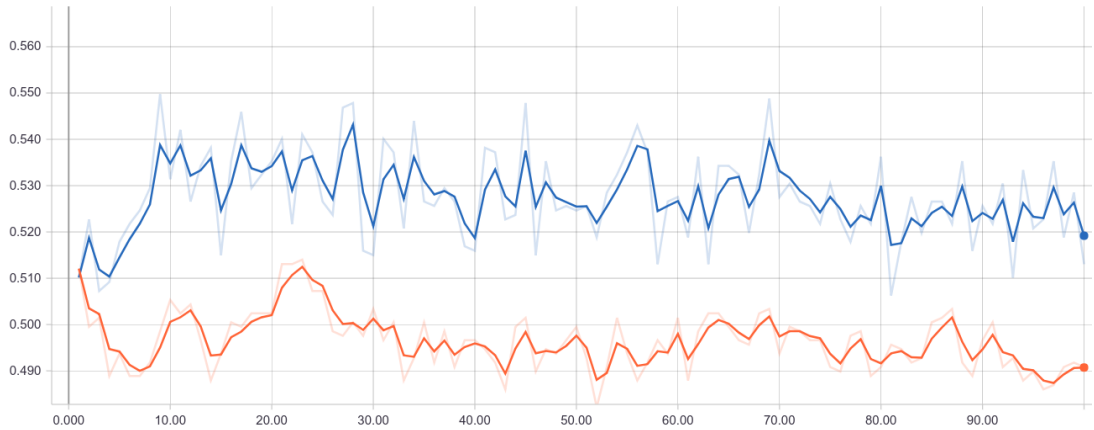
Figura 6.6: Análise da quantidade de árvores no *Random Forest*: Azul claro - 10; Vermelho - 30; Laranja - 50; Azul - 100.



(a) Acurácia por época - Treino.

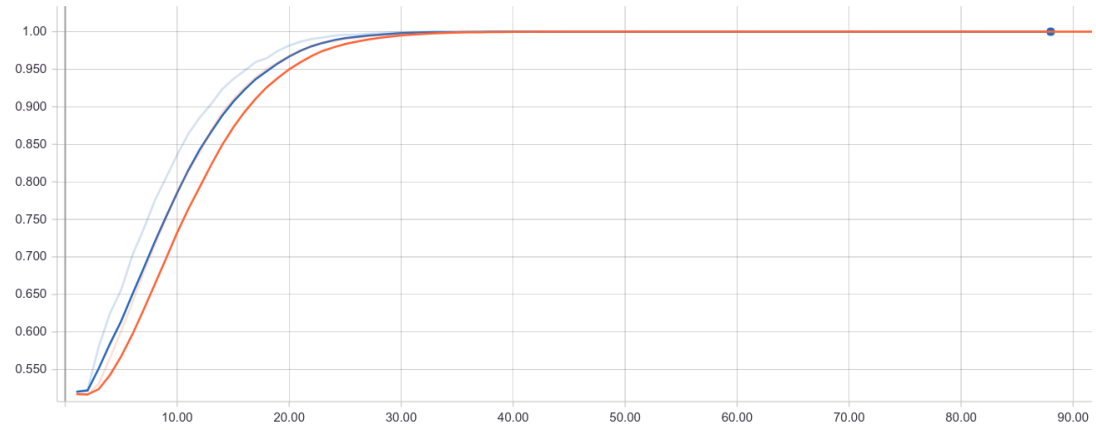


(b) *Loss* por época - Treino.

