UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE INFORMÁTICA

GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Predição de *links* em redes sociais utilizando *Extreme Learning Machines*

TRABALHO DE GRADUAÇÃO

Aluno: Hugo Neiva de Melo (hnm2@cin.ufpe.br)

Orientador: Ricardo Bastos C. Prudêncio (rbcp@cin.ufpe.br)

Avaliadora: Flávia de Almeida Barros (fab@cin.ufpe.br)

Recife, 23 de setembro de 2013

Resumo

Atualmente, com o crescimento dos estudos na área de inteligência artificial e com a grande expansão de redes sociais, torna-se muito importante que essas redes sejam analisadas sob diversos aspectos. Existem vários tipos de problemas que podem ser levantados nesse sentido, entre eles, o problema de predição de *links* dentro de uma rede social. Por ser um problema difícil de modelar através de programas e linguagens computacionais comuns, estão sendo utilizadas atualmente redes neurais tentar oferecer soluções melhores. Este trabalho de graduação apresenta um estudo sobre um algoritmo específico de redes neurais, chamado *Extreme Learning Machine*, e apresenta uma metodologia onde ele foi aplicado a fim de tentar resolver o problema de predição de *links*. Esta monografia mostra todos os passos da implementação, desde a aquisição de dados, passando pela análise de várias métricas, até a experimentação. É apresentada também uma análise da qualidade da rede neural como solução do problema, mostrando pontos fracos e fortes nos experimentos executados.

Palavras-chave: Predição de links, Análise de redes sociais, Redes neurais, Extreme Learning Machine, Redes de coautoria.

Agradecimentos

Agradeço primeiramente a Deus, que me propicia uma vida ótima e cheia de vitórias. Dedico este trabalho aos meus pais, Ubirajara Filho e Betânia Melo, e ao meu irmão, André Neiva, pois sem eles não seria nada perto do que sou hoje. As dificuldades enfrentadas apenas serviram como exercício de fortalecimento para a vida. Agradeço profundamente, também, a meus avós, Ubirajara Melo, Maria Carmen, Lindomar Santos e Severino Santos, por aguentar minhas chatices e pelo apoio que me deram por todos esses anos de minha vida, pois mesmo apesar da idade, nunca desanimaram. Agradeço, em especial, a minha falecida tia Vânia Mesquita, minha segunda mãe e sempre terá um lugar guardado no meu coração. Agradeço aos meus tios Sávio, Évora, Júnior, Jô e aos meus padrinhos, tio Humberto e tia Tereza, que sempre estiveram presentes e me ajudaram muito, mesmo sem me contar. Gostaria de agradecer também aos meus melhores amigos, Arthur Ramos, amigo de longas datas, Davi Duarte, David Hulak, Dalton Santana e Tullio José, que juntos comigo, nesses quase cinco anos de universidade, superamos inúmeras dificuldades encontradas durante o curso de graduação e também fora dele, partilhamos vários bons momentos e aturaram meus piores momentos, sejam de raiva ou tristeza. São amizades assim, valiosas e com pessoas de bom coração, que são levadas pelo resto da vida. Agradeço também aos meus primos, Rebeca e Almerinho, que sempre estiveram ao meu lado e também me apoiam todo o tempo. Não sei como agradecer a Enoque Barros, Simone Azevedo, Jéssica Azevedo e Juliane Melo, que me proporcionaram condições para que eu realizasse um sonho de infância e passasse por uma incrível experiência de vida da qual nunca vou esquecer, além da ótima companhia que eles sempre foram. Agradeço, por fim, ao meu orientador, Ricardo Prudêncio, pelas cadeiras em que lecionou, com qualidade, e por me aceitar e guiar na confecção desse trabalho de graduação, tão importante para a minha formação.

Muito obrigado.

Sumário

Índice de Figuras

Índice de Tabelas

1. Introdução

Nos últimos anos, o uso e o número de redes sociais dos mais diversos tipos emergiu entre os usuários da internet. Uma rede desse tipo é composta de usuários, e cada um está ligado a outros usuários através de algum tipo de relação (amizades, colaborações acadêmicas, entre outros tipos). Uma rede, desse modo, pode ser formalmente representada por um grafo, onde os usuários são vértices e suas relações dentro da rede são representadas por arestas.

Um aspecto importante está no fato de que dentro de uma rede social o grafo não mantém a mesma configuração, ou seja, o número de vértices e arestas, bem como os *links* existentes entre usuários estão sempre mudando, de tal forma que predizer o surgimento de novos *links*, baseado em padrões encontrados entre os nós e arestas existentes, se torna importante para analisar tanto a evolução da rede quanto os mecanismos que propiciam meios para essa evolução [1].

Uma abordagem utilizada para a predição do surgimento de novas arestas se dá através do uso de Redes Neurais Artificiais. Essas redes mostram-se bastante úteis para essa tarefa por serem capazes de aprender padrões, se adaptar e "adivinhar" novos que surgirem, com uma boa taxa de acerto. O principal problema que ocorre na predição de novas arestas está na dificuldade de encontrar padrões de comportamento dentro da rede social que ajudem a definir que indivíduos dentro dela irão se relacionar no futuro. Atualmente são utilizadas diversas medidas, que se baseiam em características comuns observadas entre os indivíduos da rede e que podem influenciar direta ou indiretamente no comportamento desses indivíduos. Redes neurais são úteis para a resolução desse tipo de problema, pois conseguem encontrar, separar e aprender os padrões dessas medidas de modo confiável, além de conseguirem identificar os mesmos padrões encontrados ao serem aplicados em instâncias diferentes do problema ou em problemas semelhantes.

Atualmente existe uma área da computação, chamada de Análise de Redes Sociais (SNA - *Social Network Analysis*) que estuda o comportamento dos usuários dentro dessas redes e é importante em vários aspectos, como economia (análise de compras), segurança (redes terroristas), entre outros, pois seus resultados permitem realizar análises estratégicas baseadas em informações extraídas das redes [2].

1.1. Objetivo

O presente trabalho propõe desenvolver e analisar o desempenho de uma rede *feedforward*, utilizando o algoritmo *Extreme Learning Machine* [3]. Essa rede deve ser capaz de, perante a análise de uma série de medidas extraídas de uma rede social qualquer, predizer o surgimento de novas arestas entre nós, dentro de alguns tipos de redes sociais, como redes colaborativas. Uma vantagem de se utilizar esse algoritmo é que o tempo gasto

com a aprendizagem da rede neural chega a ser milhares de vezes menor [4], se comparado a algoritmos mais tradicionais, como o *Backpropagation*.

Serão utilizadas como entradas da rede neural alguns tipos de medidas extraídas dessas redes sociais, como o número de nós vizinhos em comum (CN - *Number of Common Neighbors*), que acabam por caracterizar essas redes sociais como grafos com pesos nas arestas (*Weighted Networks*) [2].

O final do projeto inclui a realização de diversos experimentos de modo a avaliar e escolher a rede que melhor se saiu na tarefa de predizer o surgimentos de novos *links*, além de discutir os pontos fortes e fracos da rede encontrados durante tais experimentos.

1.2. Estrutura do documento

Neste trabalho, o capítulo 2 abrangerá um breve histórico sobre o surgimento das redes sociais e uma modelagem para sua representação utilizando a teoria dos grafos. Por fim, será discutido melhor o problema de predição de *links*. O capítulo 3 apresentará um breve histórico sobre o contexto do surgimento das primeiras redes neurais, do algoritmo *Extreme Learning Machine* e diversos conceitos básicos relacionados à área e outros mais específicos relacionados ao algoritmo, como topologias de rede, parâmetros de uma rede neural, aprendizagem, entre outros.

O capítulo 4 discute as várias medidas de entrada, extraídas diretamente do grafo em análise e também todo o processo de treinamento e testes da rede neural para a obtenção da melhor configuração. O capítulo 5 mostrará os resultados obtidos na avaliação de desempenho da rede neural implementada, e por fim, o capítulo 6 expõe uma conclusão sobre o trabalho desenvolvido e apresenta idéias para trabalhos futuros.

2. Análise de redes sociais

Uma rede social é uma estrutura formada por um conjunto de indivíduos ou organizações. Elas existem desde quando o ser humano começou a viver em sociedade e sempre se caracterizou pela constante mudança em sua estrutura interna. Com o aparecimento e popularização da internet, um tipo novo (e um pouco diferente) de rede social surgiu: a rede social online. Existem diversas redes desse tipo, cada uma reunindo um grupo de pessoas com interesses em comum, como compartilhar conhecimento, expor a vida social e fazer amizades, desenvolver *softwares* livres, redes internas de empresas privadas, entre outros. Redes online se popularizaram muito nos últimos anos, dentre as quais pode-se citar: canais do mIRC, *Classmates*, *deviantART*, *MySpace* e mais novas, como *GitHub*, *Orkut*, *Facebook*, *Twitter* e diversas outras. Outro exemplo bastante interessante que se pode observar são redes colaborativas, formadas por acadêmicos que colaboram uns com os outros na produção de artigos científicos ou no desenvolvimento de pesquisas.

