UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Lucas Augusto Ferreira Hanke

Sistema de Recomendação de Heróis para Jogos MOBA Utilizando Aprendizado de Máquina

Belo Horizonte
2017

Lucas Augusto Ferreira Hanke

Sistema de Recomendação de Heróis para Jogos MOBA Utilizando Aprendizado de Máquina

Versão Final

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Mestre em Ciência da Computação.

Orientador: Luiz Chaimowicz

Belo Horizonte
2017

Hanke, Lucas Augusto Ferreira.

H241s Sistema de recomendação de heróis para jogos MOBA utilizando aprendizado de máquina [manuscrito] / Lucas Augusto Ferreira Hanke. – 2017.
xxii, 52 f. il.

Orientador: Luiz Chamowicz.
Dissertação (mestrado) - Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação.
Referências: f.49-52
.
1. Computação – Teses. 2. Aprendizado do computador – Teses. 3. Sistemas de recomendação – Teses. 4.Jogos eletrônicos – Teses. I. Chaimowicz, Luiz. II. Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação. III.Título.

CDU 519.6*73(043)

Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa
CRB 6ª Região nº 1510

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Sistema de recomendação de heróis para jogos MOBA utilizando aprendizado de máquina

LUCAS AUGUSTO FERREIRA HANKE

Dissertação defendida e aprovada pela banca examinadora constituida pelos Senhores:

PROF. LUIZ CHAIMOWICZ - Orientador
Departamento de Ciência da Computação - UFMG

PROF. RENATO ANTÔNIO CELSO FERREIRA
Departamento de Ciência da Computação - UFMG

PROF. RODRYGO LUIS TEODORO SANTOS
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 31 de julho de 2017.

Agradecimentos

Se você está lendo este texto é porque finalmente consegui. E não foi fácil chegar até aqui. Do processo seletivo, passando pela aprovação até a conclusão do mestrado, foi um longo caminho percorrido. Nada foi fácil, tampouco tranquilo, mas *"se você está entre os nomes a seguir, você é um dos motivos"* de ter conseguido e gostaria de agradecê-la(o) que direta ou indiretamente me auxiliou no desenvolvimento e conclusão dessa etapa.

Aos meus amados pais, Tânia e Carlos, agradeço pelo amor incondicional, incentivo e meu maior exemplo de luta, trabalho e dedicação. Saibam que grande parte dessa conquista devo a vocês que sempre me ensinaram a lutar por tudo que acredito e sonho. Seus ensinamentos formam a base de tudo que sou hoje.

À minha linda irmã, Bruna, devo um grande obrigado pela cumplicidade, amizade, paciência e por ser meu maior abrigo em momentos tanto de alegria quanto de dificuldade.

Ao meu orientador, Luiz Chaimowicz, agradeço por toda a sabedoria e experiência compartilhados durante esses quase dois anos. Agradeço ainda por ter acreditado na idéia desse trabalho e pela paciência nas várias mudanças nesse período.

Lembrando que a idéia para esse trabalho surgiu em uma conversa entre eu, Eduardo e Juliano, agradeço aos meus amigos do Dotinha (e da vida) Eduardo, Juliano, Fernando, Paulo, Rodrigo e Alexandre. Agradeço tanto pelas constantes conversas, idéias e feedback sobre o trabalho, quanto pelas partidas e corujões memoráveis que já compartilhamos. *Stunou? É kill!*

Não posso deixar de agradecer a Sérgio e Alessandra Campos que tanto me apoiaram no início do mestrado, me deram as primeiras oportunidades na área de desenvolvimento e me ajudaram a descobrir minha verdadeira paixão.

Agradeço a Samir pela grande e sincera amizade. Tantos foram os momentos de angústia e dificuldades, mas seu companheirismo e aconselhamento foram essenciais para superar tudo. Além disso, agradeço pela grande ajuda com as revisões de texto.

Agradeço à amiga Ana Paula Gomes que, desde 2015, me inspira a trabalhar com ciência dos dados e a ser um desenvolvedor cada dia melhor.

A todos os meus amigos e família, agradeço pela torcida e pelos momentos de alegria e descontração que fizeram os problemas do cotidiano se tornarem insignificantes.
Footer

"*Do or do not. There's no try.*"
(Mestre Yoda)

Resumo

Jogos *Multiplayer Online Battle Arena* (MOBA) são atualmente um dos mais populares gêneros de jogos online. Na sua jogabilidade básica, dois times de múltiplos jogadores competem entre si a fim de destruir a base inimiga, controlando uma unidade poderosa denominada "herói". Cada herói tem diferentes habilidades, papéis e forças. Consequentemente, escolher uma boa combinação de heróis é fundamental para o sucesso de um determinado time em uma partida. Neste trabalho, é proposto um sistema de recomendação para seleção de heróis em uma partida de jogo MOBA. Foi desenvolvido um mecanismo baseado em regras de associação que sugere os heróis mais adequados para se compor um time, usando dados coletados de uma uma grande quantidade de partidas de Dota 2. Para avaliar a eficácia do *line-up*, foi treinada uma rede neural capaz de prever o time vencedor com uma acurácia de até 90,89%. Os resultados do sistema de recomendação foram muitos satisfatórios com até 76,4% de taxa de sucesso.

Palavras-chave: aprendizado de máquina, sistema de recomendação, jogos digitais, *e-sports*, MOBA, Dota 2

Abstract

MOBA games are currently one of the most popular online game genres. In their basic gameplay, two teams of multiple players compete against each other to destroy the enemy's base, controlling a powerful unit known as "hero". Each hero has different abilities, roles and strengths. Thus, choosing a good combination of heroes is fundamental for the success in the game. In this dissertation we propose a recommendation system for selecting heroes in a MOBA game. We develop a mechanism based on association rules that suggests the most suitable heroes for composing a team, using data collected from a large number of Dota 2 matches. For evaluating the efficacy of the line-up, we trained a neural network capable of predicting the winner team with a 90.89% accuracy. The results of the recommendation system were very satisfactory with up to 76.4% success rate.

Keywords: machine learning, recommendation system, digital games, e-sports, MOBA, Dota 2

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

Neste capítulo, serão detalhados o domínio de jogos *Multiplayer Online Battle Arena* (MOBA), a definição do problema, a motivação e as contribuições deste trabalho.

1.1 Motivação

A indústria do entretenimento conta com um grande crescimento da indústria de jogos. Esse crescimento provocou um aumento na necessidade de profissionais especializados no desenvolvimento de jogos e despertou o interesse acadêmico na área. Com o advento de computadores pessoais mais rápidos, jogos que demandavam mais recursos dos computadores se tornaram mais populares. Exemplos incluem jogos *Real Time Strategy* (RTS), jogos de esportes e jogos de simulação física.

Uma evidência desse crescimento da indústria de jogos é o surgimento de um novo tipo de esporte: os *e-sports*. Os *e-sports* são uma forma de esporte onde os aspectos principais são viabilizados através de sistemas eletrônicos [Hamari, 2015], por isso várias vezes são referidos como esportes eletrônicos. Em termos mais práticos, *e-sports* comumente se referem a jogos digitais competitivos (profissionais ou amadores). Os gêneros de jogos mais comuns associados com esportes eletrônicos são os de estratégia em tempo real (RTS), luta, tiro em primeira pessoa, e MOBA.

Os esportes eletrônicos estão em fase de expansão e, cada vez mais, atraem maior público, jogadores e investimentos [Martin, 2014]. Um exemplo disso é o *The International* [Valve Corporation, 2015a], maior campeonato profissional do jogo Dota 2, da Valve Corporation [2015b]. A edição do campeonato de 2015 acumulou um prêmio total de surpreendentes US$ 16.8 milhões, batendo o seu próprio recorde de maior premiação da história dos *e-sports*. A edição de 2014 acumulou um prêmio de US$ 11 milhões [Wingfield, 2014]. Segundo uma previsão da Fortune.com, em 2017 os fãs de *e-sports* iriam superar os da Liga Nacional de Futebol dos Estados Unidos em números [John Gaudiosi, 2015].

Dada essa nova tendência, não é de se estranhar que o interesse acadêmico em jogos

digitais aumentasse. Vários aspectos de jogos são alvos de inúmeras pesquisas atualmente, tais como: planejamento, tomada de decisões em tempo real, aprendizado e modelagem de oponentes, raciocínio espacial e temporal, gerência de recursos, colaboração entre diversos agentes e planejamento de caminhos utilizando mecanismos de Inteligência Artificial [Buro and Furtak, 2004, Yannakakis and Togelius, 2014].

Uma área que tem se destacado em diversos cenários, além de jogos, é a de sistemas de recomendação. Com a grande quantidade de dados disponibilizada na internet, a análise desses dados a fim de extrair informações relevantes e fazer recomendações para usuários se torna uma realidade em diversos domínios. Em jogos, isso pode ser especialmente valioso para jogadores iniciantes, por exemplo. A curva de aprendizagem em alguns jogos pode ser extremamente ingrata e pode até diminuir a taxa de adesão dos mesmos. Ter um sistema que já possua alguns conhecimentos do jogo e que auxilie os jogadores nas tomadas de decisão pode tornar a experiência dos mesmos muito mais prazerosa [Cunha et al., 2015].

1.2 Jogos MOBA

Recentemente, os jogos MOBA, um variante do gênero RTS, evoluíram muito e abrangeram uma fração significativa do mercado de jogos, em especial dos esportes eletrônicos. O termo MOBA pode ser usado, em sentido amplo, para qualquer jogo no qual dois times constituídos de múltiplos jogadores competem entre si em um mapa, ou arena, a fim de alcançar a vitória. Normalmente, esse objetivo é alcançado ao eliminar a base inimiga. Apesar da definição ser bastante ampla, jogos MOBA geralmente são constituídos de duas equipes com cinco jogadores cada, no qual cada jogador controla um personagem apenas (frequentemente chamado de herói). Os jogos acontecem em tempo real e, como ocorre em todos os esportes baseados em times, o *gameplay* é altamente dinâmico, tornando cada partida única [Drachen et al., 2014]. Exemplos de MOBAs altamente populares atualmente são *League of Legends* (LOL), Dota 2 e *Heroes of the Storm* (HOTS).

MOBAs são jogos complexos e apresentam dezenas ou centenas possibilidades de heróis para o jogador. Cada herói possui um conjunto individual de habilidades que podem ser evoluídas de inúmeras maneiras distintas durante a partida ao ganhar experiência (frequentemente referida como XP) ou ouro. Ouro é a moeda usada pelos jogadores a fim de comprar novos itens para seu herói e torná-lo mais forte. A fim de ganhar, cada time deve coordenar suas ações e reagir às ações do time adversário de maneira eficiente e eficaz. A dificuldade do jogo é aumentada com a adição de unidades controladas pelo

computador. Essas unidades podem atacar heróis controlados pelos jogadores e fornecer XP e ouro quando mortas. Além disso, podem existir itens que os jogadores podem adquirir com seu ouro a fim de melhorar seus heróis ou penalizar os heróis adversários. Adicionalmente, o *meta-game* desses jogos é alterado de tempos em tempos pelos seus desenvolvedores através de novos *patches*. *Meta-game* pode ser definido como o jeito, eleito informalmente pela comunidade de jogadores, o mais "eficiente" de se jogar e é altamente influenciado por mudanças de *patches*. *Patch* seria uma versão do jogo na qual se altera alguma característica dos heróis ou itens. Nesses *patches*, as habilidades e características de cada herói ou item podem ser alterados. O *meta-game* pode ser alterado também pelo comportamento dos jogadores. Por exemplo, algum jogador "descobre" uma determinada característica do jogo, e começa a explorá-la. Se der certo, os demais jogadores começam a fazer o mesmo. Enfim, MOBAs apresentam um vasto espectro de possíveis comportamentos, ou seja, dominar a técnica desses jogos pode ser bastante desafiador. A Figura 1.1 mostra a interface do jogo Dota 2 na perspectiva de um jogador. Note que na parte superior da tela, os heróis escolhidos por cada equipe são mostrados.

Figura 1.1: Tela mostrando a interface do jogo Dota 2. O painel superior mostra os heróis escolhidos por cada equipe.

1.3 Problema

Em jogos MOBA, como os heróis disponíveis, assim como suas habilidades, são bem variados, grande parte da estratégia envolve a escolha dos mesmos pelos jogadores. O número de diferentes *line-ups* (seleção dos heróis) que uma partida de Dota 2, por exemplo, pode apresentar é consideravelmente grande. Nesse jogo, o número de heróis disponíveis é 113, até a data de 9 de abril de 2017. Dessa forma, o número de *line-ups* possíveis seria dado por:

$$L = C(113, 5) \times C(108, 5), \tag{1.1}$$

$$L = \frac{113!}{108!5!} \times \frac{108!}{103!5!}, \tag{1.2}$$

$$L = \frac{113!}{103!(5!)^2}, \tag{1.3}$$

$$L = 1.56 \times 10^{16}, \tag{1.4}$$

ou seja, há 1.56×10^{16} *line-ups* possíveis para uma partida de Dota 2.

Uma boa escolha de herói deve levar em consideração a sinergia do mesmo com os seus aliados e a vantagem que o mesmo possui sobre os adversários. Além disso, existem funções que determinados heróis exercem e que não devem ser esquecidas durante a fase de escolha dos mesmos. Por isso, dados dois times que possuem o mesmo nível de habilidade, o *line-up* pode ser decisivo a ponto de fornecer a uma das equipes uma vantagem antes mesmo da partida começar.

Obviamente existem inúmeros outros fatores que podem influenciar na performance de um time e no resultado de uma partida como nível de experiência dos jogadores e perfomance individual de cada um deles durante a partida. Porém, nesse trabalho, pretende-se focar apenas nas escolhas de heróis como fator influenciador para o desempenho de um time durante a partida.

1.4 Objetivos

Como mostrado na Seção 1.3, a quantidade de possibilidades de *line-ups* para um jogo MOBA como o Dota 2 é extremamente grande. E como cada herói apresenta

habilidades e características muito distintas uns dos outros, podendo tanto ser combinadas com os heróis aliados quanto utilizadas para anular os heróis inimigos (relação de *counter*), o efeito dessas combinações pode afetar e muito as chances de cada time obter sucesso na partida. Dado esse cenário, seria desejável um mecanismo que analise dados de partidas já executadas e extraia informações úteis sobre as mesmas. A partir dessas informações, seria possível avaliar o quão bom é o *line-up* de uma equipe ou até mesmo recomendar heróis para jogadores baseando-se nas escolhas dos demais jogadores.