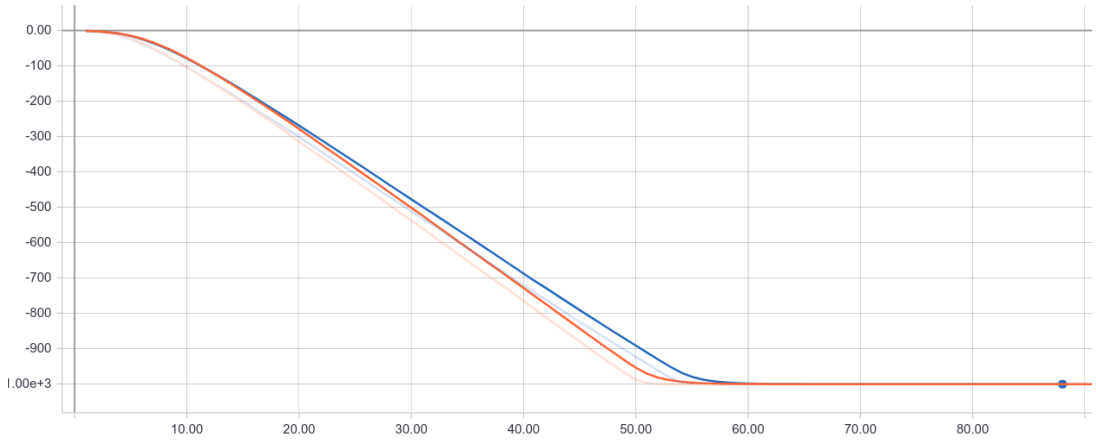


(c) Acurácia por época - Teste.

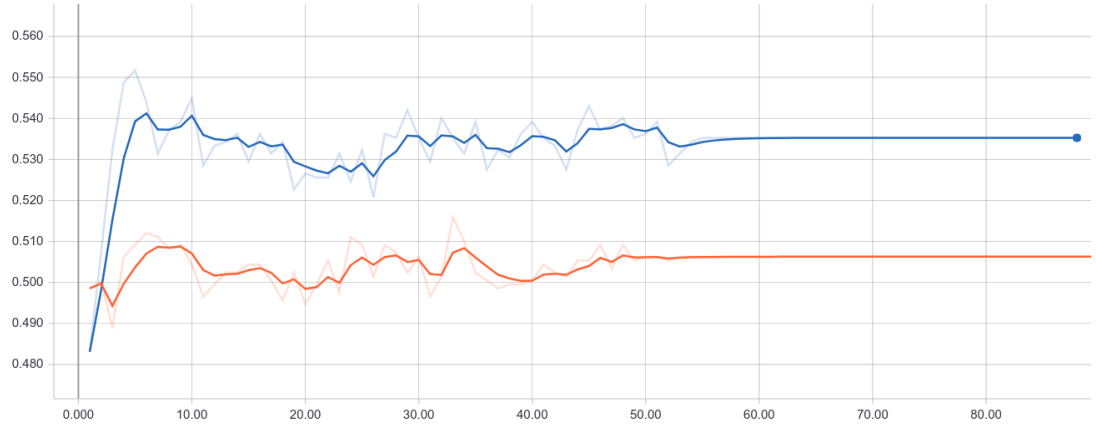
Figura 6.7: Comparação com o uso dos dados de experiência - Rede Neural: Laranja - apenas *draft*; Azul - dados de experiência.



(a) Acurácia por passo - Treino.

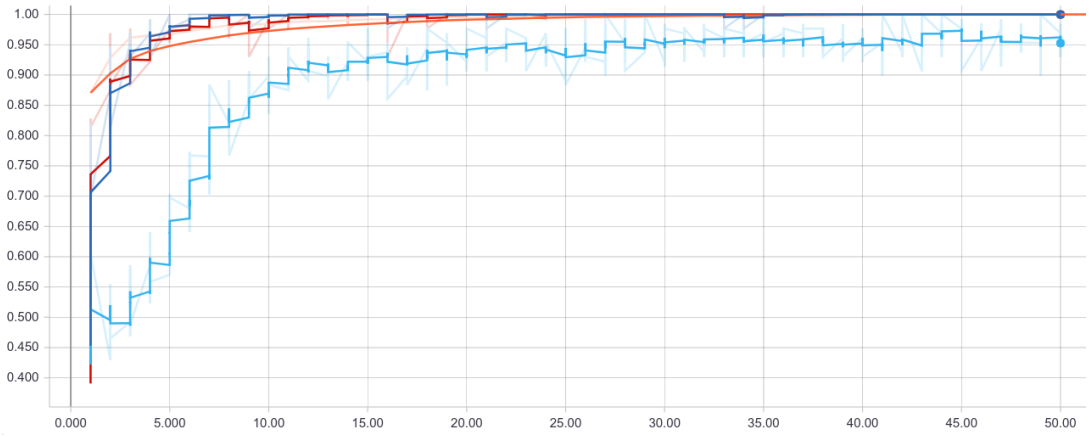


(b) *Loss* por passo - Treino.