Hoje em dia, várias dessas redes estão relacionadas a um grande mercado, por envolverem uma grande quantidade de pessoas. Também, pelo mesmo motivo, são alvo de estudos por diversas áreas do conhecimento, entre elas a computação. *Social Network Analysis* – SNA (Análise de redes sociais) é o ramo que estuda o surgimento, características e evolução de redes sociais em geral e atualmente possui diversas pesquisas sobre o assunto, concluidas ou em andamento.

A seção 2.1 deste capítulo mostra conceitos importantes sobre teoria dos grafos que são úteis para ajudar a modelar problemas de predição de *links* em redes sociais, enquanto a seção 2.2 explica mais detalhadamente o problema abordado.

2.1. Teoria dos grafos

Um grafo é uma estrutura abstrata cujas primeiras noções originaram-se de um dos trabalhos de Leonhard Euler, em 1736, e que representa um conjunto de objetos onde alguns deles estão ligados entre si. Cada objeto é representado no grafo como um nó ou vértice, enquanto uma ligação é representada por uma aresta. Formalmente, um grafo é definido como um conjunto V não vazio de vértices e um conjunto A de pares não ordenados (para o caso de grafos não direcionados) de vértices, que representam as arestas. Esses dois conjuntos formam a estrutura G(V,A).

Uma rede social pode ser representada por meio de um grafo, onde cada indivíduo que pertence à rede é representado como um nó, e os relacionamentos entre indivíduos representados como arestas entre os respectivos nós. A aresta pode ou não ser direcionada, se os relacionamentos forem recíprocos (por exemplo, a relação de amizade) ou não (relação de citação em artigos científicos).

No projeto, a representação do grafo se dá como uma lista de pares de nós, onde cada par representa uma aresta não direcionada (relacionamento recíproco), acompanhado do número total de nós da rede, a fim de identificar nós isolados. A figura 3.1 mostra um esquema de como é organizada essa representação.

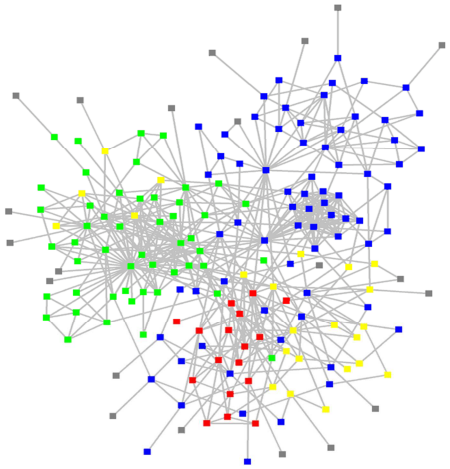

Figura 2.1: Exemplo de esquematização de entidades de uma rede social em forma de grafo (fonte: *http://www.infovis.net/printMag.php?num=136&lang=1*).

Apesar da representação formal do grafo ser de fácil entendimento e simples de se esquematizar, existem algumas representações mais adequadas para o processamento por um programa de computador, que tornam mais fáceis a extração de dados e a análise automática. Dentre as representações existentes, a seguir são mostradas as mais utilizadas:

- **Matriz de adjacência**: representação matricial de um grafo e simples de ser lida, essa estrutura é uma matriz A, quadrada (e simétrica, caso o grafo seja não direcionado), cuja posição $A_{i,j}$, onde $0 \leq i, j \leq n$ e n representa a ordem da matriz e a quantidade de vértices do grafo. Essa matriz possui a informação de ligação entre os vértices i e j, normalmente representada por um *bit* ou um peso (caso as arestas possuam valores), indicando se a ligação existe ou não. Caso o grafo seja direcionado, a posição $A_{i,j}$ indica se existe uma aresta de i que incide em j.

- **Lista de adjacência**: é uma lista de listas cujos itens são identificados pelos nós do grafo. Quando uma aresta existe entre os vértices i e j, a lista indexada por i contém j como um nó adjacente e assim por diante. A vantagem de se usar essa representação é quando o que mais interessa no grafo são as arestas presentes e pode ser bastante útil para poupar processamento durante sua leitura, quando o grafo possui poucas arestas.

Dentro de grafos que representam redes sociais, algumas características interessantes podem ser notadas. Por exemplo, existem alguns nós (*hubs*) mais centrais, com grau alto se comparados aos seus vizinhos. Também pode-se observar a formação de

clusters, ou seja, agrupamentos de nós bastante interligados [6]. Outro tipo de nó caracteristico dessas redes são os que funcionam como pontes, não possuindo um alto grau, se comparado aos *hubs* ou aos seus vizinhos, mas que interligam *clusters*. Uma ponte pode ser formada por um ou mais nós.

Outra característica importante e que precisa ser levada em consideração é o tempo de criação das arestas. Visto que a rede muda de estrutura em instantes diferentes, não se pode considerar simplesmente a presença ou não de arestas, mas deve-se também considerar quando essas arestas surgiram, a fim de extrair as informações necessárias para predizer, num instante de tempo posterior, quais arestas irão ou não surgir. Isso influencia a construção dos conjuntos de treinamento, validação e testes para o problema de predição de *links*, discutidos no próximo capítulo.

2.2. Predição de *links*

A predição de *links* é um problema dentre vários na área de análise de redes sociais, que consiste em basicamente duas perspectivas. A primeira é sob o aspecto de conexões faltantes: o problema é detectar conexões implícitas dentro da rede e utilizá-las para predizer futuros relacionamentos. Essa abordagem do problema é a escolhida para ser tratada no projeto. A segunda perspectiva é sobre temporalidade da predição, ou seja, o modo como ocorre o crescimento da rede ao longo do tempo é levado em consideração para a predição de conexões que ainda não existem e que podem ser formadas num futuro próximo. Essa vertente do problema considera a rede como uma estrutura evolutiva, procurando identificar como a rede estará estruturada no futuro.

Existem basicamente dois tipos de entradas que alimentam esse problema: características dos membros da rede e informações estruturais e topológicas sobre a rede. Essas informações servem para o cálculo de métricas, baseadas em preceitos sociológicos, que de alguma forma conseguem extrair e representar padrões, de forma mensurável, de modo a servir de base para valorar o grau de similaridade e proximidade entre dois indivíduos, ou seja, pares de nós dentro da rede. De posse dessas métricas, algum método de aprendizado supervisionado ou não supervisionado pode ser aplicado na tarefa final de predizer futuros relacionamentos. Existem vários métodos que funcionam bem para a predição de relacionamentos. Esses métodos costumam seguir uma de três abordagens: a baseada em similaridade entre nós, baseada em padrões estruturais ou baseada em modelos probabilísticos.

Soluções para esse tipo de problema costumam ser aplicadas em diversas áreas, como por exemplo, predição de colaborações em artigos (área abordada nesse trabalho), onde tenta-se predizer que autores irão colaborar entre si na produção de artigos científicos, ou na estruturação de organizações criminosas, pois essas se utilizam das conexões implícitas, comentadas anteriormente, para preservar a confidencialidade dos contatos entre indivíduos de tais redes. Outros exemplos de aplicações incluem sistemas de recomendação, onde um *link* consiste numa relação entre objeto e usuário (essa relação pode ser de compra, por exemplo), como o sistema da *Amazon*, e análise de propagação de doenças, onde o

relacionamento entre indivíduos se dá através do contágio, ou seja, as relações entre indivíduos indicam quem infectou quem.

3. Redes neurais

Uma rede neural artificial é uma abordagem para a resolução de problemas baseada no funcionamento do sistema nervoso humano. Assim como no cérebro, uma rede neural artificial possui neurônios e a transmissão de dados é feita através de sinais. Redes desse tipo são muito utilizadas em aplicações de inteligência artificial.

Neste capítulo, a seção 3.1 apresenta uma série de conceitos básicos importantes para o entendimento das etapas do projeto, principalmente a parte de experimentação. A seção 3.2 explica, em linhas gerais, como surgiu e como funciona o algoritmo *Extreme Learning Machine*, e por fim a seção 3.3 mostra as medidas utilizadas para avaliar o desempenho da rede neural.

3.1. Conceitos básicos

- **Neurônio:**

É a unidade básica de processamento da rede neural. Neurônios são tanto interligados entre si como possuem conexões com o mundo externo. As ligações entre neurônios e com o exterior são feitas através de conexões com pesos associados, responsáveis por armazenar o conhecimento da rede. A modelagem mais conhecida de um neurônio foi proposta por Warren McCulloch e Walter Pitts (neurônio MCP) em 1943 e é o modelo mais utilizado atualmente.