Objetivo Geral Dado esse cenário, o objetivo desse trabalho é implementar um sistema que auxilie os jogadores de MOBA na escolha de heróis para uma partida a partir de dados coletados de um grande número de partidas prévias.

Objetivos Específicos Mais especificamente, pretende-se:

- Coletar dados de partidas executadas de maneira categorizada (por nível de habilidade, modo de escolha de heróis e *patch*). No caso, seriam utilizados dados de partidas de Dota 2 como estudo de caso;
- Adotar técnicas de aprendizado de máquina a fim de extrair informações sobre escolhas de *line-ups* e suas chances de vitória;
- Utilizar essas informações para, principalmente, responder os seguintes questionamentos:
    1. Dado que alguns heróis já foram escolhidos, quais heróis são recomendados para se ter a maior chance de vencer? Dessa forma, um sistema de recomendação de heróis seria desenvolvido.
    2. Dada uma determinada escolha de heróis, qual equipe tem mais chance de vencer? Isso seria possível através de um modelo de predição de partidas.
- Desenvolver uma metodologia de testes de perfomance para o sistema de recomendação utilizando o modelo de predição.

1.5 Estrutura do Trabalho

Este trabalho está organizado da seguinte forma: no próximo capítulo, discutem-se alguns trabalhos relacionados na área. No Capítulo 3, são explicados alguns conceitos teóricos relacionados aos mecanismos de aprendizado de máquina utilizados no trabalho.

No Capítulo 4, apresenta-se a metodologia adotada, descrevendo o mecanismo de coleta de dados, o sistema de recomendação e a rede neural desenvolvida para a predição da vitória. Finalmente, o Capítulo 5 apresenta os resultados experimentais e o Capítulo 6 traz as conclusões.

Capítulo 2

Trabalhos Relacionados

No presente capítulo, discute-se artigos, trabalhos e sistemas relacionados aos temas tratados nesta dissertação, como jogos digitais e sistemas de recomendação.

2.1 Pesquisa em Jogos Digitais

Jogos digitais têm atraído a atenção da comunidade acadêmica. Os jogos MOBA em especial, dado a sua complexidade e desafios, têm sido um importante e amplo campo de estudo. Em Drachen et al. [2014] são estudados os padrões de exploração e movimentação dos jogadores em um mapa de Dota 2. Através de três análises diferentes dos dados de *replays* de partidas, foi possível estabelecer relação entre a movimentação e o nível de habilidade dos jogadores. Por exemplo, percebeu-se que jogadores com maior habilidade se movimentam mais frequentemente no mapa do jogo e ocupam esse espaço de maneira mais eficiente.

Yang et al. [2014] apresentam uma abordagem baseada em dados para descobrir padrões em táticas de combate que são comuns entre as equipes vencedoras nos jogos MOBA. O combate foi modelado como uma sequência de grafos e padrões de extração que prevê resultados bem-sucedidos não apenas de combate, mas da partida como um todo.

Nuangjumnong and Tinnawat [2014] fizeram uma correlação positiva entre o papel desempenhado por jogadores de MOBA durante as partidas e o estilo de liderança desses mesmos jogadores na vida real. O estudo acabou concluindo que o exercício de papéis específicos nos jogos MOBA pode levar ao desenvolvimento de certos estilos de liderança.

Eggert et al. [2015] investigam a aplicabilidade de mecanismos de aprendizado de máquina supervisionado para classificar o comportamento de jogadores em termos de papéis específicos e comumente aceitos, mas não formalmente bem definidos, dentro de uma equipe de jogadores do jogo Dota 2.

Em jogos MOBA existem cenários possíveis dentro de uma partida: *snowballing* e

comebacks são dois deles. No chamado *snowballing*, alguma equipe ou jogador tem uma sequência de jogadas acertadas e uma grande vantagem é criada sobre o oponente. No *comeback*, uma equipe, mesmo estando em desvantagem, consegue reverter a situação desfavorável no jogo e vencer a partida. A taxa de ocorrência desses dois cenários é extremamente importante para garantir imparcialidade e engajamento nesses jogos. Li et al. [2017] apresentam um sistema de análise visual para ajudar os designers de jogos a encontrar eventos importantes e parâmetros de jogo que resultam em ocorrências de *snowballing* ou *comeback* nos dados de jogos MOBA.

Outra análise foi feita por Batsford [2013]: em jogos MOBA, é comum a presença de uma parte do mapa contendo unidades controladas por computador (frequentemente conhecidas como neutrais, por não serem de nenhum time) que, ao serem mortas, fornecem ouro e experiência (XP) ao jogador. Essas partes do mapa são conhecidas como *jungles* (mata ou floresta, em inglês). O trabalho de Batsford [2013] utiliza redes neurais e algoritmos genéticos para calcular rotas ótimas que os jogadores podem realizar nas *jungles* a fim de maximizar a eficiência de seu *farm* (o termo *farm* se refere ao ato de um herói matar uma unidade inimiga ou neutra).

Figura 2.1: Tela mostrando jogadores matando os neutrais da parte do mapa denominada *jungle*. Como existem vários pontos como esse mostrado, o trabalho de Batsford [2013] ajuda a criar rotas ótimas entre esses para que o jogador maximize seu *farm*.

Outro aspecto explorado por Kang, Dae-Ki and Kim, Myong-Jong [2015] é a previsibilidade de resultados de partidas de *League of Legends*. Utilizando treinamentos a partir de dois tipos de modelos distintos, os autores alcançaram resultados satisfatórios: os resultados da avaliação de desempenho indicam que os modelos adotados neste trabalho são eficazes na previsão de resultados reais de partidas de LOL. Sistemas de predição,

inclusive, são utilizados para outros tipos de esporte. Huang and Chang [2010] utilizaram redes neurais a fim de criar um modelo de predição para partidas de futebol. Utilizando o modelo criado foi possível atingir uma taxa de acurácia de 76,9% se empates forem excluídos.

Alguns trabalhos com jogos MOBA também têm sido realizados no Laboratório Multidisciplinar de Pesquisa em Jogos da UFMG. Silva and Chaimowicz [2015b] propõem personagens controlados por computador, ou *Non-Player Character* (NPC), mais eficientes e adequados para jogos MOBA utilizando mecanismos de Inteligência Artificial (IA). Silva et al. [2015, 2017] propõem um mecanismo de ajuste dinâmico de dificuldade para NPCs em MOBA. Silva and Chaimowicz [2015a] propõem um tutor utilizando técnicas de IA para auxiliar jogadores iniciantes em jogos MOBA.

2.1.1 Panorama atual

Yannakakis and Togelius [2014] realizaram uma pesquisa sobre o panorama atual da aplicação de Inteligência Artificial (IA) e Inteligência Computacional (IC) em jogos. Nesse trabalho, foram analisadas quais técnicas eram mais utilizadas em cada uma das áreas de IA/IC em jogos. Essas áreas foram definidas no seminário de Dagstuhl [Schloss Dagstuhl, 2015] e estão listadas abaixo:

1. Aprendizagem de comportamento de personagem não-jogador
2. Pesquisa e planejamento
3. **Modelagem de Jogadores**
4. Jogos como *benchmarks* de IA
5. Geração procedural de conteúdo
6. Narrativa computacional
7. Agentes críveis
8. Design de jogos assistido por IA
9. IA para jogos em geral
10. **Agentes para jogos genéricos**

Em Yannakakis and Togelius [2014], os autores deixam claro que essas áreas podem se sobrepor e que uma pesquisa pode abranger múltiplas áreas. Levando em consideração o escopo deste trabalho, pode-se afirmar que ele abrange principalmente duas áreas: Modelagem de Jogadores e IA em jogos comerciais. Isso é justificável devido ao fato de que certa maneira de escolher os heróis é informalmente eleita como a melhor em determinado *patch* (*meta-game*). O motivo pode ser tanto comportamental (jogadores famosos, e que são formadores de opinião, começam a utilizar determinado herói por algum razão e a comunidade começa a fazer o mesmo) quanto mecânico (algum herói se tornou mais fraco no *patch*, com isso ele deixa de ser vantajoso). Como serão analisados dados de partidas reais, ambos motivos podem ter influência nos resultados. Com isso, serão modelados o comportamento dos jogadores e técnicas de IA serão utilizadas para recomendar heróis.

No trabalho de Yannakakis and Togelius [2014], foram analisadas quais técnicas eram dominantes e secundárias em cada área. Para cada área tratada nesse trabalho, estão mostrados os resultados da pesquisa na Tabela 2.1.

Tabela 2.1: Técnicas de IA/IC mais aplicadas às áreas de estudo em jogos relevantes para este trabalho. (•) significa que a técnica é dominante na área. (○) significa que a técnica é secundária. Dados retirados de Yannakakis and Togelius [2014]

Técnica	Modelagem de Jogadores	IA em jogos comerciais
Computação Evolucionária	•	
Aprendizado Reforçado		
Aprendizado Supervisionado	•	○
Aprendizado Não-Supervisionado	•	
Planejamento		•
Busca em Árvores		•

Pode-se perceber que na área de Modelagem de Jogador, as técnicas de Computação Evolucionária, Aprendizado Supervisionado e Aprendizado Não-Supervisionado são dominantes. Já em IA em jogos comerciais, a área de Busca em Árvores e Planejamento são dominantes e Aprendizado Supervisionado é secundária.

2.2 Sistemas de Recomendação

Sistemas de Recomendação são ferramentas e técnicas de software que fornecem sugestões para itens a serem usados por um usuário, como descrito por Ricci et al. [2011]. As sugestões podem se referir a vários processos de tomada de decisão, tais como produtos para comprar, músicas para ouvir, ou notícias *on-line* para ler, por exemplo.

Sistemas de recomendação são especialmente direcionados para indivíduos que não possuem experiência pessoal ou capacidade suficientes para avaliar um número potencialmente grande de alternativas que um *e-commerce* pode oferecer, por exemplo. Um outro exemplo é a gigante Amazon.com que emprega um sistema de recomendação para personalizar a loja *on-line* para cada cliente.[Ricci et al., 2011]

Já em 2013, era estimado que 35% do que os consumidores compravam na Amazon.com e 75% do que os usuários assistem no Netflix venham de sugestões de itens baseados em sistemas de recomendação utilizados pelas duas gigantes.[MacKenzie et al., 2013]

2.2.1 Recomendação em jogos MOBA

Como exposto no capítulo anterior, em jogos MOBA, os heróis disponíveis são bem variados e grande parte da estratégia envolve a escolha dos mesmos pelos jogadores. O número de diferentes *line-ups* (escolha dos heróis) que uma partida de Dota 2 pode apresentar é grande, na ordem de 10^{16}. Dado esse cenário, um sistema de recomendação que oriente os jogadores sobre as melhores escolhas pode ser muito útil.

Um trabalho similar ao proposto nesta dissertação foi realizado por Conley and Perry [2013]. Nesse, o autor utilizou dados coletados de partidas reais de Dota 2 para, através de um mecanismo de aprendizado de máquina, criar um sistema de recomendação de heróis. Os mecanismos de aprendizado de máquina utilizados foram regressão logística e *K-nearest neighbors*. A interface do sistema de recomendação de Conley and Perry [2013] está mostrada na Figura 2.2.

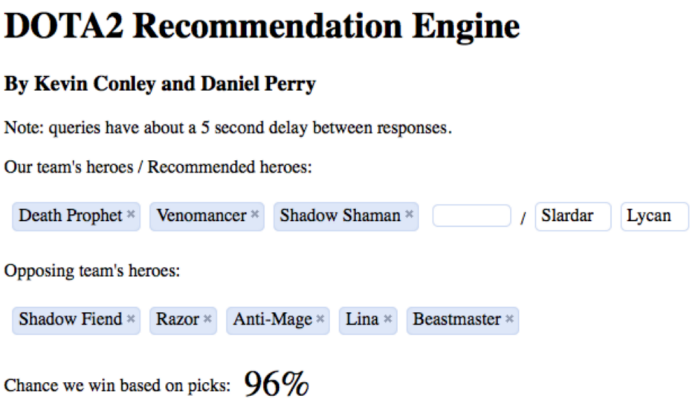

Figura 2.2: Imagem retirada do trabalho de Conley and Perry [2013] mostrando a interface do sistema de recomendação desenvolvido

Os resultados foram promissores com o modelo *K-nearest neighbors* com uma acurácia na predição dos resultados de partidas utilizadas para testes de aproximadamente 70%. A Figura 2.3 mostra o desempenho do mecanismo de regressão logística implementado. Porém, vale ressaltar que o mecanismo de recomendação utiliza o próprio modelo de predição de resultados partidas desenvolvido. Conley and Perry [2013] recomendam heróis para uma equipe usando uma busca gulosa que considera todos os heróis que possam ser adicionados à equipe e classifica os candidatos com a probabilidade de vencer a partida contra o time adversário caso o candidato seja adicionado ao time aliado.

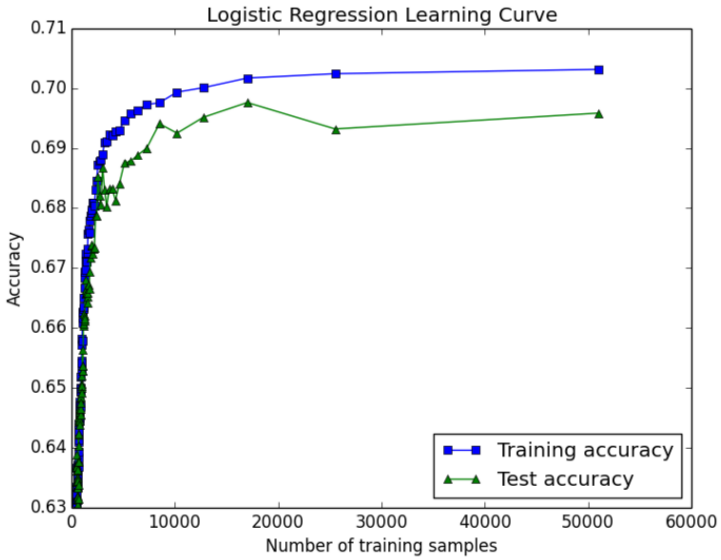

Figura 2.3: Imagem retirada do trabalho de Conley and Perry [2013] mostrando a curva de aprendizado do modelo de regressão logística

Porém alguns pontos relevantes não foram abordados no trabalho de Conley and Perry [2013]:

- No treinamento, foram utilizadas partidas de múltiplos tipos de escolha de heróis. O jogo Dota 2 possui várias modalidades para a escolha de heróis. As principais são: *All pick* (cada jogador escolhe seu herói, sem seguir nenhum tipo de ordem), *Captains Mode* (um capitão de cada time escolhe e bane - retira a possibilidade de escolha - os heróis em uma ordem estabelecida pelo jogo) e *All Random* (cada jogador recebe um herói sorteado aleatoriamente pelo jogo). Como esses modos são bastante distintos, é desejável que sejam tratados de maneiras distintas pelo sistema para não afetar o resultado da recomendação.