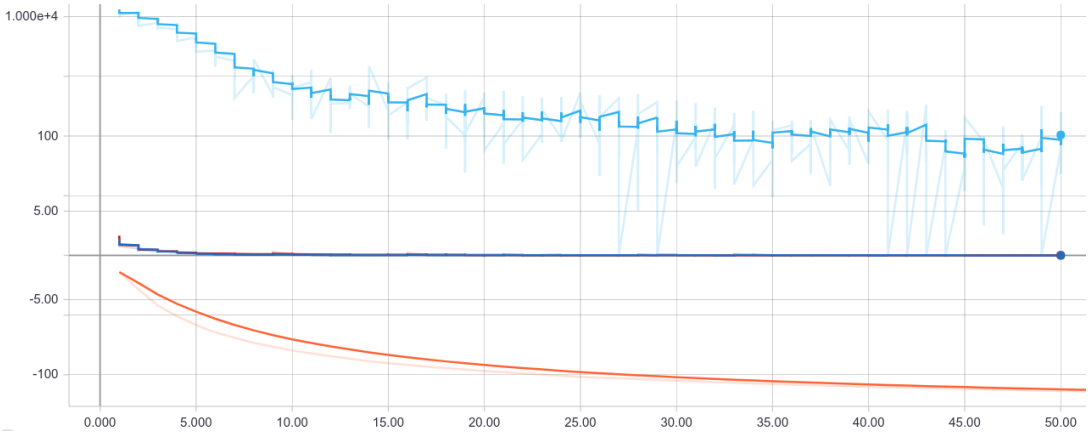


(c) Acurácia por passo - Teste.

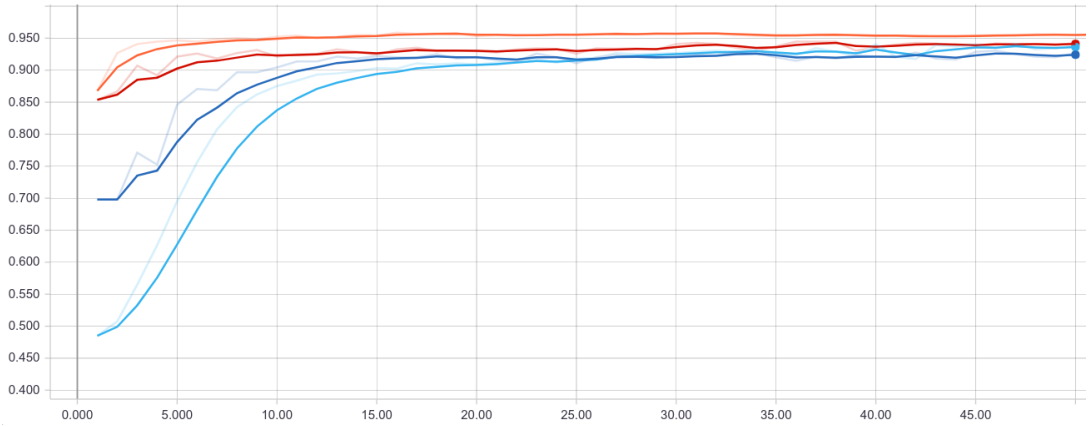
Figura 6.8: Comparação com o uso dos dados de experiência - *Random Forest*: Laranja - apenas *draft*; Azul - dados de experiência.



(a) Acurácia por época - Treino.



(b) *Loss* por época - Treino.



(c) Acurácia por época - Teste.

Figura 6.9: Predição com dados da própria partida: Laranja - *Random Forest*; Azul claro - 0 camadas internas; Vermelho - 1 camada internas; Azul - 3 camadas internas.