- **Camadas da rede:**

Uma rede *feedforward* é divida basicamente em 3 camadas: a camada de entrada, a camada escondida e a camada de saída. A camada de entrada está conectada com o ambiente externo e é responsável pela aquisição dos dados de entrada. A camada escondida (ou intermediária) pode ser composta por uma ou mais camadas de neurônios. Diz-se escondida pois ela não possui conexões com o mundo externo. Por fim, a camada de saída é a responsável pelas últimas computações e por fornecer a resposta da rede para a entrada fornecida.

- **Topologia de rede:**

A topologia de uma rede inclui a distribuição dos neurônios bem como a forma de propagação do conhecimento ao longo da rede. Existem atualmente três tipos de topologias, sendo elas *feedforward*, recorrente e construtiva. O algoritmo *Extreme Learning Machine* foi feito para redes com topologia *feedforward*, onde o conhecimento, originalmente, é propagado no sentido positivo (da camada de entrada para a camada de saída) e o erro propagado no sentido contrário. O algoritmo não só encontra a solução com menor erro associado, como também a solução com menor norma dos pesos das conexões, fazendo da rede uma melhor generalização para o problema em questão [5].

- **Parâmetros de rede:**

Alguns valores da rede utilizada, chamados de parâmetros, precisam ser escolhidos e variados durante a experimentação a fim de identificar a melhor configuração de rede para resolver o problema abordado. Dentre os diversos parâmetros, vale citar:

a. Número de neurônios na camada escondida: representa a quantidade de neurônios que pertencerão à camada intermediária da rede. Esse parâmetro será variado durante as experimentações para encontrar a melhor configuração de rede para o problema;
b. Função de ativação: A função de ativação é um parâmetro de rede importante, pois define, em cada neurônio, a regra de propagação do sinal. Foram utilizadas para os testes cinco funções de ativação, mostradas a seguir:

Função Sigmoide:

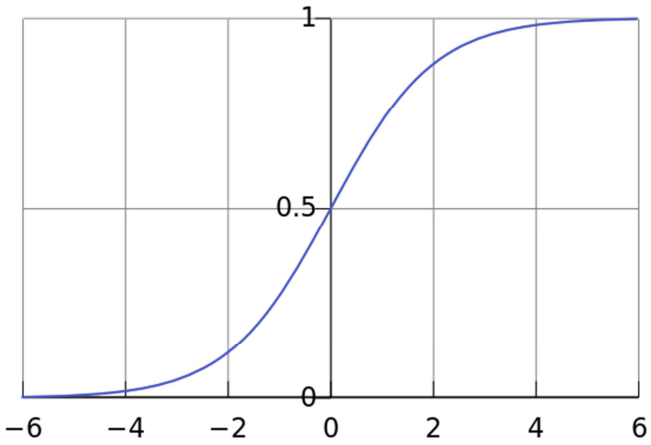

Figura 3.1: Exemplo de gráfico de função sigmoide ($S(x) = 1 / (1 + e^{-a*x})$).

Função Seno:

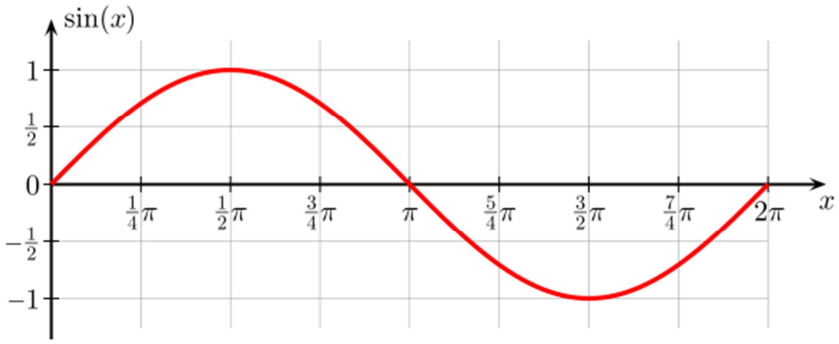

Figura 3.2: Exemplo de gráfico da função seno (S(x) = sen(x)).

Função limiar (*Hard-limit transfer function*):

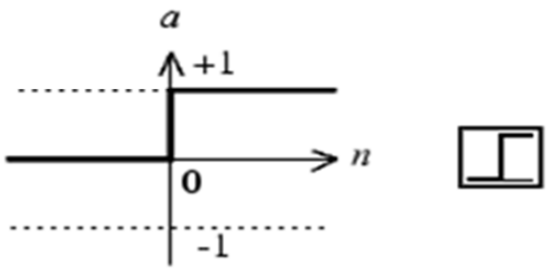

Figura 3.3: Gráfico da função limiar.

Função de base triangular (*Triangular basis transfer function*):

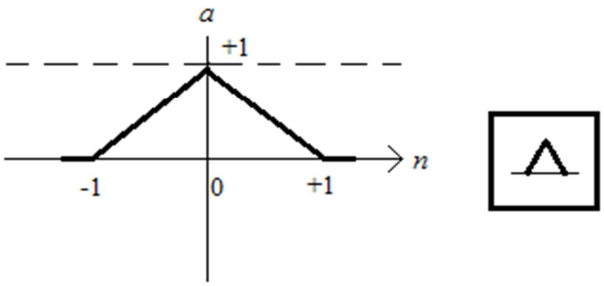

Figura 3.4: Gráfico de uma função de base triangular.

Funções de base radial (*Radial basis function* - RBF): são as mesmas funções utilizadas em redes neurais do tipo RBF.

- **Classes dos exemplos:**

Os exemplos utilizados para alimentar a rede podem ser divididos em várias classes as quais a rede irá aprender a distinguir (para os problemas de classificação). Cada classe é um grupo de exemplos que possuem o mesmo padrão (ou valor) de saída. Por exemplo, para o problema em questão, os exemplos podem ser divididos em duas classes: a classe dos pares de nós em que uma aresta surgirá (novo relacionamento entre dois indivíduos), dentro do grafo e a classe dos pares onde não surgirão arestas. As classes dos exemplos precisam ser previamente identificadas e separadas antes dos conjuntos de treinamento, validação e teste serem criados.

- **Conjuntos de treinamento, validação e teste:**

Os conjuntos de treinamento, validação e teste são os conjuntos de dados utilizados como entradas da rede neural. O conjunto de treinamento é utilizado na etapa de ajustes dos pesos da rede, onde ela aprende a classificar os padrões de entrada. O conjunto de validação serve para verificar (paralelamente) se o treinamento da rede obteve resultados satisfatórios (devendo-se treinar novamente a rede, caso seja necessário), e por fim, o conjunto de testes serve para simular um caso de aplicação real da rede e verificar se a solução encontrada atende ao problema alvo.

Como foi falado anteriormente, o tempo é uma variável que precisa ser levada em consideração ao montar os três conjuntos. Ele influencia na divisão das entradas e consequentemente na construção dos conjuntos. Os dados são divididos primeiramente por período, por exemplo, por ano, onde uma entrada é equivalente ao conjunto de métricas extraídas do grafo em um determinado período, sobre dois vértices i e j, acompanhado de um *bit* informando se uma aresta foi observada posteriormente (outro período distinto). Após a separação das entradas por períodos, elas podem ser escolhidas aleatoriamente para compor cada conjunto.

3.2. *Extreme Learning Machines*

Redes neurais são atualmente muito utilizadas e desde as suas primeiras implementações se mostraram como um método revolucionário para a resolução de diversos problemas complexos e difíceis de representar através de linguagens computacionais

comuns. Uma rede neural passa por um processo de aprendizagem (treinamento), que normalmente é bastante custoso. Como exemplo bem conhecido e bastante utilizado, pode-se citar uma rede neural que implementa o algoritmo de *Backpropagation*: esse algoritmo possui alguns problemas, como mínimos locais decorridos da utilização do gradiente descendente na implementação, que fazem com que o tempo necessário para que a rede seja treinada corretamente seja geralmente grande, pois atrasam o treinamento.

De posse desse problema, Guang-Bin Huang, Qin-Yu Zhu e Chee-Kheong Siew [5] propuseram recentemente um novo algoritmo de treinamento para redes *feedforward*, uma solução mais direta chamada *Extreme Learning Machine*, onde alguns problemas que causavam a lentidão no treinamento, como mínimos locais decorrentes da utilização do gradiente descendente, foram eliminados sem perder a capacidade de resolver corretamente os mesmos problemas, tornando muito mais rápido o treinamento da rede. Atualmente vários acadêmicos estão utilizando essa nova alternativa, pela grande vantagem mostrada quando comparada aos algoritmos tradicionais de redes neurais do mesmo tipo.

A rede neural implementada utilizando o algoritmo proposto [5] possui a camada de entrada com pesos aleatoriamente inicializados e a camada escondida com apenas uma camada de neurônios. O número de neurônios dessa camada é um parâmetro de rede já discutido.