- O trabalho não descreve se durante o período de coleta de dados houve mudança de *patch* do jogo. Se houve, o treinamento ficaria comprometido, já que podem existir alterações no *meta-game*. Um treinamento deve ser feito apenas com partidas de um mesmo *patch*.

- O trabalho não discute os resultados específicos do sistema de recomendação, apenas fornece a acurácia do sistema de predição.

O site Dotapicker [Toma and Tibeica, 2015] também oferece um serviço de recomendações de heróis para Dota 2, porém não é detalhado o mecanismo utilizado por tal modelo. Além disso, a ferramenta não é *opensource*.

O portal Dotabuff [Elo Entertainment LLC, 2015] fornece análises estatísticas sobre partidas e jogadores de Dota 2 recuperados pela *Application Programming Interface* (API) do jogo fornecida pela Steam. O site apresenta vários dados interessantes como taxa de vitórias de heróis, inclusive separados por *patch*, e desempenho detalhado dos jogadores nas partidas. Porém, até o momento, o site não utiliza nenhum mecanismo de recomendação para auxiliar os jogadores, sendo realmente apenas uma ferramenta estatística.

2.2.2 Recomendação em *E-commerce*

Como pode-se perceber na seção anterior, ainda não há um grande investimento em pesquisas acadêmicas voltadas para recomendação em jogos MOBA. Porém, nessa dissertação, optou-se por buscar outras áreas em que sistemas de recomendação já têm uma aplicação prática ampla e frequente. Uma dessas áreas é o comércio eletrônico, ou *e-commerce*.

Os sistemas de recomendação são um dos componentes chave no comércio eletrônico e auxiliam os serviços a fornecer um serviço personalizado aos usuários, aumentar o tamanho da compra recomendando acessórios no momento de sua conclusão e aumentar a lealdade e engajamento do usuário [Zhu et al., 2014].

Em estratégias como o marketing por e-mail personalizado e a personalização de páginas da Web, um usuário geralmente é exposto a vários itens simultaneamente. A relevância ou atratividade de um item pode realmente depender dos outros itens mostrados ao usuário. Levando isso em consideração, Zhu et al. [2014] argumentam que deve-se considerar todo o conjunto de itens recomendados como um pacote ao invés de tratá-los individualmente e de forma independente. Exemplos de pacotes são laptop e acessórios, ou filmes dirigidos pelo mesmo diretor. Essa idéia é suportada por dois fatos apontados por Zhu et al. [2014]:

- **Clientes tendem a comprar em pacotes**: Foi relatado que o tamanho médio da ordem para *Amazon.com* é 1,5 e 2,3 para *Walmart.com*.

- $\mathbf{1 + 1 > 2}$: existem inúmeras pesquisas de marketing clássico que relatam a sensação

de vantagem dada ao consumidor quando se oferece pacotes de produtos cuidadosamente selecionados.

Nessa conjuntura, alguns trabalhos foram desenvolvidos a fim de explorar sistemas de recomendação por conjunto (ou pacotes) de items. Zhu et al. [2014] introduziram um modelo e algoritmo para o definido problema da recomendação de pacote, que foi provado ter solução *NP-hard*. Depois de realizar testes *offline* e *online*, os resultados mostraram que o algoritmo desenvolvido pode melhorar os modelos usados como *baseline* em termos de recompensas pré-definidas, como conversões ou receitas. Já Saxena and Gaur [2015] aplicaram o algoritmo *Apriori* para criar recomendações baseadas em conjunto frequentes de itens através da extração de regras de associação.

Ambos os fatos apontados anteriormente, característicos do mundo do marketing, podem ser aplicados também ao domínio desta dissertação. Nos jogos MOBA, por serem disputados por equipes, os jogadores geralmente escolhem um *line-up* de heróis que tem uma boa sinergia entre eles e pontos fortes contra os inimigos (*clientes tendem a comprar em pacotes*). Dada essa sinergia que pode existir entre dois ou mais heróis, fica bem evidente que a partir de um herói já escolhido, existem vantagens em escolher um bom herói que combine e complete o primeiro. Dessa forma, o valor entregue pelo conjunto desses heróis é potencializado (fenômeno *1 + 1 > 2*). Esses dois fenômenos guiarão o desenvolvimento do sistema de recomendação para jogos MOBA deste trabalho.

2.3 Resumo do Capítulo

Jogos Digitais têm atraído a atenção da comunidade acadêmica. Elaboração de NPCs mais adequados, análise de previsibilidade de partidas, e detecção de padrões de comportamento em jogadores são apenas algumas das várias áreas de pesquisas exploradas até o momento.

Sistemas de recomendação se mostram cada vez mais valiosos quando aplicados em cenários onde a quantidade de possibilidades de escolhas dadas aos usuários é notoriamente grande. Empresas como a Amazon e a Netflix têm uma taxa de conversão alta para os sistemas de recomendação adotados por cada uma.

Em jogos MOBA, como Dota 2, o número de possibilidades de escolha de heróis é muito grande, a elaboração de um sistema de recomendação para auxílio de jogadores pode ter muito valor. Especialmente se levados em consideração os usuários mais inexperientes. Apesar disso, ainda não existem muitos estudos voltados para sistemas de recomendação em jogos MOBA. Há algumas ferramentas desenvolvidas e que apresentaram resultados

promissores, embora alguns fatores não tenham sido levados em consideração, como as constantes alterações no *meta-game* através de novos *patches*.

Capítulo 3

Referencial Teórico

O aprendizado de máquina desempenha um papel fundamental nos campos de estatística, mineração de dados e inteligência artificial, fazendo interseção com áreas como engenharia e outras disciplinas. Este capítulo descreve alguns conceitos teóricos que foram utilizados na metodologia do presente trabalho.

3.1 Aprendizado Supervisionado e Não-supervisionado

Em um problema típico de aprendizado de máquina, tem-se uma medida de resultado, ou saída, usualmente quantitativa (como valor de uma ação da bolsa de valores) ou qualitativa (como ocorrência ou não de um ataque cardíaco) que deseja-se predizer o valor baseado em um conjunto de características (como o tipo de dieta e exames clínicos do paciente). Essas características são comumente chamadas de entradas. A partir de um conjunto de dados conhecidos, utilizados para treinamento, observa-se o resultado e as características para um conjunto de objetos (como pacientes). Usando esses dados, é possível a construção de um modelo que viabilizará a predição de resultados para novos e não analisados objetos. Um bom modelo aprendiz (ou *learner*) é aquele que acuradamente prevê tais resultados.

O cenário apresentado acima descreve o que é denominado problema de aprendizado supervisionado. Supervisionado por causa da presença dos valores das variáveis de saída para guiar o processo de aprendizado. Em problemas de aprendizado não-supervisionado, observam-se apenas as características e não se tem nenhuma medida das saídas. A tarefa se torna descrever como os dados são organizados ou agrupados.

Como analisado no capítulo anterior, uma das técnicas mais utilizadas para modelagem de jogador e IA em jogos comerciais é o Aprendizado Supervisionado. No presente trabalho, essa técnica será aplicada já que os dados das partidas contêm, além das entradas (heróis escolhidos), o resultado da partida (qual equipe venceu). A Figura 3.1 mostra uma representação simplificada da modelagem do problema de aprendizado su-

pervisionado do presente trabalho: as entradas são os heróis escolhidos para a partida e o resultado final é a saída.

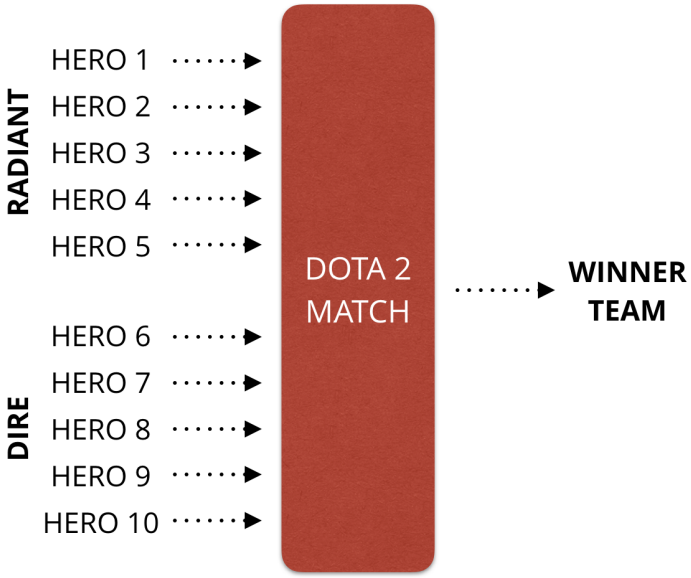

Figura 3.1: Representação do problema de aprendizado supervisionado do presente trabalho: os heróis escolhidos são as entradas e o resultado da partida é a saída

3.2 Regras de Associação

Como citado na Subseção 2.2.2, existem alguns comportamentos apontados pelo marketing clássico que podem ser aplicados ao cenário deste trabalho: (i) *Clientes tendem a comprar em pacotes* e (ii) $1 + 1 > 2$. Dessa forma, a fim de explorar essa característica intrínseca de "pacotes" de heróis em partidas de MOBA, um dos mecanismos de aprendizado aplicado neste trabalho é a extração de regras de associação.

A análise de regras de associação surgiu como uma ferramenta para minerar bases de dados comerciais. Com essa ferramenta, é possível extrair interseções mais frequentes de elementos em uma base de dados. Ou seja, a meta é encontrar subconjuntos de características ou elementos que possuem uma probabilidade alta de aparecerem juntos a partir de um conhecido conjunto de dados.

A forma comum de uma regra de associação é dada por $I \Rightarrow J$, onde I é um subconjunto de elementos, denominado antecedente, e J é um subconjunto adicional, denominado consequente. A implicação dessa regra de associação é que se todos os elementos de I aparecem em um conjunto K, então J é provável de aparecer nesse conjunto também

[Saxena and Gaur, 2015].

Cada regra de associação possui diversas propriedades baseadas na prevalência dos subconjuntos antecedente e consequente na base de dados. O **suporte** de uma regra $S(I \Rightarrow J)$ é a fração de observações da união dos elementos de I e J no conjunto completo de dados K do qual eles foram derivados. A **confiança** $C(I \Rightarrow J)$ de uma regra é seu suporte dividido pelo suporte do antecedente I [Hastie et al., 2009] e pode ser interpretada como uma estimativa para a probabilidade $P_r(I|J)$:

$$C(I \Rightarrow J) = \frac{S(I \Rightarrow J)}{S(I)}, \tag{3.1}$$

Finalmente, o ***lift*** de uma regra é definido pela razão entre a confiança da mesma regra e o suporte do subconjunto consequente. Essa última propriedade indica a independência nas ocorrências dos subconjuntos antecedente e consequente. Um valor maior que 1 indica que antecedente e consequente aparecem mais frequentemente que o esperado, ou seja, a ocorrência do antecedente tem um efeito positivo na ocorrência do consequente [Hastie et al., 2009].

$$L(I \Rightarrow J) = \frac{C(I \Rightarrow J)}{S(J)}, \tag{3.2}$$

3.3 Redes Neurais Artificiais

Houve uma grande *hype* em torno de Redes Neurais Artificiais (serão referidas simplesmente como Redes Neurais neste trabalho) nas últimas décadas, fazendo com que esse mecanismo parecesse algo mágico ou misterioso. Porém, de maneira simplificada, redes neurais são apenas modelos estatísticos não-lineares inspirados pelo funcionamento do sistema nervoso central de um animal. Esses modelos são utilizados para fins de classificação ou regressão de um dado conjunto de entradas. Por exemplo, dada a altura, peso e idade de alguém pode-se classificar essa pessoa como candidata a uma doença cardíaca ou não (classificação). Ou dada a idade e área de uma casa e sua distância a boas escolas, pode-se prever por quanto ela poderia ser vendida (regressão).

Basicamente, uma rede neural possui neurônios de entrada, intermediários (camadas escondidas) e de saída. Entre esses neurônios existem ligações que simulam as sinapses do sistema nervoso animal, como mostrado na Figura 3.2. Nessas redes, cada ligação entre neurônios possui uma propriedade denominada peso. Geralmente, o treinamento de uma rede neural é basicamente o balanceamento dos valores de peso de cada ligação baseado em uma base de dados de treinamento [Bishop, 1995, Hastie et al., 2009].

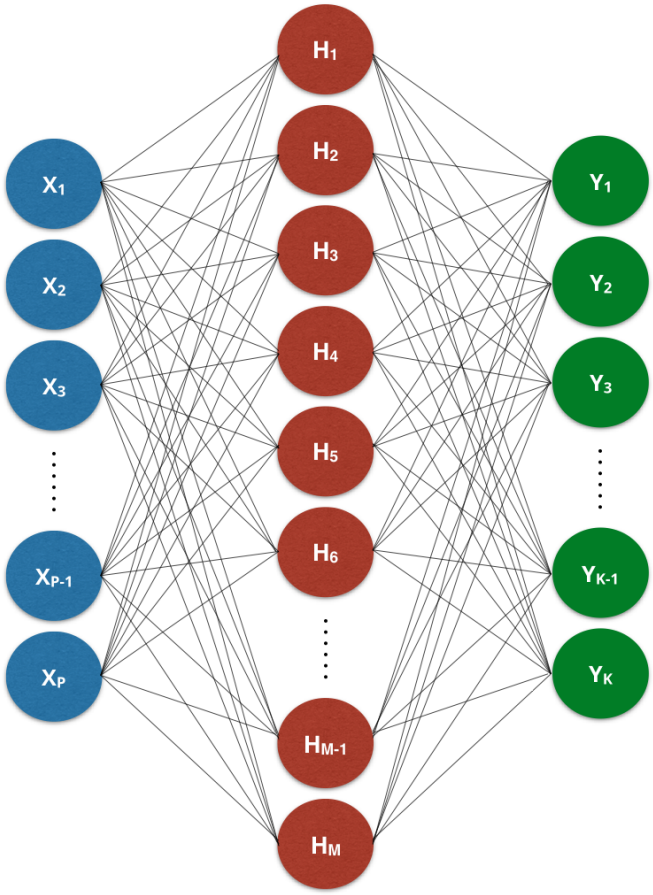

Figura 3.2: Representação genérica de uma rede neural com apenas uma camada escondida (neurônios intermediários). Entre cada camada de neurônios (entrada, saída e escondida) existem ligações que simulam as sinapses do sistema nervoso animal.