Capítulo 7

CONCLUSÃO

Este trabalho teve como objetivo principal a aplicação e análise de métodos de Aprendizado de Máquina em dados de esportes eletrônicos, mais especificamente do jogo League of Legends. Diversas discussões foram feitas ao longo do texto, comparando os resultados com características do jogo, dos dados, ou dos métodos. LoL é um MOBA jogado em equipe e, para vencer uma partida, é necessário muito conhecimento e estratégia de equipe. Foram analisados diversos tipos de dados do League of Legends, desde informações básicas sobre os personagens até estatísticas de partidas e jogadores do mundo inteiro. Para a realização dos experimentos, muitos dados precisaram ser coletados. Foram utilizadas ferramentas de *crawler*, para extrair informações de diversas fontes. A mais importante foi a API do League of Legends, de onde foram coletadas informações de partidas e jogadores.

A primeira parte deste trabalho foi focada na aplicação de aprendizado não supervisionado. Foram utilizadas diversas técnicas de agrupamento, e os grupos gerados foram analisados em termos de aproximação estatística dos dados e do jogo. Através da avaliação externa, foi estudado como os dados se aproximam do que é esperado pelos desenvolvedores e do comportamento real dos jogadores. As comparações foram feitas com as classes dos personagens e com a funções dentro de um time durante uma partida, que são quesitos muito importantes dentro do jogo. A aplicação de diversos métodos de agrupamento foi importante para analisar diferentes propriedades desses algoritmos e como isso influenciou os grupos gerados. As análises com os dados dos personagens mostraram que as suas características básicas não representam tão fielmente as classes e que isto se deve a uma grande sobreposição dos dados, que foi verificada pela avaliação relativa. Através da utilização de técnicas *fuzzy* e de agrupamento por densidade, foi possível validar hipóteses de que os personagens especialistas são mais difíceis de serem agrupados, o que está de acordo com a sua classificação.

Estatísticas sobre o desempenho dos jogadores durante as partidas foram utilizadas para comparar o comportamento deles jogando com os diferentes tipos de personagens e com a função que teoricamente deveriam ter durante a partida. A primeira análise foi

realizada com diversas estratégias e todas elas demonstraram que um mesmo personagem é utilizado de diversas formas. Na análise das partidas em relação às funções na equipe foi possível entender quais dos papéis, como os de caçador e suporte, são mais distinguíveis, a partir das estatísticas dos jogadores. Também foi possível, verificar que diversos personagens são utilizados em mais de uma função.

Na segunda etapa do trabalho, os esforços foram voltados à aplicação de aprendizado supervisionado na tarefa de predizer o time vencedor de uma partida de LoL. A base de dados criada foi composta por mais de 5 mil partidas. As análises envolvendo a predição do resultado das partidas mostraram que esta tarefa é muito complexa devido ao balanceamento dos times, e que a quantidade de dados foi pequena para alcançar resultados relevantes. No entanto, foi possível validar a importância da utilização da experiência dos jogadores para ajudar a inferir as chances de vitória dos times. Nesta etapa do trabalho, foram aplicados os algoritmos de Aprendizado *Random Forest* e Rede Neural. Foi estudado, ainda, os impactos no aprendizado e nos resultados dos modelos ao aplicar diferentes arquiteturas e técnicas de otimização para esses algoritmos.

Resultados enfatizaram que é preciso ainda mais estudo sobre como representar as habilidades de um jogador. Análises e modelagens do comportamento dos jogadores nas partidas, como realizadas em alguns trabalhos citados no Capítulo 3, podem auxiliar este processo. É necessário também a criação bases de dados mais robustas. Parte dos dados coletados sobre a experiência dos jogadores já eram computados pelo LoL e extraídos diretamente do perfil do jogador. A outra parte foi computada a partir do seu histórico de partidas e, por isso, foram muito custosos. Pode ser testada a criação de uma base sem esse segundo grupo, o que facilitaria a aquisição de mais exemplos.

REFERÊNCIAS BIBLIOGRÁFICAS

ABADI, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Software available from tensorflow.org. Disponível em: ⟨https://www.tensorflow.org/⟩.

AGARWALA, A.; PEARCE, M. Learning dota 2 team compositions. In: . [S.l.: s.n.], 2014.

BATSFORD, T. E. Calculating optimal jungling routes in dota2 using neural networks and genetic algorithms. In: . [S.l.: s.n.], 2015.