3.3. Medidas de desempenho

Existem diversas métricas que avaliam de formas diferentes como a rede neural se saiu na tarefa de classificar os padrões de entrada. Essas métricas são utilizadas durante as fases de treinamento e validação, onde os parâmetros de rede são alterados. Quando todas as configurações de rede são testadas e seus desempenhos avaliados, são selecionadas as melhores configurações (ou apenas a melhor), baseadas nos valores obtidos através do cálculo dessas métricas. Algumas, bastante utilizadas, são explicadas a seguir:

a. Taxa de acerto (*Classification Accuracy*): é uma medida de desempenho que relaciona a quantidade de erros (*e*) com o total de entradas da rede (*t*). O valor obtido é a porcentagem de acertos:

$$CA = 1 - (e/t) \quad (1)$$

b. Matriz de confusão (*Confusion matrix*): A matriz de confusão armazena quatro tipos de resultados da rede: verdadeiros positivos, falsos positivos, verdadeiros negativos e falsos negativos. A diagonal principal da matriz contém o número de padrões classificados corretamente, enquanto os valores da diagonal secundária representam os padrões classificados erroneamente. A figura 3.6 mostra um exemplo genérico de matriz de confusão, utilizada para analisar um problema de classificação binário.

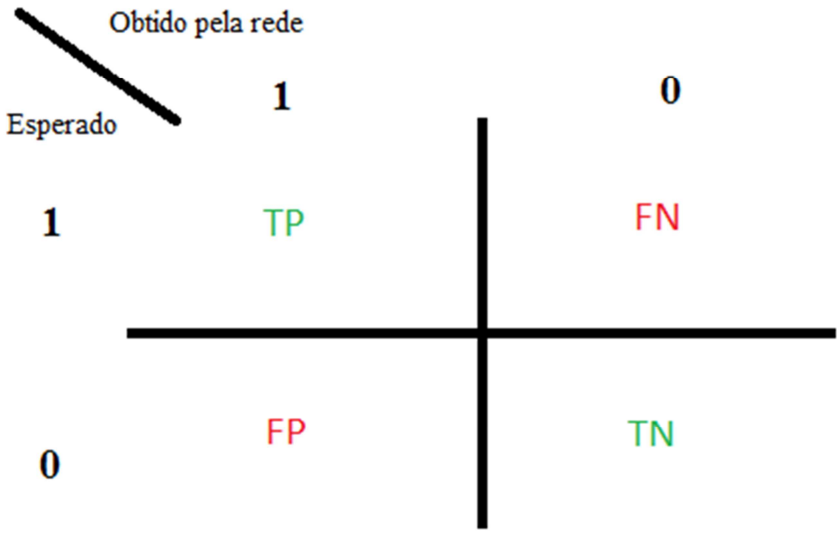

Figura 3.5: Exemplo genérico de matriz de confusão.

Na matriz acima, o valor A contabiliza o número de padrões de entrada positivos que foram classificados corretamente (1 ou 0, verdadeiro ou falso, acima ou abaixo de um limiar, etc.), o valor B contabiliza o número de resultados negativos classificados corretamente, o valor C representa o número de falsos negativos e o D o número de falsos positivos. Claramente, da matriz acima pode-se encontrar a taxa de acerto da rede:

$$CA = (TP + TN)/(TP + TN + FP + FN) \quad (2)$$

Outra medida útil é a soma dos elementos da diagonal principal ($TP + TN$), que representa o número total exato de padrões classificados corretamente.

c. Cobertura e Precisão (*Recall* e *Precision*): essas duas métricas são muito utilizadas em problemas de recuperação de informação, principalmente para avaliar resultados de consultas em recuperação de documentos. A precisão representa o número de documentos corretos que foram retornados por uma consulta. Já a cobertura mostra, do total de documentos classificados como resultados corretos para determinada consulta, quantos foram retornados. Da matriz de confusão anterior, podemos obter os valores relacionados a essas duas medidas:

$$Precision = TP/(TP + FP) \quad (3)$$

$$Recall = TP/(TP + FN) \quad (4)$$

d. *F-measure* (ou F_1-*Score*): é a média harmônica entre a precisão e a cobertura:

$$F = 2 \cdot (Precision \cdot Recall)/(Precision + Recall) \quad (5)$$

e. Curva ROC (*Receiver Operating Characteristic*): essa medida é representada por uma curva num espaço cartesiano bidimensional: o eixo das abscissas mostra as taxas de falsos positivos, ou seja, quantos resultados incorretos positivos ocorrem em todas as amostras negativas, enquanto o eixo das coordenadas representa a taxa de verdadeiros positivos, isto é, a quantidade de resultados positivos em relação ao total de amostras positivas. Por ser um gráfico, essa curva não é útil, em seu formato original, para servir como medida de desempenho. Para tanto, é calculada a área abaixo da curva ROC e quanto maior esse valor, melhor é o desempenho da rede com relação a essa medida. A figura 3.7 mostra um exemplo de curva ROC plotada utilizando o MATLAB.

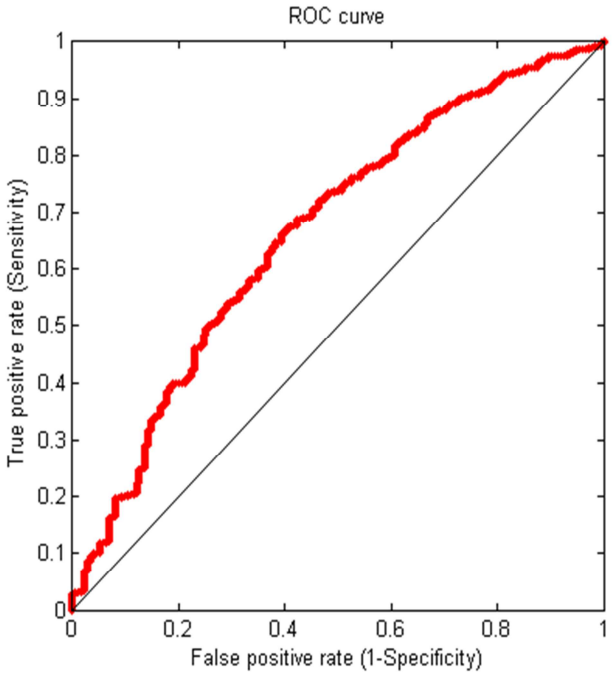

Figura 3.6: Exemplo de curva ROC plotada utilizando o MATLAB.

4. Desenvolvimento da rede neural utilizando *Extreme Learning Machine*

Com base nos conceitos apresentados no capítulo 3, foi desenvolvida uma rede neural capaz de predizer o surgimento de *links* entre indivíduos de uma rede social qualquer, seja ela online ou comum. O algoritmo *Extreme Learning Machine* utilizado foi o mesmo proposto por [5], disponível em [7]. O processo de desenvolvimento pode ser dividido em duas grandes fases: pré-processamento dos dados e experimentação. A fase de pré-processamento inclui desde a aquisição dos dados até a criação da matriz de adjacência, de onde são extraídas as medidas em breve discutidas. Já a fase de experimentação engloba a execução (treinamento, validação e testes) da rede, a análise de desempenho e a variação de parâmetros, a fim de encontrar a melhor configuração de rede que resolva o problema. A figura 4.1 apresenta o fluxo de execução e todas as fases da implementação da solução.

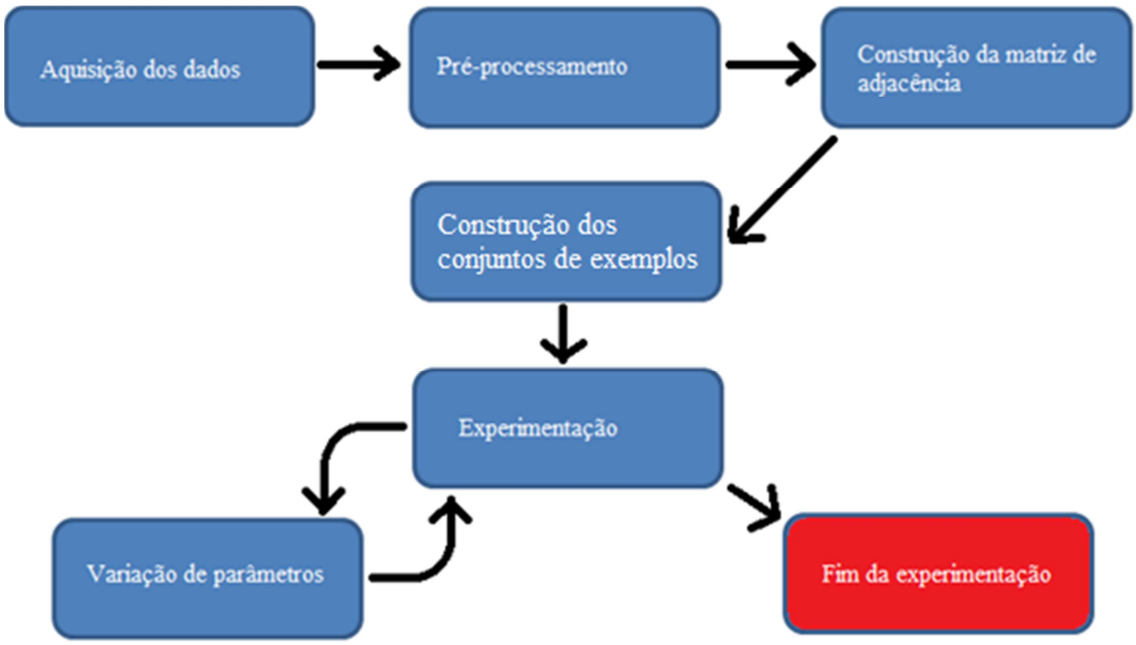

Figura 4.1: Fluxo de execução de todas as fases da implementação da solução.