Um dos modelos mais conhecidos de redes neurais são os perceptron multi-camadas (MLP): rede neural *feedforward* que possui mais de uma camada de neurônios. Em redes neurais *feedforward*, cada camada se conecta à próxima camada, porém não há caminho de volta. A camada de entrada recebe estímulos que representam os valores de entrada e a camada de saída recebe os estímulos da camada intermediária e constrói a resposta.

O aprendizado nesse tipo de rede é geralmente feito através do algoritmo de retro-propagação do erro, mas existem outros algoritmos para este fim. O algoritmo de retro-propagação se trata de um treinamento supervisionado e se baseia na retro-propagação dos erros para realizar os ajustes de pesos das camadas intermediárias. Com os erros calculados, o algoritmo corrige os pesos em todas as camadas, partindo da saída até a entrada [Bishop, 1995].

3.4 Resumo do Capítulo

Neste capítulo, foram apresentados alguns conceitos teóricos fundamentais para a execução deste trabalho, envolvendo aprendizado de máquina. Foram apresentadas as regras de associação, um mecanismo de extrair e modelar relações entre eventos que pode ser aplicado a diversas naturezas de problema. Além disso, foi retratado o papel fundamental de redes neurais artificiais na área de aprendizado de máquina supervisionado.

Capítulo 4

Metodologia

A metodologia aplicada neste trabalho consiste na coleta de dados de partidas reais de Dota 2 utilizando uma API provida pelos criadores do jogo (*Steam*), e a utilização desses dados para (i) desenvolver um sistema de recomendação de heróis baseado em regras de associação e (ii) treinar uma rede neural para prever o resultado de uma partida entre dois dados times. Foi criada uma aplicação web denominada *House of Dota* que pode ser acessada online (`https://houseofdota.herokuapp.com/`) onde tanto o sistema de recomendação quanto outras informações estatísticas dos treinamentos podem ser acessados. Nas próximas seções, esses módulos serão detalhados.

4.1 Coleta de Dados

A *Steam* expõe uma Web API baseada em HTTP que pode ser usada para acessar muitos recursos. Essa API contém métodos públicos que podem ser acessados a partir de qualquer aplicativo capaz de fazer uma requisição HTTP, como um cliente de jogo ou servidor [Steam, 2017]. Usando a API da Steam voltada para o jogo Dota 2, pode-se coletar dados relacionados a um grande número de partidas. Esses dados são formatados em JSON e incluem quais heróis foram escolhidos por cada equipe, a equipe vencedora, o modo de seleção, data e hora, informações sobre a conexão dos jogadores durante a partida, entre outros. As informações disponibilizadas pela API e que foram úteis para o presente trabalho estão representadas no Código 4.1.

Usando essa API, desenvolveu-se um coletor que obtém as informações de uma série de partidas a cada 5 minutos. A quantidade máxima de correspondências obtidas é 500 e é limitada pela própria API. A partir dessas 500 partidas (coletadas a cada 5 minutos), selecionaram-se apenas as que atendem a alguns requisitos:

Código 4.1: Campos do JSON fornecido pela API da *Steam* que foram úteis para este trabalho

```
{
  "result": {
    "players": [{
      "player_slot": 1, // informa em qual time o jogador pertence
      "hero_id": 1, // informa o id unico do heroi do jogador
      "leaver_status": 0, // status da conexao do jogador durante a partida
    }],
    "radiant_win": true, // true se o radiant venceu, false caso contrario
    "start_time": 1908203, // horario do inicio da partida
    "match_id": 1, // id unico da partida
    "lobby_type": 0, // indica qual tipo de partida (por exemplo: publica, tutorial,
        etc.)
    "human_players": 10, // indica quantos jogadores da partida eram humanos
    "game_mode": 1, // indica qual modo de escolha de herois
  }
}
```

- **Partidas com nível de habilidade muito alto**: Assim, garante-se que os jogadores dessas partidas, utilizadas como treinamento, tenham um conhecimento considerável sobre o jogo e sobre os heróis escolhidos.

- **Sem desconexões**: Durante o curso das partidas, pode haver desconexões de alguns jogadores (queda de conexão com internet, abandono de partida, etc). Essas partidas são excluídas pois seus resultados podem sofrer influência da ausência, mesmo que temporária, de alguns jogadores.

- **Modos de escolha com todos os heróis disponíveis**: Como é desejado que se faça a análise de qualquer associação entre heróis, são utilizadas apenas partidas que possuem todos os heróis disponíveis para escolha no início da mesma.

- **Partidas públicas com todos os jogadores humanos**: Há modos de partidas no Dota 2 que não são públicas (não são acessíveis pela API) e que possuem jogadores NPCs. Como o sistema de recomendação baseia-se na modelagem do comportamento dos jogadores utilizando os heróis, deseja-se que todos esses sejam humanos.

Através dessa metodologia, foi possível a construção de uma base de dados na qual partidas relevantes para o problema têm 10 heróis distribuídos entre os times *radiant* e *dire* (nomes dos dois times de jogo Dota 2) e tem-se a informação de qual time venceu, como mostrado na Figura 4.1.

Infelizmente, as APIs da *Steam* para Dota 2 não fornecem informações sobre o *patch* das partidas. Como o *meta-game* pode ser alterado consideravelmente com uma atualização de *patch*, é desejável que se registre essa informação de cada uma das partidas coletadas. Dessa forma, foi desenvolvido um *crawler* que extrai as informações de atualização de *patch* da página `http://dota2.gamepedia.com/Game_Versions` e as registra no banco de dados. Assim, quando uma partida é coletada, pode-se determinar seu *patch*

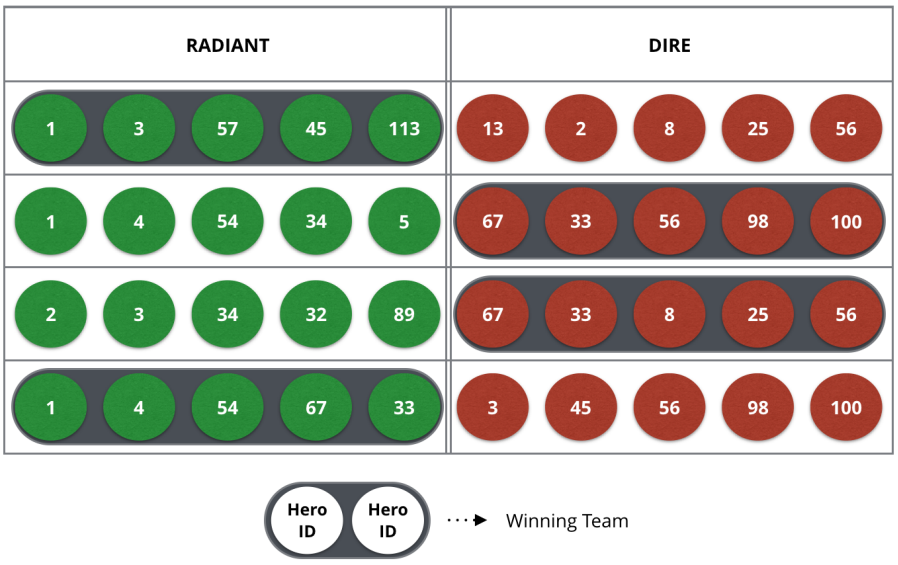

Figura 4.1: Representação do conjunto de dados que foram coletados das APIs: cada partida tem 10 heróis distribuídos entre os times *radiant* e *dire* e tem-se a informação de qual time venceu.

pela data na qual foi jogada. Assim que um *patch* é lançado, as versões anteriores se tornam indisponíveis para os jogadoes.

A partir desses dados coletados, foi possível a criação da base de dados utilizada neste trabalho. Utilizou-se um esquema de dados relacional. Basicamente, tem-se três entidades (como mostrado na Figura 4.2):

- *Match*: dados gerais de uma partida como a data de realização, modo de escolha de heróis e qual o time vencedor.
- *Slot*: dados de uma determinada "vaga" da partida como o herói utilizado e a qual o time ele pertencia.
- *Patch*: dados de uma determinada versão do jogo como a data de atualização.

4.2 Sistema de Recomendação usando Regras de Associação

Conforme mencionado em seções anteriores, regras da associação são comumente usadas para identificar conjuntos de itens frequentes de grandes quantidades de dados.

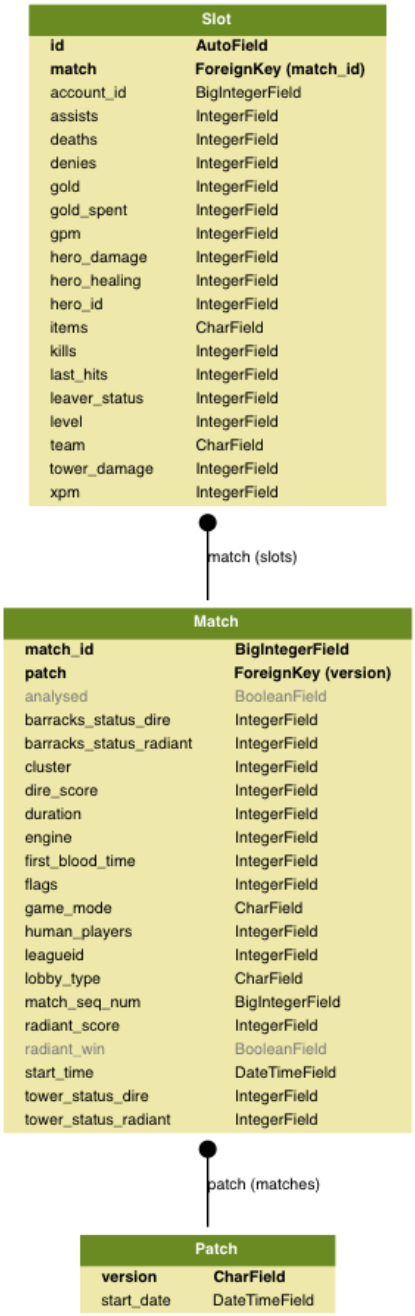

Figura 4.2: Representação do esquema de dados utilizados neste trabalho: *Match*, *Slot* e *Patch* são as principais entidades.

Dessa forma, neste trabalho, utilizaram-se regras de associação a fim de recomendar heróis para um dado *line-up* em Dota 2.

A fim de extrair essas regras de associação, foi feita uma preparação do conjunto de dados coletado das APIs e armazenado no banco de dados. Uma vez que se deseja recomendar heróis baseados em aliados e inimigos, dois conjuntos de regras de associação seriam interessantes: um para as associações entre heróis que ganharam uns com os outros e um para os heróis que ganharam contra um herói em particular. Dessa forma,

dois conjuntos de dados são analisados separadamente (essa preparação é mostrada na Figura 4.3):

- Times vencedores: composição dos times que venceram suas respectivas partidas. Esse conjunto será utilizado para extrair regras de associação para recomendar baseado em aliados e será refereciado como W. Cada partida (ou time vencedor) é representado por um *array* de 5 posições contendo os IDs dos heróis vencedores. Ex: [1, 3, 45, 57, 113].
- Times completos com informação de quais heróis venceram/perderam: composição dos times perdedores/vencedores de suas respectivas partidas. Esse conjunto será utilizado para extrair regras de associação para recomendar baseado em inimigos e será referenciado como M. Heróis que perderam são representados pelo valor negativo de seu ID. Dessa forma, cada partida é representada por um *array* de 10 posições contendo os IDs dos heróis. Ex: [-56, -25, -13, -8, -2, 1, 3, 45, 57, 113]

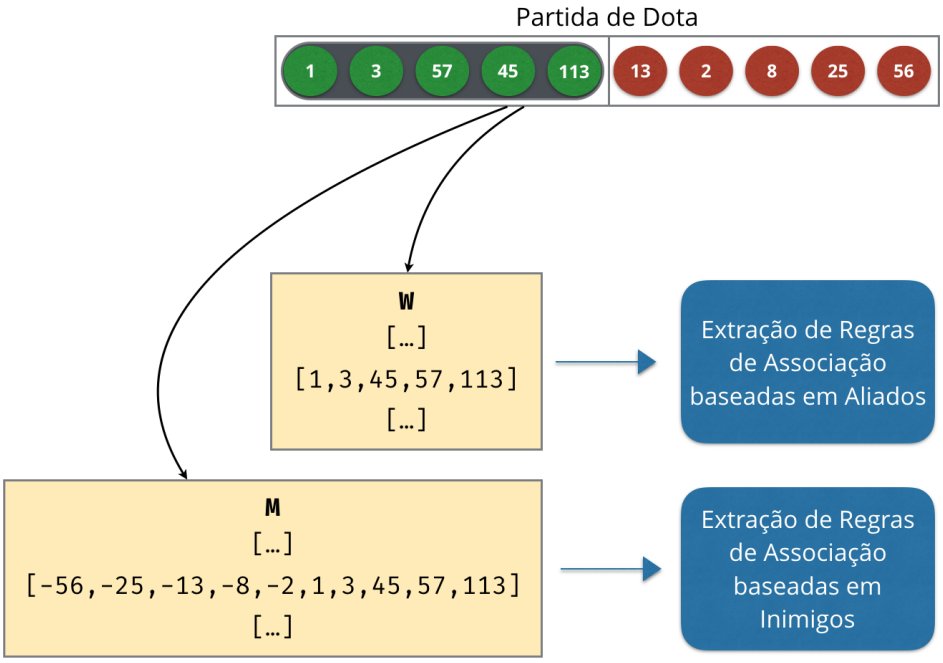

Figura 4.3: Representação da preparação feita nos dados das partidas para se extrair as regras de associação através dos dois conjuntos de dados W e M.