BREIMAN, L. Random forests. *Machine Learning*, v. 45, n. 1, p. 5–32, Oct 2001. ISSN 1573-0565. Disponível em: ⟨https://doi.org/10.1023/A:1010933404324⟩.

BROWNLEE, J. *How Much Training Data is Required for Machine Learning?* 2017. Disponível em: ⟨https://machinelearningmastery.com/much-training-data-required-machine-learning/⟩.

CONLEY, K. M.; PERRY, D. How does he saw me? a recommendation engine for picking heroes in dota 2. In: . [S.l.: s.n.], 2013.

DAHAL, P. *Classification and Loss Evaluation - Softmax and Cross Entropy Loss*. 2018. Disponível em: ⟨https://deepnotes.io/softmax-crossentropy⟩.

DEVINIAK, O. *Does the optimal number of trees in a random forest depend on the number of predictors?* 2012. Disponível em: ⟨https://stats.stackexchange.com/questions/36165/does-the-optimal-number-of-trees-in-a-random-forest-depend-on-the-number-of-pred⟩.

DOUKKALI, F. *Batch normalization in Neural Networks*. 2017. Disponível em: ⟨https://towardsdatascience.com/batch-normalization-in-neural-networks-1ac91516821c⟩.

DRACHEN, A. et al. Skill-based differences in spatio-temporal team behaviour in defence of the ancients 2 (dota 2). *Games Media Entertainment (GEM), 2014 IEEE*, p. 8, oct 2014.

EARNINGS, E. sports. *Highest Earnings By Country*. 2018. Disponível em: ⟨https://www.esportsearnings.com/countries⟩.

EGGERT, C. et al. Classification of player roles in the team-based multi-player game dota 2. *Entertainment Computing - ICEC 2015*, p. 14, dec 2015.

ESTER, M. et al. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. AAAI Press, 1996. (KDD'96), p. 226–231. Disponível em: ⟨http://dl.acm.org/citation.cfm?id=3001460.3001507⟩.

FANDOM. *League of Legends Wiki*. 2018. Disponível em: ⟨http://leagueoflegends.wikia.com/wiki/League_of_Legends_Wiki⟩.

FARZA. *DeepLeague: leveraging computer vision and deep learning on the League of Legends mini map*. 2018. Disponível em: ⟨https://medium.com/@farzatv/deepleague-leveraging-computer-vision-and-deep-learning-on-the-league-of-legends-mini-map-giving-d275

GAMES, R. *Riot Developer Portal*. 2018. Disponível em: ⟨https://developer.riotgames.com/⟩.

GAO, L. et al. Classifying dota 2 hero characters based on play style and performance. In: . [S.l.: s.n.], 2013.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. ⟨http://www.deeplearningbook.org⟩.

GRAPH, L. of. *Champions stats - League of Graphs*. 2018. Disponível em: ⟨https://www.leagueofgraphs.com/champions/stats⟩.

HALL, K. T. *Deep Learning for League of Legends Match Prediction*. 2017. Disponível em: ⟨https://minihat.github.io/LoL-Match-Prediction/⟩.

HSIEH, J. L.; SUN, C. T. Building a player strategy model by analyzing replays of real-time strategy games. In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. [S.l.: s.n.], 2008. p. 3106–3111. ISSN 2161-4393.

JOHANSSON, F.; WIKSTRöM, J. Result prediction by mining replays in dota 2. In: . [S.l.: s.n.], 2015.

KALYANARAMAN, K. To win or not to win? a prediction model to determine the outcome of a dota2 match. In: . [S.l.: s.n.], 2014.

KINKADE, N.; LIM, K. yul K. Dota 2 win prediction. In: . [S.l.: s.n.], 2015.

LIU, S.; BALLINGER, C.; LOUIS, S. Player identification from rts game replays. p. 313–318, 01 2013.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. Berkeley, Calif.: University of California Press, 1967. p. 281–297. Disponível em: ⟨https://projecteuclid.org/euclid.bsmsp/1200512992⟩.

MINOTTI, M. *The history of MOBAs: From mod to sensation*. 2014. Disponível em: ⟨https://venturebeat.com/2014/09/01/the-history-of-mobas-from-mod-to-sensation/⟩.

MITCHELL, T. *Machine Learning*. [S.l.: s.n.], 1997.