A implementação das fases de extração de medidas e da experimentação da rede foram feitas utilizando a ferramenta MATLAB [8]. A ferramenta foi escolhida pois apresenta uma variada gama de recursos e *toolboxes* que ajudam na implementação e na obtenção de estatísticas sobre os resultados obtidos, além de ser bastante conhecida e utilizada. Já as fases de aquisição dos dados até a construção dos vetores de entrada foram implementadas utilizando a linguagem C# [9].

4.1. Aquisição dos dados

Passo inicial no processo de desenvolvimento do projeto, a aquisição de dados pode ser feita a partir de vários repositórios online, como em [10] e [11], que contenham informações sobre as redes sociais que se quer analisar. É importante que esses repositórios contenham dados sobre os períodos (meses, anos) de observação da rede, cuja importância já foi explicada anteriormente.

Existem muitos dados disponíveis dentro desses repositórios, porém apenas alguns são essenciais para o projeto. A rede em análise é uma rede colaborativa, e como tal, dados sobre autores, publicações, mês/ano dos artigos e coautorias são indispensáveis para a análise desses tipos de rede.

Para a experimentação, foi escolhido o repositório *quantum-ph* [12], o repositório de artigos sobre física quântica. O repositório encontrado em [11] provê uma API codificada em HTTP [13] para a extração de metadados relacionados aos artigos. Esses metadados contém todas as informações necessárias para o prosseguimento do projeto. Os dados extraídos do repositório *quantum-ph* [12] foram metadados sobre artigos publicados entre os anos de 2000 e 2007. A tabela 4.1 sumariza as principais informações sobre essa rede.

Característica	Valor
Período	2000 - 2007
Número de autores	16.988
Número de artigos	36.749
Proporção artigo/autor	2,1632
Grau médio por autor	5,094

Tabela 4.1: Informações sobre o repositório *quantum-ph*. Os autores considerados são aqueles que publicaram ou são coautores de algum artigo dentro do período referido.

4.2. Pré-processamento

Após a obtenção dos dados, é necessário criar a estrutura principal que será fonte de todas as medidas a serem extraídas: o grafo. Como já foi falado anteriormente, cada indivíduo pertencente à rede é representado por um nó e suas relações (no caso, a relação de coautoria) são vistas como arestas desse grafo.

Essa fase é executada através de uma série de passos, sendo eles:

1- Leitura das coautorias obtidas na fase anterior;

2- Identificação de todos os membros que são autores ou coautores de publicações;

3- Criação do conjunto V de nós do grafo, sendo um nó para cada autor encontrado;

4- Identificação das coautorias e criação do conjunto A de arestas;

5- Escrita do grafo em arquivo de texto, no formato de lista de adjacência.

4.3. Construção da matriz de adjacência

Do grafo criado é calculada a sua matriz de adjacência, que irá conter as informações de coautoria. Essa matriz é fonte para o cálculo de métricas que são utilizadas como entrada para a rede neural, que tenta inferir, a partir das medidas e da informação se o *link* surgiu ou não em um instante de tempo posterior, se dados dois vértices V_i e V_j do grafo, os quais não possuem relacionamento, a aresta E_{ij} será observada posteriormente.

4.4. Construção dos conjuntos de exemplos

No presente trabalho, cada exemplo de treinamento para a rede neural é associado a um par de nós da rede e é composto de: (1) atributos preditores, que são métricas de proximidade usadas para descrever o par de nós e (2) um atributo alvo, que indica a classe positiva (caso exista a conexão entre os nós) ou negativa (caso não exista a conexão).

Para construção dos exemplos de treinamento, validação e testes, os metadados obtidos foram divididos em dois períodos, de 2000 a 2005 e de 2006 a 2007. A divisão foi feita para que o período de 2006 a 2007 fosse utilizado pela rede como janela de tempo para a predição de futuros *links* (definição do atributo alvo), enquanto o período de 2000 a 2005 fosse utilizado como base para a extração das métricas discutidas (definição dos atributos preditores). A tabela 4.2 mostra os dados do repositório, por período.

Característica	**Período**	
	2000 - 2005	2006 - 2007
Número de autores	12.229	4.759
Número de artigos	14.944	21.805
Proporção artigo/autor	2,1632	1,2836
Grau médio por autor	4,438	5,094

Tabela 4.2: Informações sobre o repositório *quantum-ph*, por período. A proporção artigo/autor de 2006 a 2007 inclui os de 2000 a 2005, pois eles também participaram de artigos naquele período.

A matriz de adjacência é utilizada para calcular algumas métricas relacionadas aos nós da rede, baseadas em informações de distância e vizinhança de cada nó e utilizadas para

alimentar parte das entradas da rede neural. As medidas obtidas são baseadas em diversos preceitos da sociologia, alguns expostos por [14]:

- **Homofilia**: dois indivíduos que possuem mais características em comum um com o outro, ao invés de com outros indivíduos da sociedade, tem mais chances de estabelecer uma relação no futuro.

- **Raridade**: características comuns entre os membros de uma sociedade tendem a ser menos importantes e a ter menos influência no surgimento de um relacionamento entre dois indivíduos. Características mais peculiares, pelo contrário, tendem a interferir mais no surgimento de tais relacionamentos, por exemplo, dentro de uma comunidade de programadores, simpatizantes de uma mesma linguagem de programação tendem a criar mais relacionamentos entre si.

- **Relacionamentos em comum**: o número de relacionamentos em comum que dois membros de uma sociedade possuem torna maior a probabilidade de que tais pessoas venham a se conhecer no futuro. Os indivíduos que pertencem aos relacionamentos em comum intermediam informações que fazem com que os dois membros em questão possam vir a se conhecer.

- **Conexões preferenciais**: membros de uma sociedade tendem a criar relações com outros mais populares [15], ou seja, nós mais centrais dentro do grafo.

- **Influência social**: características de membros de uma sociedade que se relacionam com um certo indivíduo tendem a influenciar outro indivíduo qualquer, que possui as mesmas características, a se relacionar com o indivíduo específico. Por exemplo: se muitos amigos de um indivíduo A gostam de programar, é mais provável que outro indivíduo B, que também goste de programar, venha a se relacionar com A.

- **Proximidade social**: a proximidade entre dois nós (indivíduos) dentro de um grafo (distância entre os nós) tende a ser um fator relevante para o surgimento de uma aresta (relacionamento) entre eles. Uma característica interessante é a expressa na teoria do mundo pequeno, de onde foi constatado experimentalmente que a média de distância entre duas pessoas quaisquer no mundo, através da relação “amigo do amigo”, é de 6 graus (teoria dos seis graus de separação).