A fim de extrair as regras de associação, o sistema de recomendação proposto utilizou o Algoritmo *Apriori*. O principal objetivo do algoritmo (e de mineração de dados, em geral) é extrair informações úteis de grandes quantidades de dados. O algoritmo visa encontrar as regras que satisfaçam um limite de suporte mínimo e/ou um limite de confiança mínimo [Saxena and Gaur, 2015]. O *Apriori* utiliza uma abordagem "de baixo

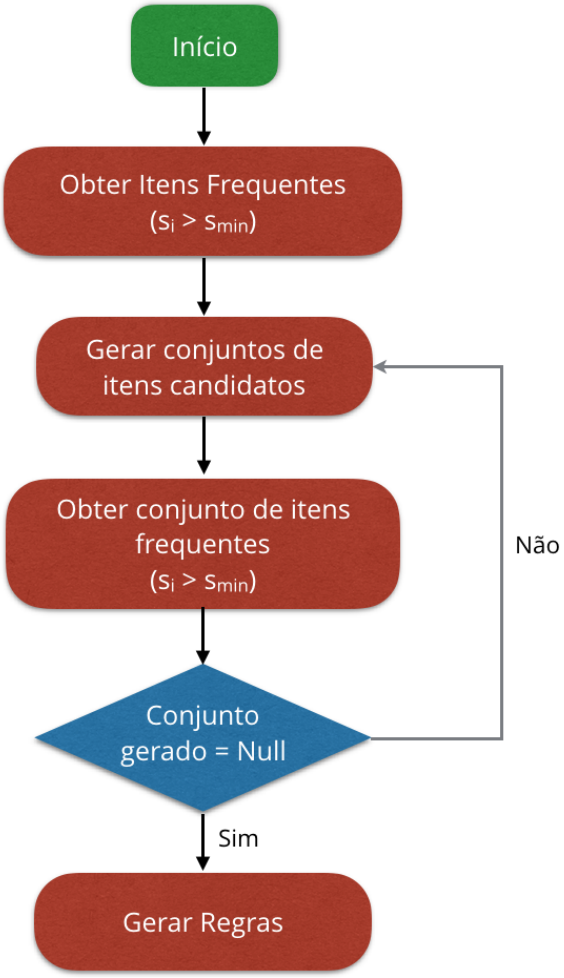

Figura 4.4: Uma visualização simplificada do algoritmo *Apriori*: o algoritmo utiliza uma abordagem "de baixo para cima", através da qual subconjuntos freqüentes são estendidos um item por vez. O algoritmo termina quando nenhuma extensão adicional bem-sucedida é encontrada.

para cima", através da qual subconjuntos frequentes são estendidos um item por vez. O algoritmo termina quando nenhuma extensão adicional bem-sucedida é encontrada.

O primeiro passo do algoritmo *Apriori* é realizar a contagem de ocorrências dos itens para determinar os conjunto de itens frequentes de tamanho unitário. Os passos posteriores consistem de duas fases: primeiro, os conjunto de itens frequentes de tamanho $k-1$, encontrados no passo anterior são utilizados para gerar os conjuntos de itens potencialmente frequentes, os conjuntos de itens candidatos de tamanho k. Na sequência, é realizada uma nova busca no banco de dados, contando-se o suporte de cada candidato. Enquanto existem novos conjuntos de itens que satisfaçam o limite de suporte mínimo e/ou um limite de confiança mínimo dado, o algoritmo tenta estendê-los e procurar novos candidatos. Se não é mais possível encontrar novos conjuntos de itens frequentes, o algoritmo para e as regras encontradas são geradas. Uma visão geral simplificada desse algoritmo é mostrada na Figura 4.4.

Tabela 4.1: Número de regras de associação extraídas para o *patch* 7.06d categorizadas pelo seu valor de suporte

Intervalo de valor de Suporte	Número de Regras de Associação Extraídas
$10{,}0\% < s$	8
$1{,}00\% < s < 10{,}0\%$	146
$0{,}10\% < s < 1{,}00\%$	3096
$0{,}05\% < s < 0{,}10\%$	2573
$0{,}02\% < s < 0{,}05\%$	14316
$0{,}01\% < s < 0{,}02\%$	29786
Total	49925

Para o primeiro conjunto, foi fornecida apenas a informação das equipes vencedoras W ao algoritmo e extraiu-se as associações de qualquer tamanho possível, com um tamanho máximo de 5 (número de heróis em uma equipe). Uma vez que o número de *line-ups* possíveis é consideravelmente grande, adotou-se um suporte mínimo relativamente pequeno de $S_{min} = 0,01\%$ para o Algoritmo *Apriori*. Assim, para que a associação seja considerada, essa precisa estar presente 1 vez em cada 10000 do conjunto de times vencedores W. Esse valor mínimo de suporte é justificado quando se analisam as regras de associação extraídas. Para o *patch* 7.06d, encontrou-se um total de 49925 regras de associação. A maior parte dessas (29786) tem um valor de suporte entre 0,01% and 0,02%, como mostrado na Tabela 4.1.

Para o segundo conjunto, foi fornecida a informação de ambas as equipes M (incluindo a informação de quais heróis venceram/perderam) ao algoritmo e extraiu-se associações de tamanho 2 entre heróis e seus oponentes. Usou-se o mesmo valor para suporte mínimo ($S_{min} = 0,01\%$).

Uma vez que as regras de associação foram extraídas, foi possível recomendar heróis com base em escolhas já feitas. Foram propostos dois conjuntos diferentes de cinco heróis recomendados: um baseado em aliados e outro baseado em inimigos já escolhidos na partida do usuário.

- **Baseado em Aliados**: dado o conjunto A, com tamanho s, de heróis aliados já escolhidos, procura-se por regras de associação R_W, extraídas de W, de tamanho máximo $s + 1$ que contenham qualquer combinação de A, denominada A'. Todas as regras R_W seriam compostas por $A' \Rightarrow r$, sendo r um possível herói recomendado para o usuário. Conhecendo R_W, ordenam-se essas regras de associação em ordem decrescente baseado em alguma propriedade das mesmas e recomendam-se os 5 primeiros r's (as regras de associação de maior tamanho são priorizadas). As propriedades utilizadas para ordenação serão detalhadas no próximo capítulo e, no caso da recomendação baseada em aliados, serão denominadas métricas de aliados.

- **Baseado em Inimigos**: dado o conjunto E de heróis inimigos já escolhidos, procura-se por regras de associação R_M, extraídas de M, de tamanho 2 que contenham algum herói $-e$ de E. Foram utilizadas associações de tamanho 2 porque relações de *counter* são relações de 1 para 1. Todas as regras R_M seriam compostas por $-e \Rightarrow r$, sendo r um possível herói recomendado para o usuário. A simbologia $-e$ é utilizada apenas para representar que a associação representa uma relação de oposição (r é uma boa opção contra o oponente e). Conhecendo R_M, ordenam-se essas regras de associação em ordem decrescente baseado em alguma propriedade das mesmas e recomendam-se os 5 primeiros r's. As propriedades utilizadas para ordenação serão detalhadas no próximo capítulo e, no caso da recomendação baseada em inimigos, serão denominadas métricas de *counter*.

No Capítulo 5, serão detalhadas as métricas de aliados e *counters* que foram utilizadas para ordenar as regras de associação e selecionar os conjuntos recomendados.

4.3 Rede Neural para Predição de Partidas

Dado que o objetivo principal de um sistema de recomendação para a seleção de heróis é ajudar o usuário/jogador na escolha de um herói que o levará à vitória, uma maneira de avaliar uma recomendação é verificar se o *line-up* final tem boas chances de vencer a partida. Levando isso em consideração, seria desejável ter algum tipo de predição do resultado do jogo.

No caso deste trabalho, como os resultados das partidas no banco de dados são conhecidos, uma técnica de aprendizagem supervisionada foi utilizada para desenvolver um modelo de predição. As entradas são os heróis escolhidos (para cada equipe) e a saída é o resultado da partida (qual time ganhou), como mostrado na Figura 3.1.

Seguindo as idéias de Huang and Chang [2010], o modelo de predição adotado é baseado em um perceptron de múltiplas camadas (MLP) e um treinamento baseado em retro-propagação de erro. A camada de entrada possui um neurônio para cada herói possível, totalizando 113. O valor de entrada para um herói particular $hero_i$ é dado pela Equação 4.1.

$$X_i = \begin{cases} 1, & \text{se } hero_i \in \text{radiant} \\ -1, & \text{se } hero_i \in \text{dire} \\ 0, & \text{caso contrário} \end{cases} \tag{4.1}$$

A camada escondida é composta por 300 neurônios e a camada de saída possui apenas um neurônio que representa o resultado da partida e possui uma função de ativação sigmoide. Para o treinamento, o resultado da correspondência é dado pela Equação 4.2. Uma representação da rede neural adotada é exibida na Figura 4.5.

$$Y = \begin{cases} 1, & \text{se time radiant venceu} \\ 0, & \text{caso contrário} \end{cases} \tag{4.2}$$

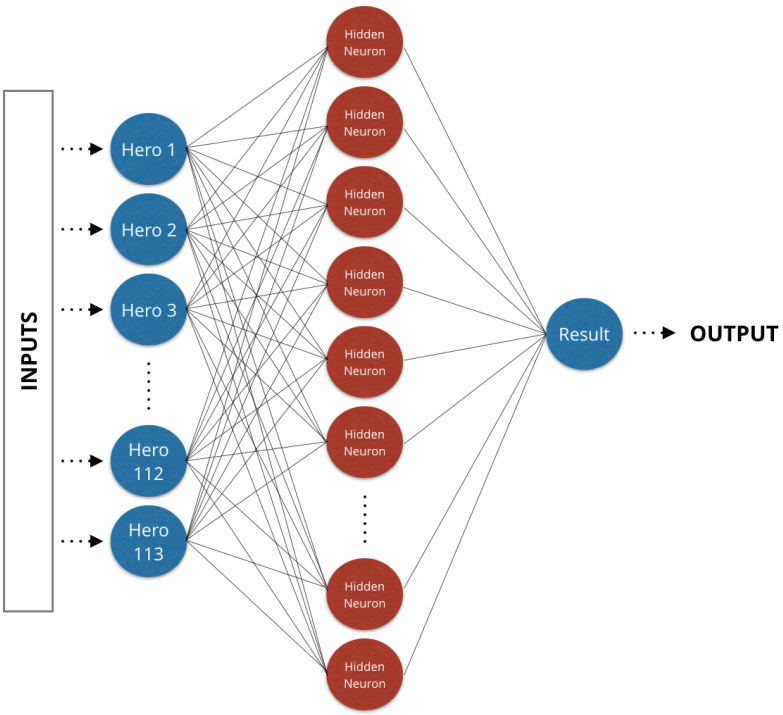

Figura 4.5: Representação do modelo de predição adotado: uma rede neural perceptron de múltiplas camadas. A camada de entrada tem um neurônio por herói, a camada oculta é composta por 300 neurônios e a camada de saída (sigmoide) possui apenas um neurônio.

Vale ressaltar que, com a utilização do modelo de predição baseado em rede neural, pretende-se avaliar a performance do sistema de recomendação através de um mecanismo totalmente independente. Como mencionado no Capítulo 2, Conley and Perry [2013] utiliza o modelo de predição como mecanismo para recomendação e avaliar sua performance.

4.4 *House of Dota*

Como mencionado no início deste capítulo, utilizando os mecanismos anteriormente explicados, foi possível a criação de uma aplicação Web nomeada *House of Dota* que

pode ser acessada online (`https://houseofdota.herokuapp.com/`). Na Figura 4.6 está mostrada a página inicial da aplicação.

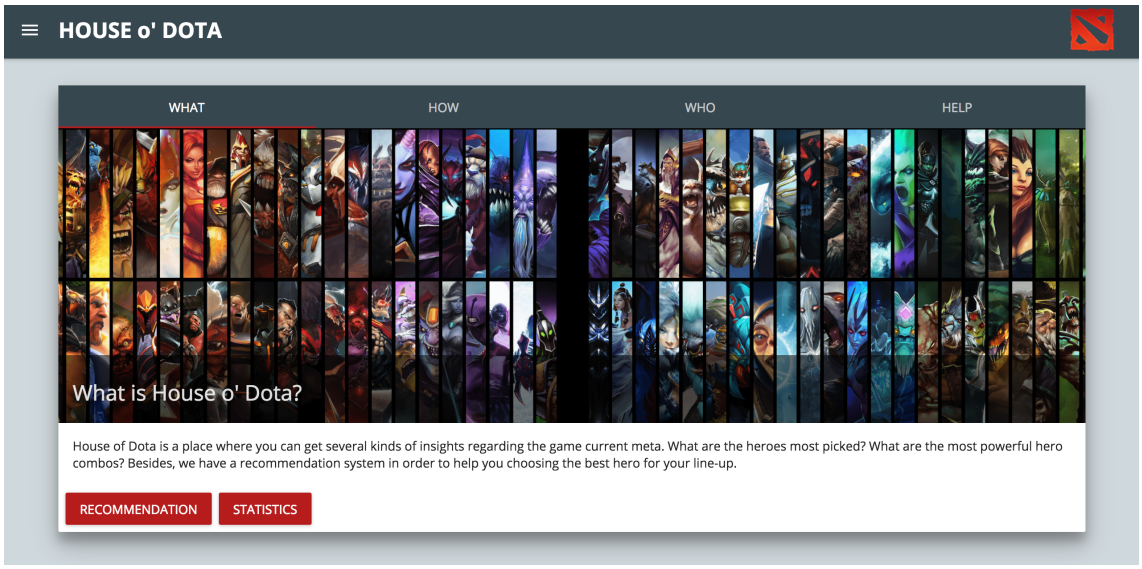

Figura 4.6: Captura de tela da interface do sistema *House of Dota*.

Abaixo estão listadas algumas tecnologias utilizadas na aplicação desenvolvida:

- Django Rest como framework Python back-end (`http://www.django-rest-framework.org/`)
- ReactJS, uma biblioteca JavaScript para criação de interfaces, desenvolvida pelo Facebook (`https://facebook.github.io/react/`)
- PostgreSQL como banco de dados relacional para armazenar dados de partidas, regras de associação extraídas, etc.
- PyBrain, uma biblioteca Python que contém métodos de aprendizado de máquina como Redes Neurais (`http://pybrain.org/`)
- Apyori, uma biblioteca Python que implementa o algoritmo *Apriori* (`https://pypi.python.org/pypi/apyori/1.0.0`)

4.4.1 Tarefas Periódicas

A fim de executar tarefas relacionadas tanto à coleta de dados quanto à execução dos algoritmos de aprendizado de máquina desenvolvidos, foram criadas tarefas periódicas que são executadas em plano de fundo na aplicação:

- Coleta de Partidas (a cada 5 minutos): utilizando as APIs da *Steam*, filtra-se quais partidas das 500 últimas são relevantes para a aplicação e as registra no banco de dados.
- Crawler de *Patches* (a cada 24 horas): utilizando um mecanismo de *crawler* na página `http://dota2.gamepedia.com/Game_Versions`, os *patches* de atualização, juntamente com duas datas de lançamento, são registradas no banco de dados.
- Extração das Regras de Associação (a cada 24 horas): executa-se a preparação dos dados das partidas do *patch* atual e executa-se o algoritmo *Apriori* para extrair regras de associação para recomendação baseada em aliados e inimigos (*counters*).
- Treinamento da Rede Neural (a cada 1 hora): executa-se um treinamento da rede neural de 100 épocas com 70% das partidas do *patch* atual para que o modelo de predição esteja sempre atualizado. As outras partidas (30%) são reservadas para os testes.