NEWZOO. *2018 Global Esports Market Report. Free version*. 2018. Disponível em: ⟨https://newzoo.com/insights/trend-reports/global-esports-market-report-2018-light/⟩.

NUANGJUMNONG, T.; MITOMO, H. Leadership development through online gaming. *19th ITS Biennial Conference: : Moving Forward with Future Technologies: Opening a Platform for All,*, p. 24, nov 2012.

OPENAI. *OpenAI Five*. 2016. Disponível em: ⟨https://openai.com/five/⟩.

OP.GG. *Melhores jogadores :: OP.GG BR*. 2018. Disponível em: ⟨http://br.op.gg/ranking/ladder/⟩.

POBIEDINA, N. et al. On successful team formation. *Business Informatics (CBI), 2013 IEEE 15th*, p. 8, jul 2013.

REEVES, B.; MALONE, T. Leadership in games and at work: Implications for the enterprise of massively multiplayer online role-playing games. jan 2007.

RIOULT, F. et al. Mining tracks of competitive video games. *AASRI Conference on Sports Engineering and Computer Science (SECS 2014)*, p. 6, jun 2014.

SAPIENZA, A.; GOYAL, P.; FERRARA, E. Deep neural networks for optimal team composition. In: . [S.l.: s.n.], 2018.

SCHUBERT, M.; DRACHEN, A.; MAHLMANN, T. Esports analytics through encounter detection. *MIT SLOAN Sports Analytics Conference*, p. 18, mar 2016.

SEMENOV, A. et al. Applications of machine learning in dota 2: Literature review and practical knowledge sharing. In: *Workshop on Machine Learning and Data Mining for Sports Analytics 2016*. [S.l.: s.n.], 2016.

SILVA, V. do N.; CHAIMOWICZ, L. *On the Development of Intelligent Agents for MOBA Games*. 2015. 142-151 p.

SONG, K.; ZHANG, T.; MA, C. Predicting the winning side of dota2. In: . [S.l.: s.n.], 2015.

STAFF, L. *Fan Contributions to the MSI 2018 Prize Pool*. 2018. Disponível em: ⟨https://www.lolesports.com/en_US/articles/fan-contributions-msi-2018-prize-pool⟩.

TASSI, P. *Riot's 'League of Legends' Reveals Astonishing 27 Million Daily Players, 67 Million Monthly*. 2014. Disponível em: ⟨https://www.forbes.com/sites/insertcoin/2014/01/27/riots-league-of-legends-reveals-astonishing-27-million-daily-players-67-million-monthly/\#6219018b6d39⟩.

TECHTUDO. *Entenda a premiação do CBLoL 2018: R$ 200 mil em disputa*. 2018. Disponível em: ⟨https://www.techtudo.com.br/noticias/2018/04/entenda-a-premiacao-do-cblol-2018-r-200-mil-em-disputa-esports.ghtml⟩.

TENSORFLOW. *Embeddings*. 2018. Disponível em: ⟨https://www.tensorflow.org/guide/embedding⟩.

THUSHV. *A Practical Guide to Understanding Stochastic Gradient Descent Methods: Workhorse of Machine Learning*. 2017. Disponível em: ⟨http://www.thushv.com/deep-learning/a-practical-guide-to-understanding-stochastic-optimization-methods-workhorse-of-machine-learning/⟩.

VOLK, P. *League of Legends now boasts over 100 million monthly active players worldwide*. 2016. Disponível em: ⟨https://www.riftherald.com/2016/9/13/12865314/monthly-lol-players-2016-active-worldwide⟩.

XU, R.; WUNSCHANDERS, D. *Clustering*. [S.l.]: Wiley, 2008.

YANG, P.; HARRISON, B.; ROBERTS, D. L. Identifying patterns in combat that are predictive of success in moba games. *Proceedings of Foundations of Digital Games, (Miami, Florida)*, p. 8, jan 2014.

YIN, J. *League of Legends: Predicting Wins In Champion Select With Machine Learning*. 2018. Disponível em: ⟨https://hackernoon.com/league-of-legends-predicting-wins-in-champion-select-with-machine-learning-6496523a7ea7⟩.