Por serem embasadas em uma estrutura idealizada na teoria dos grafos, as métricas extraídas são bastante confiáveis por possuirem um alicerce matemático que permite aplicá-las facilmente em diversas situações. As principais medidas utilizadas foram as seguintes:

- **Conexão preferencial (*Preferential Attachment* - PA)**: como dito anteriomente, a conexão preferencial representa o fato de que membros de uma sociedade tendem a criar relações com outros mais populares. De acordo com esse pensamento (*rico fica mais rico*) [17] [18], verificou-se que a probabilidade de um *link* entre dois nós

pode ser calculada e é proporcional ao produto do número de vizinhos (*N)* que cada nó possui.

$$PAij = N(i) \cdot N(j) \quad (6)$$

- **Vizinhos em comum (*Common neighbors* - CN)**: métrica mais simples que serve como base para a maioria das outras métricas calculadas e baseia-se na idéia de que dois vértices tem maior probabilidade de se conectarem se o número de vizinhos em comum também for maior [17]. Seja *C* o conjunto de vértices vizinhos a um vértice qualquer, então:

$$CNij = |C(i) \cap C(j)| \quad (7)$$

- **Grau de proximidade de Adamic-Adar (AA)** [16]: baseado no índice de Adamic e Adar e na métrica CN, essa medida representa o grau de proximidade entre dois vértices i e j, considerando que vizinhos em comum com menor número de conexões são mais relevantes.

$$AA_{ij} = \sum_{z \in \{C(i) \cap C(j)\}} 1/log(N(z)) \quad (8)$$

- **Coeficiente de Jaccard (*Jaccard's Coefficient* - JC)**: valor que mede a probabilidade de dois nós se conectarem através da razão entre o número de vizinhos em comum e o número total de vizinhos distintos de dois vértices i e j.

$$JCij = |C(i) \cap C(j)| / |C(i) U C(j)| \quad (9)$$

- **Coeficiente de Katz (*Katz*)** [19]: valor resultado da soma ponderada do tamanho de todos os caminhos entre dois nós, com maior peso dado aos caminhos mais curtos.

$$Katz_{ij} = \sum_{l=1}^{\infty} \beta^l \cdot |paths_{i,j}^{\langle l \rangle}| \quad (10)$$

Onde $paths_{i,j}^{\langle l \rangle}$ é o conjunto de todos os caminhos de tamanho *l* que tem i como vértice inicial e j como vértice final e $\beta \in (0,1)$ pondera a relevância dos caminhos de acordo com seus tamanhos. A forma matricial dessa métrica pode ser obtida através da seguinte fórmula:

$$Katz(A) = (1 - \beta \cdot A) - 1 - In \quad (11)$$

Onde A é a matriz de adjacência da rede e I_n é a matriz identidade de ordem n (mesma ordem de A).

- **Menor caminho (*Shortest path* - SP)** [20]: considera que quanto menor a distância entre dois vértices, maior a probabilidade de eles se conectarem. Como esses dois valores são inversamente proporcionais, o valor é o negativo da menor distância.

$$SPi\,j = -dist(i,j) \quad (12)$$

- ***Hitting Time* - HT**: é o número de passos esperado que será gasto, através de uma caminhada aleatória, ao sair de um nó i e chegar a outro nó j. Essa propriedade é característica de cadeias de Markov.

- ***Commute Time* - CT** [20]: é o número de passos esperado que será gasto, através de uma caminhada aleatória, ao sair de um nó i, chegar a outro nó j e voltar ao nó i inicial, ou seja, o *Commute Time* é a soma dos *Hitting Times* de ida, ou seja, de i a j, e de volta, j a i, cada um multiplicado pela probabilidade estacionária π do nó destino). Vale ressaltar que os *Hitting Times* de ida e de volta não são simétricos ($HT_{i\,j} \neq HT_{j\,i}$).

$$CTi\,j = -((HTi\,j \cdot \pi j) + (HTj\,i \cdot \pi i)) \quad (13)$$

A probabilidade estacionária de um vértice é um valor proporcional a quantidade de vezes o vértice é selecionado dentro de uma caminhada aleatória.

- ***Rooted PageRank* - RPR**: o algoritmo de *PageRank* [21] é muito utilizado para a recuperação de informação na internet. Sua proposta inicial foi indicar o quanto uma página *web* é importante na rede. O *Rooted PageRank* é uma variação para predição de *links*. Basicamente, é feita uma caminhada aleatória entre dois vértices i e j com uma probabilidade α de se retornar ao vértice anterior e probabilidade $1 - \alpha$ da caminhada continuar, selecionando o próximo nó (vizinho ao nó atual) da caminhada de forma aleatória.

$$RPRi\,j \;=\; (1 \;-\; \alpha) \cdot (1 - (\alpha \cdot T)) - 1 \qquad (14)$$

Onde $T = D^{-1} \cdot A$, D é uma matriz diagonal, onde $D_{i,i} = \sum_j A_{i,j}$ e A é a matriz de adjacência.

- ***SimRank*** [22]: valor resultado da medida de similaridade dos vizinhos de dois nós i e j: i e j têm maior probabilidade de se relacionarem se seus vizinhos possuirem características em comum. Os vizinhos de um vértice são todos os outros que podem ser alcançados a partir do vértice inicial.

$$SIMRANKi\,j \;=\; \gamma \cdot \left(\sum_{a \in C(i)} \sum_{b \in C(j)} SimRank(a,b)\right) / \,|C(i)| \cdot |C(j)| \qquad (15)$$

Onde $\gamma \in (0,1)$ e $SimRank_{i\,i} = 1$ (caso base).

Além das métricas calculadas, outra informação que deve ser adicionada aos vetores de entrada da rede neural, visto que a rede em questão utiliza-se de aprendizado supervisionado, é a resposta esperada da rede, ou seja, se o *link* entre os dois nós i e j (resposta esperada para cada entrada), caso não exista, foi observado posteriormente. Sem essa informação, não há como predizer o surgimento de novos *links*.

4.5. Experimentação

A experimentação consiste na principal etapa do projeto. Até o momento, a rede neural não foi utilizada e apenas os valores das entradas foram calculados e os vetores de entrada montados. Esta etapa é onde a rede neural será empregada, passando pelo treinamento e validação e posteriormente a fase de teste, sempre retornando ao treinamento e variando parâmetros até que todas as possibilidades de configurações de rede sejam esgotadas. A figura 4.2 detalha melhor a fase de experimentação do projeto.

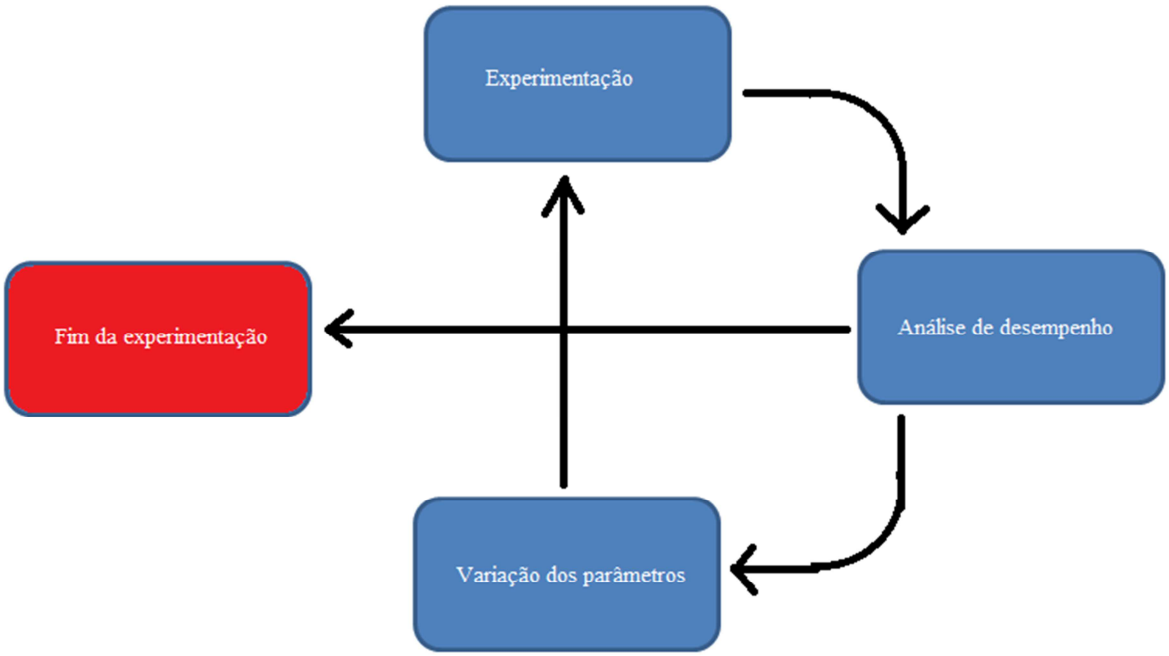

Figura 4.2: Fluxo de execução detalhado da fase de experimentação.

Foram feitos, no total, quatro experimentos, e para cada foram gerados 149.548.441 exemplos de entrada (possibilidades de combinações de autores, dados os 12.229 encontrados entre 2000 e 2005). Porém, como as matrizes das métricas são simétricas, apenas 74.740.970 de exemplos foram realmente utilizados. Esse valor exclui exemplos cujos vértices já se relacionavam (valor 1 na matriz de adjacência), ou seja, para a geração de exemplos foram considerados apenas pares de vértices que ainda não se relacionavam. A tabela 4.3 mostra algumas informações sobre os exemplos gerados.

Característica	**Valor**	
	Quantidade	%
Número de exemplos aproveitados	74.740.970	49,9778%
Número de exemplos excluídos	54.272	0,0362%
Número de exemplos negativos	74.738.008	99,9999%
Número de exemplos positivos	2.962	0,0001%
Proporção positivo/negativo	1/25.232	-

Tabela 4.3: Informações sobre os exemplos gerados. O número de exemplos positivos, negativos e a proporção positivo/negativo são baseados na quantidade de exemplos aproveitados.