4.5 Resumo do Capítulo

Neste capítulo foi detalhada a metodologia aplicada neste trabalho, assim como cada um de seus módulos que estão representados na Figura 4.7. A partir da coleta de dados de partidas reais de Dota 2 utilizando uma API provida pelos criadores do jogo (*Steam*), foi possível (i) extrair regras de associação entre heróis e desenvolver um sistema de recomendação de heróis baseado tanto em aliados quanto em inimigos e (ii) treinar uma rede neural MLP com retro-propagação de erro para prever o resultado de uma partida entre dois dados times. Além disso, foi criada uma aplicação web denominada *House of Dota* que pode ser acessada online (`https://houseofdota.herokuapp.com/`) onde tanto o sistema de recomendação quanto outras informações estatísticas dos treinamentos podem ser acessados.

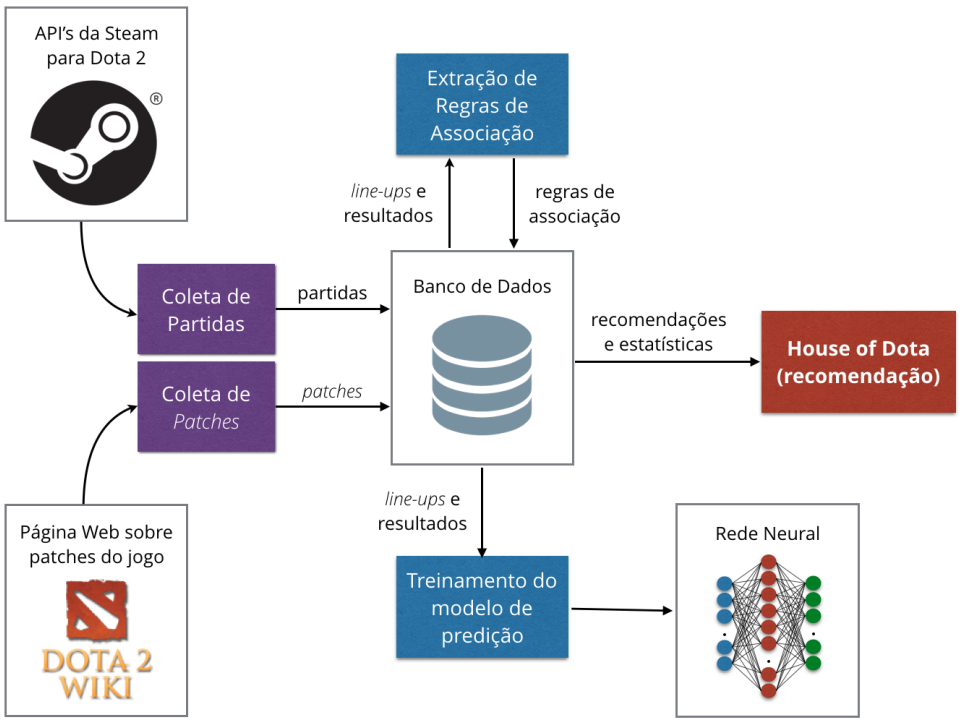

Figura 4.7: Uma visualização simplificada dos módulos deste trabalho: (i) coleta de partidas e *patches*, (ii) sistema de recomendação baseado em extração de regras de associação e (iii) modelo de predição de resultado de partidas utilizando treinamento de rede neural MLP.

Capítulo 5

Experimentos e Resultados

Neste capítulo, serão detalhados os resultados obtidos com o sistema de recomendação e o modelo de predição de partidas baseado em rede neural. Como mencionado, o sistema está disponível para uso em uma aplicação web (Figura 5.1). O usuário pode selecionar os heróis aliados e inimigos e, baseado nisso, são recomendados cinco heróis baseados nos aliados e cinco baseados em inimigos (*counter picks*).

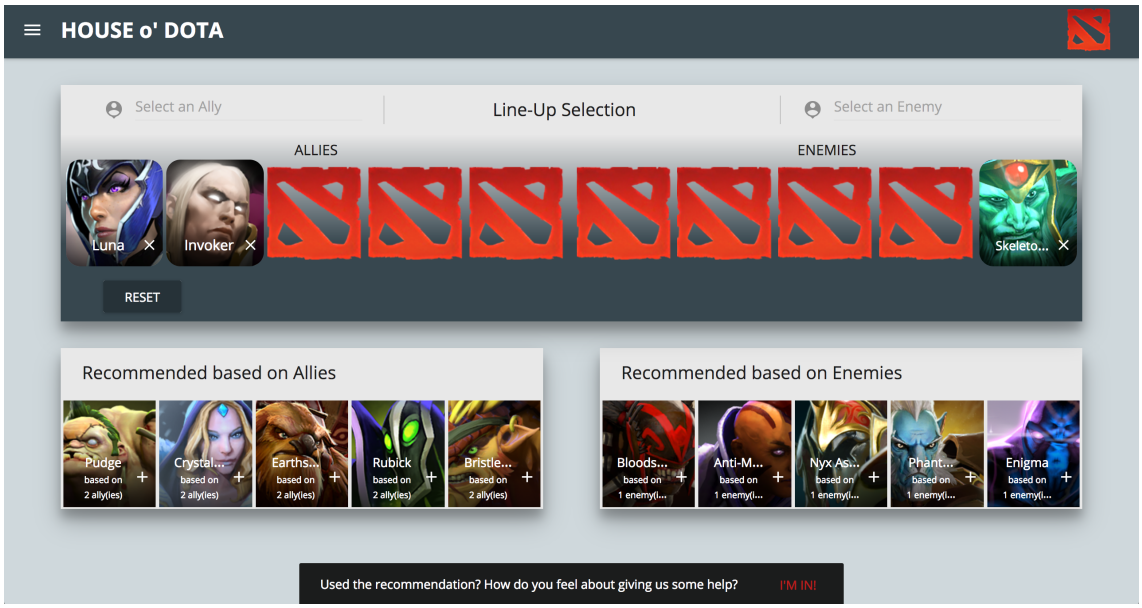

Figura 5.1: Captura de tela da interface do sistema. O usuário já informou dois heróis como aliados e um como inimigo. Dessa forma, o sistema recomenda cinco heróis baseados em aliados e cinco com base em inimigos (*counter picks*).

5.1 Performance do Modelo de Predição com Rede Neural

Para manter a rede neural atualizada com o *meta-game* atual, ela é constantemente treinada com dados de partidas mais recentes. A cada hora, executa-se um treinamento incremental (com 100 épocas) utilizando partidas do *patch* atual. Para o treinamento, é utilizado 70% do conjunto de partidas da base de dados (ordenadas temporalmente). Os outros 30% são utilizados para testar a performance da rede neural. Como este treinamento é um processo em andamento, a rede neural está sempre sendo reciclada para o atual *meta-game* e *patch*. Por exemplo, quando foi detectada a atualização do *patch* 6.88f, a rede neural foi reinicializada e novos treinamentos começaram a ser executados com as partidas coletadas desse *patch*, como mostrado no gráfico da Figura 5.2. Depois de 670 treinamentos (cada um com 100 épocas), em torno de 180 mil partidas eram utilizadas para treinamento e 78 mil para teste. A acurácia da rede neural foi de 85.23% para dados de treinamento e 82.03% para dados de testes ao final do *patch*. Definiu-se como *baseline* a taxa de vitória real para o time *radiant*. Esse time tem uma vantagem estatística sobre o time *dire* em partidas de Dota 2. Utilizou-se essa taxa como *baseline* devido ao fato de que, caso o modelo de predição "chute" que o time *radiant* sempre irá vencer, não ter-se-ia uma chance de 50% de acerto, mas em torno de 62% como mostrado no gráfico.

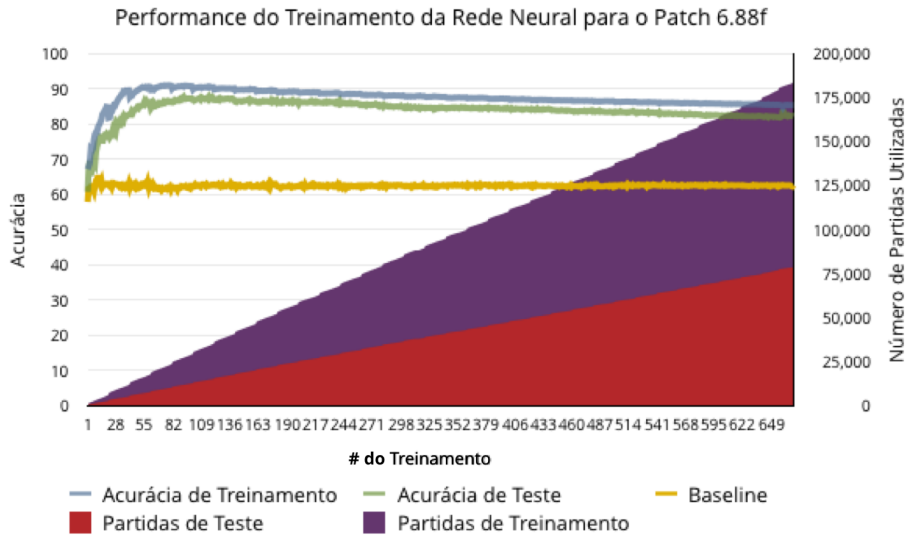

Figura 5.2: Acurácia para a predição do resultado de jogos do *patch* 6.88f usando a rede neural. 70% das partidas foram utilizados para treinamento e 30% para testes. A acurácia da rede neural foi de 85.23% para dados de treinamento e 82.03% para dados de testes ao final do *patch*. O *baseline* representa a taxa de vitória para o time *radiant*, que tem uma vantagem estatística em relação ao time *dire*.

Para cada novo *patch* lançado, o treinamento é executado novamente para que a rede neural esteja sempre atualizada com o *meta-game* atual. Para os últimos dois *patches* (7.06c e 7.06d), a performance da predição de resultados de partidas de suas redes neurais está mostrada nos gráficos das Figuras 5.3 e 5.4. Esses dois patches foram utilizados para a realização dos experimentos do sistema de recomendação que serão explicados posteriormente.

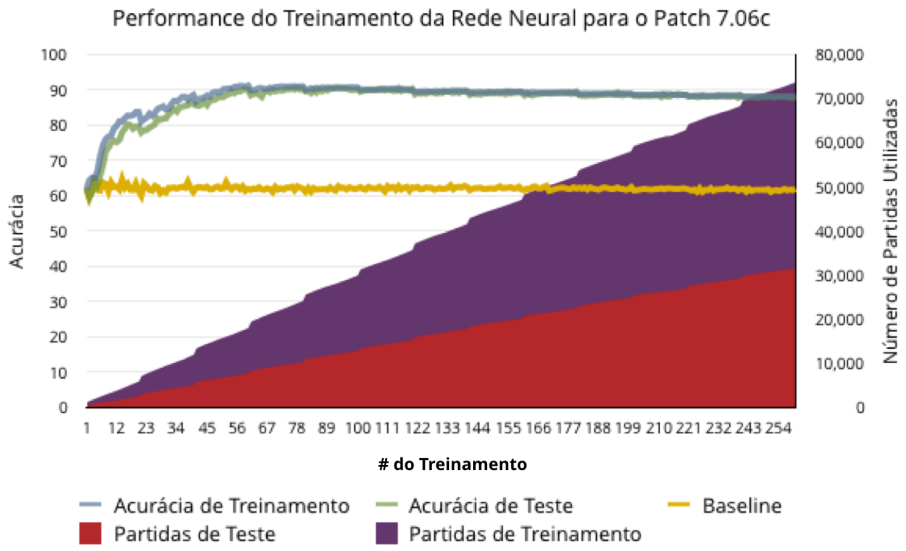

Figura 5.3: Acurácia para a predição do resultado de jogos do *patch* 7.06c usando a rede neural. A acurácia da rede neural foi de 87,97% para dados de treinamento e 87,87% para dados de testes ao final do *patch*.

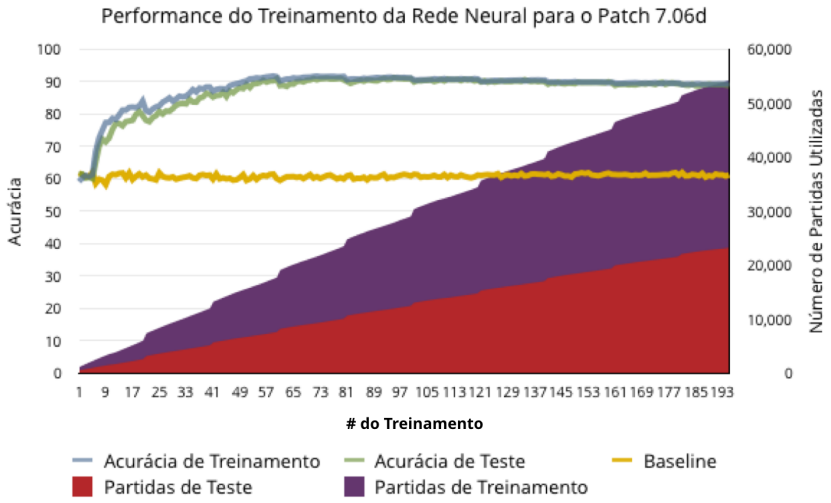

Figura 5.4: Acurácia para a predição do resultado de jogos do *patch* 7.06d usando a rede neural. A acurácia da rede neural foi de 89,28% para dados de treinamento e 88,63% para dados de testes no momento da coleta dessas informações. (O *patch* 7.06d era a versão atual do jogo quando esse texto foi escrito.)