Primeiramente, são escolhidos valores iniciais para os parâmetros da rede, que serão variados durante todo o processo, juntamente com os conjuntos de treinamento, validação e testes. O conjunto de vetores de entrada é dividido em duas classes, a classe positiva (um *link* é observado entre dois nós) e a classe negativa (um *link* não é observado). Após a

divisão, são escolhidos aleatoriamente 50% das entradas de cada classe para o treinamento, 25% para validação e 25% para o conjunto de testes. Após a separação dos três conjuntos, seus elementos são misturados entre si, de forma aleatória, de modo a aumentar a variabilidade dos conjuntos.

Após essa escolha inicial, a rede passa pelo treinamento, validação e testes até o fim da execução e posteriormente o desempenho da rede é calculado e seus valores, discutidos anteriormente, armazenados. De posse daqueles, os parâmetros são variados e novamente a rede passa pelo mesmo processo, até que todas as combinações de parâmetros tenham sido testadas. Por fim, a configuração que for melhor avaliada, baseado nas medidas de desempenho extraídas, é eleita como a melhor solução para o problema.

A análise de desempenho é feita da seguinte forma: primeiramente, cada possível configuração de rede é obtida pela variação dos parâmetros:

- Número de neurônios na camada escondida: variando de 05 a 30, com intervalos de 5 neurônios;

- Funções de ativação: Sigmoide, Seno, Limiar, Base Triangular e Base Radial (como apresentado na seção 3.1).

Para cada configuração de parâmetros, a rede é treinada, validada e testada uma vez e seu desempenho é armazenado. Após o esgotamento de todas as possíveis configurações, é feita a média de todas as medidas armazenadas, e então cada configuração, individualmente, é comparada com a média. A rede então é reinicializada e a melhor configuração escolhida através dessa comparação é utilizada para rodar a rede mais dez vezes, guardando o desempenho de cada iteração e computando as médias dos valores obtidos.

5. Resultados

Este capítulo apresenta os vários resultados obtidos durante os experimentos realizados, além dos critérios utilizados e do desempenho de cada rede neural.

5.1. Experimentos 1 e 2

O primeiro experimento foi feito utilizando 50.000.000 de exemplos, escolhidos aleatoriamente da base de exemplos inicial. Esse número foi escolhido devido à restrição de memória, problema encontrado nesses e em vários experimentos posteriores. Como existe um enorme desbalanceamento entre as classes, verificado anteriormente, foi garantido que todos os exemplos da classe positiva fossem escolhidos. Apenas os exemplos da classe negativa foram escolhidos aleatoriamente. A tabela 5.1 mostra os dados sobre o experimento.

Característica	**Valor**	
	Quantidade	%
Número total de exemplos	50.000.000	100%
Número de exemplos negativos	49.997.038	99,994%
Número de exemplos positivos	2.962	0,006%
Proporção positivo/negativo	1/16.880	-

Tabela 5.1: Dados sobre os exemplos utilizados pelo primeiro experimento.

Após a separação dos exemplos, a rede neural foi treinada e testada. Como será visto através das medidas de desempenho, a rede obteve uma taxa muito próxima ou igual a 100% de acerto, pois as classes estão desbalanceadas e a maioria dos exemplos é da classe negativa. Outra conclusão que pode ser deduzida é que mesmo com uma boa taxa de acerto, a área abaixo da curva ROC apresentou um valor muito próximo a 0,5, ou seja, é como se a rede neural decidisse aleatoriamente a qual classe pertence cada entrada. Além disso, a matriz de confusão mostra que a rede não aprendeu a distinguir os exemplos positivos, o que inutiliza a rede como bom classificador para o problema. A tabela 5.2 mostra a média dos dados sobre o experimento.

Característica	Valor médio	
	Treinamento/validação	Testes
Tempo de execução (em segundos)	75,235	8,026
Taxa de acerto (%)	0,9998	0,9998
Precisão (*Precision*)	0	0
Cobertura (*Recall*)	0	0
F_1 - *Score*	0	0
Área abaixo da curva ROC	0,5011	0,5

Tabela 5.2: Informações sobre os valores médios encontrados no primeiro experimento.

Na tentativa de diminuir a discrepância entre as classes, foi feito um segundo experimento, dessa vez utilizando 100.000 entradas, apenas. Novamente foi garantido que todos os exemplos da classe positiva fossem incluídos nas entradas, sendo apenas os vetores da classe negativa escolhidos aleatoriamente. A tabela 5.3 sumariza as informações sobre os exemplos do experimento.

Característica	Valor	
	Quantidade	%
Número total de exemplos	100.000	100%
Número de exemplos negativos	97.038	97,038%
Número de exemplos positivos	2.962	2,962%
Proporção positivo/negativo	1/33	-

Tabela 5.3: Dados sobre as entradas do segundo experimento.

Novamente o experimento não rendeu resultados satisfatórios. Da mesma maneira que no primeiro experimento, a rede obteve uma taxa de acerto boa, porém a área abaixo da curva ROC foi muito pequena e a matriz de confusão mostrou o mesmo problema: a rede neural não aprendeu a classificar bem exemplos da classe positiva, portanto não pode-se dizer que a rede resultante é um bom classificador para o problema. A tabela 5.4 mostra os resultados obtidos.

Característica	Valor médio	
	Treinamento/validação	Testes
Tempo de execução (em segundos)	0,4181	0,0452
Taxa de acerto (%)	90,174	90,168
Precisão (*Precision*)	0,036	0,036
Cobertura (*Recall*)	0,113	0,112
F_1 - *Score*	0,055	0,055
Área abaixo da curva ROC	0,5109	0,5114

Tabela 5.4: Valores médios encontrados no segundo experimento.

5.2. Experimento 3

Para o terceiro experimento, foram utilizados todos os exemplos da classe positiva, e a mesma quantidade de exemplos para a classe negativa, sempre escolhidos aleatoriamente. Os dados sobre as entradas são mostrados na tabela a seguir. Essa decisão foi tomada para que se tivesse os conjuntos de treinamento, validação e testes com as classes balanceadas. As informações sobre os exemplos gerados são mostrados na tabela 5.5.

Característica	**Valor**	
	Quantidade	%
Número total de exemplos	5.924	100%
Número de exemplos negativos	2.962	50%
Número de exemplos positivos	2.962	50%
Proporção positivo/negativo	1/1	-

Tabela 5.5: Dados sobre as entradas do terceiro experimento.

Dessa vez, com as classes balanceadas, a rede neural foi treinada corretamente e não ocorreu o problema encontrado nos experimentos anteriores. As tabelas 5.6 e 5.7 mostram a média dos resultados e os parâmetros e resultados da melhor configuração, respectivamente. As melhores configurações foram obtidas utilizando como função de ativação as funções sigmoide e seno, além de uma quantidade razoável de neurônios (25 e 30). As outras configurações obtiveram desempenho baixo, principalmente com relação à área abaixo da curva ROC, o que acabou por diminuir a média geral dos valores das métricas. A melhor configuração foi rodada mais dez vezes e foram computadas as médias dos resultados obtidos.

Característica	**Valor médio**	
	Treinamento/validação	Testes
Tempo de execução (em segundos)	0,0229	0,0114
Taxa de acerto (%)	64,01	63,07
Precisão (*Precision*)	0,672	0,663
Cobertura (*Recall*)	0,553	0,549
F_1 - *Score*	0,607	0,601
Área abaixo da curva ROC	0,6507	0,6405

Tabela 5.6: Valores médios encontrados no terceiro experimento.

Parâmetro	**Valor**	
Número de neurônios da camada escondida	25	
Função de ativação	Sigmoide (sig)	
Característica	**Valor médio**	
	Treinamento/validação	Testes
Tempo de execução (em segundos)	0,0125	0,0109
Taxa de acerto (%)	77,64	76,38
Precisão (*Precision*)	0,765	0,741
Cobertura (*Recall*)	0,881	0,807
F_1 - *Score*	0,819	0,773
Área abaixo da curva ROC	0,7764	0,7638

Tabela 5.7: Valores médios encontrados ao rodar dez vezes a melhor configuração.

5.3. Experimento 4

O último experimento foi feito utilizando a técnica de *bootstrap*, onde um subconjunto das entradas de cada classe são escolhidos aleatoriamente, preservando a proporção de padrões por classe (50% para cada classe). Para cada conjunto, foram escolhidos 1.000 exemplos representantes da classe positiva e 1.000 da classe negativa, totalizando 2.000 vetores de entrada por conjunto.

Para cada configuração de rede, foram gerados 100 conjuntos e extraídas as médias dos resultados obtidos. Novamente, a melhor configuração foi rodada mais dez vezes e os valores médios dos resultados, computados. Esses valores podem ser vistos nas tabelas 5.8 e 5.9.

Característica	**Valor médio**	
	Treinamento/validação	Testes
Tempo de execução (em segundos)	0,0082	0,0047
Taxa de acerto (%)	62,12	61,6
Precisão (*Precision*)	0,6434	0,5076
Cobertura (*Recall*)	0,5261	0,634
F_1 - *Score*	0,5027	0,5203
Área abaixo da curva ROC	0,6216	0,6169

Tabela 5.8: Valores médios encontrados no último experimento.

Parâmetro	**Valor**	
Número de neurônios da camada escondida	30	
Função de ativação	Sigmoide (sig)	
Característica	**Valor médio**	
	Treinamento/validação	Testes
Tempo de execução (em segundos)	0,0173	0,0106
Taxa de acerto (%)	74,56	74,12
Precisão (*Precision*)	0,7363	0,7324
Cobertura (*Recall*)	0,7651	0,7603
F_1 - *Score*	0,7505	0,7461
Área abaixo da curva ROC	0,7456	0,7412

Tabela 5.9: Valores médios encontrados ao rodar dez vezes a melhor configuração.

Durante este experimento, foi observado que algumas configurações tiveram um desempenho bem ruim (com a área abaixo da curva ROC perto de 0.5). Essas configurações utilizavam como função de ativação a de base triangular e a de base radial, além de uma quantidade pequena de neurônios na camada escondida (entre 5 e 20 neurônios). As melhores configurações, assim como no experimento anterior, utilizavam valores altos para o número de neurônios na camada escondida (25 e 30) e também utilizavam as funções sigmoide e seno como função de ativação. Como visto na tabela anterior, a melhor configuração utilizava 30 neurônios na camada escondida e a função sigmoide para ativação, resultando em uma área abaixo da curva ROC próxima a 0.75.

6. Conclusão

A técnica de redes neurais utilizando o algoritmo *extreme learning machine* se mostrou como uma saída bastante interessante para a solução do problema de predição de *links*. Como pôde ser observado, todo o processo empregado convergiu em uma rede capaz, com uma boa taxa de acerto, de predizer futuras relações entre indivíduos de uma rede social, em especial a abordada neste projeto.

Observou-se também a grande dificuldade que é tratar o desbalanceamento entre classes, visto que a maioria das entradas sempre representa a classe negativa. Foi necessária a redução da classe negativa, em vários experimentos sucessivos, para que a rede neural pudesse alcançar o desempenho desejado. O processo buscou aumentar a variabilidade das entradas para que o resultado fosse uma rede mais robusta. Esse foi observado no último experimento, onde a melhor configuração de rede se comportou praticamente da mesma maneira ao tentar predizer todos os possíveis *links* dentro de todos os conjuntos aleatórios gerados.

Além disso, os experimentos 3 e 4 convergiram em valores próximos quanto a melhor configuração de rede e ao desempenho. A função Sigmoide mostrou-se robusta para a rede de física quântica, pois os valores da taxa de acerto e da área abaixo da curva ROC se mostraram bastante compatíveis (e proporcionais), tanto durante o treinamento quanto após os testes, enquanto um valor entre 25 e 30 neurônios na camada escondida parece ser uma quantidade razoável e que torna a rede mais genérica como classificador para esse problema, nessa rede social específica.

Por fim, outro ponto a ser comentado é a dificuldade existente em tratar o grande volume de dados pertencentes a redes desse tipo, sejam elas colaborativas ou quaisquer outras redes sociais. Foi necessário implementar, durante todo o processo, métodos eficientes de utilização da memória para que toda a rede social pudesse ser processada sem que precisasse haver mudanças em sua estrutura ou necessidade da redução do período de amostragem e sem que fosse necessário uma grande quantidade de tempo gasto durante todas as etapas do projeto.

6.1. Trabalhos futuros

Com relação ao processo no todo, pretende-se investigar maneiras ainda mais eficientes de se utilizar a memória, que foi um gargalo em todas as etapas da execução. O computador utilizado para o projeto possui 8 *cores* e 8GB de memória. Outro ponto de melhoria é a eficiência dos algoritmos utilizados. Foram necessários vários procedimentos que tratavam texto ou que faziam operações com vetores grandes e matrizes de ordem muito alta, que junto com o uso total da memória RAM do computador, fez com que a velocidade de processamento caisse drasticamente.

Além de questões de eficiência, é necessário verificar melhor a importância de cada métrica utilizada e qual seu peso na hora de predizer a existência de um *link* dentro de uma rede social. Essa informação é importante para melhorar a predição, dando mais valor às métricas que se mostram mais determinantes para que um *link* surja, fazendo assim com que a qualidade dos resultados e a robustez da rede neural aumentem. Tentar também estudar métodos melhores (o método utilizado foi o de *undersampling*) que amenizem o problema de desbalanceamento entre classes, pois é algo que afeta muito o aprendizado da rede neural.

Por fim, também tentar descobrir outras fontes melhores de dados de redes sociais. A fonte utilizada tinha capacidade bastante limitada e a aquisição de todos os dados necessários tornou-se o primeiro gargalo do projeto.

Referências

[1] SÁ, Hially Rodrigues de; PRUDÊNCIO, Ricardo B. C.. Supervised link prediction in weighted networks. In: NEURAL NETWORKS (IJCNN), THE 2011 INTERNATIONAL JOINT CONFERENCE, 2011, San Jose, Ca. p. 2281 - 2288.

[2] WASSERMAN, S., FAUST, K., Social Network Analysis. Methods and Applications. Cambridge University Pres, 1994.

[3] NANYANG TECHNOLOGICAL UNIVERSITY. Extreme Learning Machines. Disponível em: <http://www.ntu.edu.sg/home/egbhuang/>. Acesso em: 26 jun. 2013.

[4] RAJESH, R.; PRAKASH, J. Siva. Extreme Learning Machines - A Review and State-of-the-art. International Journal Of Wisdom Based Computing, p. 35-49. abr. 2011.

[5] HUANG, Guang-Bin, ZHU, Qin-Yu, SIEW, Chee-Kheong, Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks, International Joint Conference on Neural Networks, Vol. 2, p. 985-990, 2004.

[6] FREEMAN, Linton C., Centrality in Social Networks: Conceptual Clarification., Lausanne, p.215-239, 1978.

[7] HUANG, Guang-bin et al. Basic ELM Algorithms. Disponível em: <http://www.ntu.edu.sg/home/egbhuang/elm_codes.html>. Acesso em: 24 jul. 2013.

[8] MATLAB, “The Language of Technical Computing”. Disponível em: <http://www.mathworks.com/products/matlab/>. Acesso em: 24 jul. 2013.

[9] MICROSOFT. “Visual C#”. Disponível em: <http://msdn.microsoft.com/en-us/library/vstudio/kx37x362.aspx>. Acesso em: 24 jul. 2013.

[10] LESKOVEC, Jure. Stanford University. Stanford Large Network Dataset Collection. Disponível em: <http://snap.stanford.edu/data/#canets>. Acesso em: 25 jul. 2013.

[11] CORNELL UNIVERSITY. “ArXiv.org e-Print archive”. Disponível em: <arXiv.org>. Acesso em: 25 jul. 2013.

[12] CORNELL UNIVERSITY. "Quantum Physics”. Disponível em: <http://arxiv.org/archive/quant-ph>. Acesso em: 25 jul. 2013.

[13] CORNELL UNIVERSITY. “Open Archives Initiative”. Disponível em: <http://arxiv.org/help/oa/index>. Acesso em: 25 jul. 2013.

[14] YIN, Z., GUPTA, M., WEINGER, T., HAN, J. A unified framework for link recommendation using random walks. Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining, ASONAM ’10, p.152–159, 2010.

[15] BARABÁSI, A. L. e ALBERT, R. Z.. Emergence of scaling in random networks. Science, 286, p. 509–512, 1999.

[16] ADAMIC, Lada; ADAR, Eytan. Friends and Neighbors on the Web. Social Networks, p. 211-230, 2001.

[17] NEWMAN, M. E. J. (2001a). Clustering and preferential attachment in growing networks. Physical Review E 64.

[18] BARABÁSI, A. L. et al. Evolution of the social network of scientific collaborations. *Physica A*, 311(3-4), p. 590–614, 2002.

[19] KATZ, L.. A new status index derived from sociometric analysis. Psychometrika. 18(1), p. 39–43, 1953.

[20] LIBEN-NOWELL, D., KLEINBERG, J.. The link prediction problem for social networks. Proceedings of the 12th International Conference on Information and Knowledge Management, CIKM ’03, p. 556–559, 2003.

[21] PAGE, Lawrence et al. “The PageRank Citation Ranking: Bringing Order to the Web”. Disponível em: <http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf>. Acesso em: 09 set. 2013.

[22] JEH, G., WIDOM, J.. Simrank: a measure of structural-context similarity. Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’02, p. 538–543, 2002.