Tabela 5.1: Resultados dos últimos treinamentos da rede neural para cada um dos *patches* coletados a partir do 6.88b

Patch	Partidas de treinamento	Acurácia de treinamento	Partidas de teste	Acurácia de Teste
6.88b	104433	91,707	44757	90,899
6.88c	183497	88,906	78641	84,266
6.88e	84699	88,371	36299	85,523
6.88f	183424	85,234	78610	82,037
7.01	105000	82,253	45000	82,251
7.02	105000	82,925	45000	82,893
7.03	33611	90,583	14404	90,238
7.04	97212	86,737	41661	86,227
7.05	105000	81,469	45000	81,247
7.06	35795	90,806	15340	90,541
7.06b	47376	89,718	20304	89,110
7.06c	73608	87,974	31545	87,871
7.06d	54151	89,289	23207	88,633

Na Tabela 5.1 são mostrados os resultados dos últimos treinamentos da rede neural para cada um dos *patches* coletados a partir do 6.88b.[1] Nota-se que o modelo de predição teve a melhor performance para o *patch* 6.88b com 90,89% de acurácia para partidas de teste. A pior performance ocorreu no *patch* 7.05 com 81,24% de acurácia para partidas de teste.

5.2 Resultados para o Sistema de Recomendação

Antes de se detalhar experimentos executados para testar o desempenho do sistema de recomendação, é importante detalhar o processo de escolha de heróis em uma partida de Dota 2. Inicialmente, um time é selecionado aleatoriamente para começar a escolher. A partir desse momento, os times se revezam na seleção de um herói do grupo disponível até que o *line-up* esteja completo (cada equipe com 5 heróis selecionados). Assim, desenvolveu-se um experimento para simular a mesma lógica onde:

- Equipes se revezam na seleção de heróis do grupo disponível;
- Os heróis inimigos são selecionados de acordo com uma determinada política (serão descritas posteriormente);

[1]Não há dados para o *patch* 7.00 porque o *crawler* desenvolvido apresentou problemas de implementação durante esse *patch*, por isso não se tem dados para o mesmo.

- Os heróis aliados são selecionados usando o sistema de recomendação: obtêm-se as recomendações com base em aliados e inimigos (5 heróis de cada um) e seleciona-se aleatoriamente um herói desse grupo;
- Com o *line-up* completo, utiliza-se a rede neural treinada para prever qual equipe tem maiores chances de ganhar.

A lógica implementada pelo experimento é mostrada no Algoritmo 1. Um ponto desse algoritmo deve ser detalhado: as funções que buscam as recomendações baseadas em aliados ($get\_recommendations$) e inimigos ($get\_counters$) recebem os parâmetros $allies\_metric$ e $counters\_metric$. Esses parâmetros são utilizados para informar ao sistema de recomendação qual propriedade das regras de associação (suporte, confiança, etc.) deve ser utilizada como critério de ordenação das mesmas para, assim, selecionar as melhores e recomendar os heróis.

Algorithm 1 Algoritmo para um experimento único, adotando os parâmetros allies_metric e counters_metric para selecionar as recomendações. Os inimigos são selecionados de acordo com uma determinada política. No final, a rede neural é usada para prever qual time tem mais chances de ganhar a partida com base no *line-up* final.

```
function EXPERIMENT(allies_metric, counters_metric)
    A, E ← []
    while A.size ≠ 5 and E.size ≠ 5 do
        E.insert(some enemy)
        R ← []
        R.insert(get_recommendations(allies_metric))
        R.insert(get_counters(counters_metric))
        A.insert(pick randomly from R)
    end while
    result ← get_nn_prediction(A, E)  return result
end function
```

Foram utilizados dois tipos diferentes de propriedade para cada parâmetro do experimento (a fim de ordenar e selecionar as regras de associação relevantes). Para a recomendação baseada nos aliados A ($allies\_metric$), usou-se o suporte S_W das regras de associação $A' \Rightarrow r$ extraído de dados de equipes vencedoras, sendo A' qualquer combinação de heróis de A (mostrado na Equação 5.1). Por exemplo, suponha que dentre um conjunto de 3 partidas, os heróis x e y venceram juntos uma vez. Dessa maneira, o suporte dessa associação é de 33,33%. A outra métrica para recomendar com base em aliados foi a taxa de vitória do pacote de heróis $A \cup r$, que pode ser calculado como a razão entre o suporte S_W da regra de associação $A \Rightarrow r$ extraído dos dados das equipes vencedoras W e a soma desse mesmo suporte e o suporte S_M da regra $-A \Rightarrow -r$ (A e r perderam suas partidas) extraída de todos os dados das equipes (mostrado na Equação 5.2). Para o mesmo exemplo anterior, suponha que, apesar de vencerem apenas uma vez, os heróis

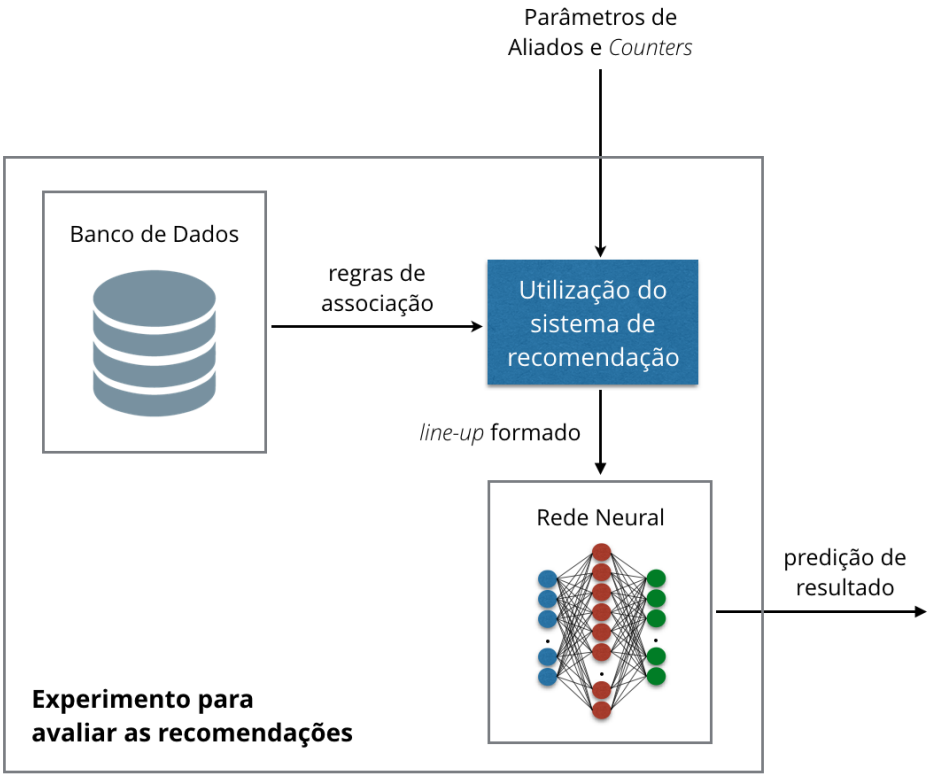

Figura 5.5: Uma visualização simplificada do experimento executado: adotando os parâmetros de aliados e *counters*, utiliza-se o sistema de recomendação para montar um line-up contra um time gerado seguindo uma determinada política. No final, a rede neural é usada para prever qual time tem mais chances de ganhar a partida com base no *line-up* final.

x e y estiveram no mesmo time apenas uma vez também. Dessa maneira, o win_rate dessa associação é de 100,00%.

$$support(A, r) = S_W(A' \Rightarrow r) \tag{5.1}$$

$$win\_rate(A, r) = \frac{S_W(A' \Rightarrow r)}{S_W(A' \Rightarrow r) + S_M(-A' \Rightarrow -r)} \tag{5.2}$$

Para a recomendação baseada nos inimigos E, utilizou-se a confiança C_M de regras de associação $-e \Rightarrow r$ extraídas de todos os dados das equipes (mostrado na Equação 5.3). Por exemplo, suponha que dentre um conjunto de 3 partidas, o herói x venceu uma partida contra o herói y uma vez, mas o herói y estava presente em todas as partidas e perdeu todas. Dessa maneira, a confiança dessa associação é de 33,33%/100%=33,33%. Porém, se z perdeu uma partida também contra o herói x, mas essa foi a única partida que z perdeu, a confiança dessa associação seria de 33,33%/33,33%=100,00%

A outra propriedade para recomendar com base em inimigos foi denominada coeficiente de *counter* (*counter coefficient*) e pode ser calculada como o suporte S_M da regra

de associação $-e \Rightarrow r$ multiplicado pelo valor de *lift* dessa mesma regra, como mostrado na Equação 5.4. Esta segunda propriedade tenta eliminar casos em que a relação entre e e r existe simplesmente porque r é muito escolhido independentemente do adversário. Como o *lift* determina a independência de eventos, multiplicando seu valor pelo suporte da associação $-e \Rightarrow r$, pode-se ter uma maior certeza da relação de *counter* entre esses heróis.

$$confidence(e, r) = C_M(-e \Rightarrow r) = \frac{S_M(-e \Rightarrow r)}{S_M(-e)} \tag{5.3}$$

$$counter\_coefficient(e, r) = S_M(-e \Rightarrow r) * L(-e \Rightarrow r) \tag{5.4}$$

Para cada combinação de parâmetros, executou-se o experimento, representado em Algoritmo 1, 1000 vezes e verificou-se a taxa de sucesso da equipe que usou o sistema de recomendação desenvolvido através do modelo de predição baseado em rede neural.

5.2.1 Sistema de Recomendação vs. Time Aleatório

Nos primeiros experimentos, a política de escolha do time adversário foi aleatória. Os resultados são apresentados nas Tabelas 5.2 e 5.3.

Tabela 5.2: Resultados para os experimentos usando o sistema de recomendação e a rede neural para o *patch* 7.06c. Os números representam as taxas de vitória (com resultados de partidas previstos pela rede neural) para a equipe construída usando o sistema de recomendação com combinação de diferentes tipos de allies e counters metrics. O time adversário foi escolhido aleatoriamente.

		Parâmetro de Counters	
		Counter coefficient	Confidence
Parâmetro de Aliados	Support	52,9%	54,0%
	Win rate	70,4%	**72,0%**

Analisando os resultados para o *patch* 7.06c, pode-se concluir que a pior combinação de parâmetros é suporte para recomendação de aliados e coeficiente de *counter* para recomendação de *counters* com taxa de sucesso de 52.9%. Ou seja, a equipe que usa o sistema de recomendação ganhou apenas 52.9% das partidas simuladas de acordo com a predição da rede neural. No entanto, usando a combinação de parâmetros de taxa de

Tabela 5.3: Resultados para os experimentos usando o sistema de recomendação e a rede neural para o *patch* 7.06d. Os números representam as taxas de vitória (com resultados de partidas previstos pela rede neural) para a equipe construída usando o sistema de recomendação com combinação de diferentes tipos de allies e counters metrics. O time adversário foi escolhido aleatoriamente.

		Parâmetro de Counters	
		Counter coefficient	Confidence
Parâmetro de Aliados	Support	56,9%	60,5%
	Win rate	73,4%	**76,4%**

vitória (*win_rate*) para recomendação de aliados e confiança para a recomendação de *counter*, atingiu-se uma taxa de sucesso de 72.0%.

Analisando os resultados para o *patch* atual (7.06d), pode-se concluir também que a pior combinação de parâmetros é suporte para recomendação de aliados e coeficiente de *counter* para recomendação de *counters* com taxa de sucesso de 56,9%. Ou seja, a equipe que usa o sistema de recomendação ganhou apenas 56,9% das partidas simuladas de acordo com a predição da rede neural. Novamente, usando a combinação de parâmetros de taxa de vitória (*win_rate*) para recomendação de aliados e confiança para a recomendação de *counter*, atingiu-se uma taxa de sucesso maior, com valor igual a 76,4%.

Para fim de testes de sanidade, executou-se o mesmo experimento, porém ambos os times utilizando o sistema de recomendação e ambos escolhendo aleatoriamente. Para ambos os times escolhendo os seus heróis aleatoriamente, alcançou-se uma taxa de vitória de 49,6/50,4%. O resultados para ambos os times utilizando o sistema de recomendação estão mostrados na Tabela 5.4.

Tabela 5.4: Resultados para os experimentos usando o sistema de recomendação para ambos os times e a rede neural para o *patch* 7.06d. Os números representam as taxas de vitória (com resultados de partidas previstos pela rede neural) para uma das equipes construída usando a combinação diferentes tipos de allies e counters metrics.

		Parâmetro de Counters	
		Counter coefficient	Confidence
Parâmetro de Aliados	Support	50,4%	48,8%
	Win rate	49,5%	50,8%

Como esperado, os experimentos utilizando o sistema de recomendação para ambos os times obtiveram um equilíbrio na taxa de vitórias, cada time ficando com aproximadamente 50% de aproveitamento.

5.2.2 Sistema de Recomendação vs. Heróis mais Escolhidos

Nos experimentos seguintes, o time adversário foi escolhido a partir do conjunto dos 10 heróis mais escolhidos no *patch*. Os resultados são apresentados nas Tabela 5.5.

Tabela 5.5: Resultados para os experimentos usando o sistema de recomendação e a rede neural para o *patch* 7.06c. Os números representam as taxas de vitória (com resultados de partidas previstos pela rede neural) para a equipe construída usando o sistema de recomendação com combinação de diferentes tipos de allies e counters metrics. O time adversário foi escolhido a partir do conjunto dos 10 heróis mais escolhidos no *patch*

		Parâmetro de Counters	
		Counter coefficient	Confidence
Parâmetro de Aliados	Support	59,9%	60,2%
	Win rate	71,7%	**74,9%**

Analisando os resultados para o *patch* atual (7.06d) com o time inimigo sendo escolhido entre os 10 heróis mais escolhidos, pode-se concluir também que a pior combinação de parâmetros é suporte para recomendação de aliados e coeficiente de *counter* para recomendação de *counters* com taxa de sucesso de 56,9%. Ou seja, a equipe que usa o sistema de recomendação ganhou apenas 59,9% das partidas simuladas de acordo com a predição da rede neural. Novamente, usando a combinação de parâmetros de taxa de vitória (*win_rate*) para recomendação de aliados e confiança para a recomendação de *counter*, atingiu-se uma taxa de sucesso maior, com valor igual a 74,9%.

5.2.3 Sistema de Recomendação vs. Heróis mais presentes em Times Vitoriosos

Finalmente, nos últimos experimentos, o time adversário foi escolhido a partir do conjunto dos 10 heróis mais presentes em times vitoriosos no *patch*. Os resultados são apresentados na Tabela 5.6.

Analisando os resultados para o *patch* atual (7.06d) com o time inimigo sendo escolhido entre os 10 heróis mais presentes em times vitoriosos, pode-se concluir também que a pior combinação de parâmetros é suporte para recomendação de aliados e coeficiente de *counter* para recomendação de *counters* com taxa de sucesso de 56,9%. Ou seja, a

Tabela 5.6: Resultados para os experimentos usando o sistema de recomendação e a rede neural para o *patch* 7.06d. Os números representam as taxas de vitória (com resultados de partidas previstos pela rede neural) para a equipe construída usando o sistema de recomendação com combinação de diferentes tipos de allies e counters metrics. O time adversário foi escolhido a partir do conjunto dos 10 heróis mais presentes em times vitoriosos no *patch*

		Parâmetro de Counters	
		Counter coefficient	Confidence
Parâmetro de Aliados	Support	45,9%	52,2%
	Win rate	66,3%	**66,8%**

equipe que usa o sistema de recomendação ganhou apenas 45,9% das partidas simuladas de acordo com a predição da rede neural. Novamente, usando a combinação de parâmetros de taxa de vitória (*win_ rate*) para recomendação de aliados e confiança para a recomendação de *counter*, atingiu-se uma taxa de sucesso maior, com valor igual a 66,8%.

5.2.4 Análises dos Resultados

Como mencionado anteriormente, o parâmetro de aliados que apresentou a melhor performance, de acordo com o modelo de predição, foi o de *win rate*. Como mostrado na Equação 5.2, esse parâmetro é dado pela razão entre o suporte S_W da regra de associação $A \Rightarrow r$ extraído dos dados das equipes vencedoras W e a soma desse mesmo suporte e o suporte S_M da regra $-A \Rightarrow -r$ (A e r perderam suas partidas) extraída de todos os dados das equipes. Em contrapartida, o parâmetro suporte é dado apenas pelo suporte S_W da regra de associação $A \Rightarrow r$ extraído dos dados das equipes vencedoras W. Dessa forma, pode-se fazer uma análise sobre a recomendação baseada em aliados: de acordo com o modelo de predição adotado, é mais importante que um herói (ou conjunto de heróis) tenham uma taxa de vitória alta em suas partidas mesmo que essas sejam poucas. O valor alto de suporte S_W da regra de associação $A \Rightarrow r$ não indica isso, ele apenas indica que aquele conjunto de heróis aparece várias vezes nos times vencedores, mas ele podem aparecer também em grande parte dos times perdedores. O valor alto de *win rate* já garante o contrário.

Para o parâmetro de *counters*, a propriedade de confiança apresentou os melhores resultados, para ambos os *patches* analisados. Como mostrado na Equação 5.3, a confiança é dada pela razão entre o suporte S_M da regra de associação $-e \Rightarrow r$ extraído dos dados

M (frequência de partidas nas quais r venceu e) e o suporte S_M da regra $-e$ (frequência de partidas nas quais e perdeu). Por outro lado, o coeficiente de *counter* é dado pelo produto entre suporte S_M e o lift L_M (independência entre os eventos e perder e r vencer) da regra de associação $-e \Rightarrow r$. Por conseguinte, pode-se fazer uma análise sobre a recomendação baseada em inimigos: de acordo com o modelo de predição adotado, é mais importante que um herói (ou conjunto de heróis) tenha uma taxa de vitória alta em relação a um inimigo que não perdeu muito. Isso significa que o sistema de recomendação vai priorizar a recomendação de *counters* para inimigos que tenham uma taxa de vitória alta.

De toda forma, independentemente dos parâmetros utilizados, o sistema de recomendação apresentou uma boa performance em praticamente todos os experimentos. Neles, o time que utilizou sempre apresentou uma vantagem no número de vitórias em relação ao time adversário. Isso demonstra que a utilização do mecanismo de regras de associação apresentou um bom comportamento e pode ser avaliado com um bom potencial no contexto de escolha de heróis para jogos MOBA.

Capítulo 6

Conclusão

Neste trabalho, foi desenvolvido um sistema de recomendação a fim de auxiliar jogadores na escolha de heróis em Dota 2. Ao coletar dados de milhares de partidas de Dota 2, desenvolveu-se um mecanismo baseado em regras de associação que sugere os heróis mais adequados para compor um time de aliados contra um time inimigo. O algoritmo *Apriori* foi utilizado e visa encontrar as regras que satisfaçam um limite de suporte mínimo e/ou um limite de confiança mínimo através de um abordagem "de baixo para cima". Para avaliar a eficácia da recomendação de *line-up*, treinou-se uma rede neural MLP com treinamento baseado em retro-propagação de erro que, dada a formação de duas equipes, foi capaz de prever a equipe vencedora com uma boa precisão de 88,63% para o *patch* atual.

Com relação ao sistema de recomendação, foram executados vários experimentos para determinar quais propriedades das regras de associação deveriam ser utilizadas como parâmetros a fim de maximizar taxa de vitórias usando a rede neural como modelo de predição. Os experimentos simularam uma fase de seleção de heróis para Dota 2: as equipes se revezam para escolher seus heróis. Os heróis inimigos escolheram aleatoriamente e os aliados são escolhidos com base nos heróis recomendados pelo sistema desenvolvido. Usando os parâmetros de *win rate* para recomendar baseado em aliados e a confiança para recomendar baseado em inimigos, atingiu-se uma taxa de vitória de 76,4% de um conjunto de 1000 experimentos para o *patch* 7.06d. Foram executados também experimentos onde o time inimigo foi escolhido baseados nos heróis (i) mais escolhidos e (ii) mais vitoriosos. Os resultados também foram satisfatórios com uma taxa de vitória de até (i) 74,9% e (ii) 66,8%.

Os resultados mostraram que o mecanismo de regras de associação apresentou uma boa performance e pode ser avaliado com um bom potencial no contexto de escolha de heróis para jogos MOBA. Independentemente dos parâmetros utilizados, o sistema de recomendação apresentou uma boa performance em praticamente todos os experimentos. Neles, o time que utilizou sempre apresentou uma vantagem no número de vitórias em relação ao time adversário.

6.1 Trabalhos Futuros

Como trabalhos futuros, podem ser realizados mais experimentos e análises com o sistema. Podem ser avaliadas outras propriedades das regras de associação como parâmetro e também investigar outros mecanismos para melhorar os resultados obtidos. Além disso, seria interessante realizar algum tipo de validação qualitativa com usuários reais. Por exemplo, poderia ser solicitado a um conjunto de jogadores reais que usassem o sistema durante a fase de seleção de algumas partidas. E, no final, esses jogadores poderiam preencher um formulário a fim de ajudar a avaliar a satisfação do usuário e a qualidade do sistema de recomendação.

Referências Bibliográficas

Thomas E Batsford. Calculating Optimal Jungling Routes in DOTA2 Using Neural Networks and Genetic Algorithms. 2013.

Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY, USA, 1995. ISBN 0198538642.

Michael Buro and Timothy M Furtak. RTS games and real-time AI research. *Proceedings of the Behavior Representation in Modeling and Simulation Conference*, pages 63–70, 2004.

Kevin Conley and Daniel Perry. How Does He Saw Me? A Recommendation Engine for Picking Heroes in Dota 2. 2013. [`http://cs229.stanford.edu/proj2013/PerryConley-HowDoesHeSawMeARecommendati onEngineForPickingHeroesInDota2.pdf` accessed in may 15th 2017.].

Renato Luiz De Freitas Cunha, Marlos C. Machado, and Luiz Chaimowicz. Rtsmate: Towards an advice system for rts games. *Comput. Entertain.*, 12(1):1:1–1:20, February 2015. ISSN 1544-3574. doi: 10.1145/2582193.2633441. URL `http://doi.acm.org/10.1145/2582193.2633441`.

Anders Drachen, Matthew Yancey, John Maguire, Derrek Chu, Iris Yuhui Wang, Tobias Mahlmann, Matthias Schubert, and Diego Klabajan. Skill-based differences in spatio-temporal team behaviour in defence of the Ancients 2 (DotA 2). *Games Media Entertainment (GEM), 2014 IEEE*, 2(DotA 2):1–8, 2014. doi: 10.1109/GEM.2014.7048109.

Christoph Eggert, Marc Herrlich, Jan Smeddinck, and Rainer Malaka. Classification of player roles in the team-based multi-player game dota 2. In *Entertainment Computing - ICEC 2015: 14th International Conference, ICEC 2015, Trondheim, Norway, September 29 - Ocotober 2, 2015, Proceedings*, pages 112–125. Springer International Publishing, 2015.

Elo Entertainment LLC. Dotabuff, 2015. URL `https://www.dotabuff.com`. [`https://www.dotabuff.com` accessed in 13 de Novembro de 2015].

Juho Hamari. What is eSports and why do people watch it ? What is eSports and why do people watch it ? *SSRN Working Paper Series*, pages 1–31, 2015.

Travor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, second edition, 2009.

Kou-Yuan Huang and Wen-Lung Chang. A neural network method for prediction of 2006 World Cup Football Game. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, July 2010. ISBN 978-1-4244-6916-1. doi: 10.1109/IJCNN.2010.5596458. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5596458.

John Gaudiosi. Watching other people play video games could soon be a billion dollar industry, 2015. URL http://fortune.com/2015/04/20/esports-billion-dollar-industry/.

Kang, Dae-Ki and Kim, Myong-Jong. Poisson Model and Bradley terry Model for predicting multiplayer online battle games. In *2015 Seventh International Conference on Ubiquitous and Future Networks*, pages 882–887. IEEE, July 2015. ISBN 978-1-4799-8993-5. doi: 10.1109/ICUFN.2015.7182671. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7182671.

Quan Li, Peng Xu, Yeuk Yin Chan, Yun Wang, Zhipeng Wang, Huamin Qu, and Xiaojuan Ma. A Visual Analytics Approach for Understanding Reasons behind Snowballing and Comeback in MOBA Games. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):211–220, jan 2017. ISSN 1077-2626. doi: 10.1109/TVCG.2016.2598415. URL http://ieeexplore.ieee.org/document/7534855/.

Ian MacKenzie, Chris Meyer, and Steve Noble. How retailers can keep up with consumers. *mckinsey.com*, 2013. URL http://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers.

Michael Martin. As e-sports profitability booms, don't forget the players, 2014. URL http://www.statepress.com/article/2014/07/as-e-sports-profitability-booms-dont-forget-the-players/. [http://www.statepress.com/article/2014/07/as-e-sports-profitability-booms-dont-forget-the-pl accessed in 13 de Novembro de 2015].

Tinnawat Nuangjumnong and Tinnawat. The Effects of Gameplay on Leadership Behaviors: An Empirical Study on Leadership Behaviors and Roles in Multiplayer Online Battle Arena Games. In *2014 International Conference on Cyberworlds*, pages 300–307. IEEE, oct 2014. ISBN 978-1-4799-4677-8. doi: 10.1109/CW.2014.48. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6980775.

Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to Recommender Systems Handbook. In *Recommender Systems Handbook*, pages 1–35. Springer US, Boston,

MA, 2011. doi: 10.1007/978-0-387-85820-3_1. URL http://link.springer.com/10.1007/978-0-387-85820-3{_}1.

Abhishek Saxena and Navneet K Gaur. Frequent Item Set Based Recommendation using Apriori. *International Journal of Science, Engineering and Technology Research*, 4(5), 2015. URL http://ijsetr.org/wp-content/uploads/2015/05/IJSETR-VOL-4-ISSUE-5-1609-1612.pdf.

Schloss Dagstuhl. Dagstuhl seminars, 2015. URL http://www.dagstuhl.de/en/program/dagstuhl-seminars/. [http://www.dagstuhl.de/en/program/dagstuhl-seminars accessed in 13 de Novembro de 2015].

Mirna Paula Silva, Victor Nascimento Silva, and Luiz Chaimowicz. Dynamic Difficulty Adjustment Through an Adaptive AI. In *Brazilian Symposium on Games and Entertainment (SBGames)*, 2015.

Mirna Paula Silva, Victor do Nascimento Silva, and Luiz Chaimowicz. Dynamic difficulty adjustment on MOBA games. *Entertainment Computing*, 18:103 – 123, 2017. ISSN 1875-9521.

Victor Nascimento Silva and Luiz Chaimowicz. A Tutor Agent for MOBA Games. *Brazilian Symposium on Games and Entertainment (SBGames)*, 2015a.

Victor Nascimento Silva and Luiz Chaimowicz. On the Development of Intelligent Agents for MOBA Games. *Brazilian Symposium on Games and Entertainment (SBGames)*, 2015b.

Steam. Web api (steamworks documentation), 2017. URL https://partner.steamgames.com/documentation/webapi. [https://partner.steamgames.com/documentation/webapi accessed on May 22nd, 2017].

Adrian Toma and Marius Tibeica. Dotapicker, 2015. URL https://www.dotapicker.com. [https://www.dotapicker.com accessed in 13 de Novembro de 2015].

Valve Corporation. Dota 2 - the international 2015, 2015a. URL https://www.dota2.com/international/announcement. [https://www.dota2.com/international/announcement accessed in 13 de Novembro de 2015].

Valve Corporation. Valve, 2015b. URL http://www.valvesoftware.com/. [http://www.valvesoftware.com/ accessed in 13 de Novembro de 2015].

Nick Wingfield. In E-Sports, Video Gamers Draw Real Crowds and Big Money - The New York Times, 2014. URL http://www.nytimes.com/2014/08/31/technology/esports-explosion-brings-opportunity-riches-for-video-gamers.html?_r=0.

Pu Yang, Brent E. Harrison, and David L. Roberts. Identifying patterns in combat that are predictive of success in MOBA games. In *Proceedings of the 9th International Conference on the Foundations of Digital Games, FDG 2014*, 2014. URL `http://www.fdg2014.org/papers/fdg2014_paper_36.pdf`.

Georgios Yannakakis and Julian Togelius. A Panorama of Artificial and Computational Intelligence in Games. *IEEE Transactions on Computational Intelligence and AI in Games*, pages 1–1, 2014. ISSN 1943-068X. doi: 10.1109/TCIAIG.2014.2339221. URL `http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6855367`.

Tao Zhu, Patrick Harrington, Junjun Li, and Lei Tang. Bundle recommendation in ecommerce. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval - SIGIR '14*, pages 657–666, New York, New York, USA, 2014. ACM Press. ISBN 9781450322577. doi: 10.1145/2600428.2609603. URL `http://dl.acm.org/citation.cfm?doid=2600428.2609603`.

Apêndice A

Lista de Siglas

IA Inteligência Artificial

IC Inteligência Computacional

API *Application Programming Interface*

HOTS *Heroes of the Storm*

LOL *League of Legends*

NPC *Non-Player Character*

MOBA *Multiplayer Online Battle Arena*

RTS *Real Time Strategy*