Aplicação de Redes Neurais Artificiais no Balanceamento de Carga em Serviços de Nuvem

Karoline K. F. Setoue[1], **Kelton A. P. da Costa**[1]

[1]Departamento de Computação – Universidade Estadual Paulista (UNESP)
Faculdade de Ciências - Bauru, SP - Brasil

{karoline.setoue,kelton}@fc.unesp.br

Abstract. *The concept of cloud computing has become popular in recent years. Its goal is to provide services allow access to applications and files through the Internet in a flexible and simple way. In this context, exploring ways to efficiently secure this access becomes a task in Computer Science. Therefore, the present work presents a load balancing application prototype in a cloud computing architecture using Artificial Neural Networks as a technique for load distribution of requests. The application consists in a neural network applied to monitoring and adjustment of the weights for each server to which requests are directed.*

Resumo. *O conceito de computação em nuvem tem se tornado popular nos últimos anos. Seu objetivo é prover serviços de acesso à aplicações e arquivos por meio da internet de maneira flexível e simples. Neste contexto, explorar maneiras de garantir este acesso de maneira eficiente torna-se uma tarefa importante na área de computação. Tendo isto em vista, o presente trabalho apresenta um protótipo de aplicação de balanceamento de cargas em uma arquitetura de computação em nuvem utilizando Redes Neurais Artificiais como técnica para distribuição das cargas de requisições. A construção da aplicação consiste na incorporação da rede neural no monitoramento e ajuste dos pesos em cada servidor do sistema em nuvem para os quais as requisições serão direcionadas.*

1. Introdução

A Computação em Nuvem propõe uma arquitetura diferente da cliente-servidor, na qual os recursos são diversificados e encontram-se dispersos em servidores ao redor do mundo. A disponibilidade e o acesso a estes recursos e transparente ao usuário final é adaptável de acordo com a demanda. Neste contexto, grande parte da preocupação se concentra em atribuir as tarefas para os nos da nuvem de forma eficiente. Isto é feito para que o esforço e processamento de requisições seja feito da melhor maneira possível [Taleb-Bendiab 2010].

Como todas as abordagens e estruturas de serviços em nuvem precisam garantir a disponibilidade do serviço nela hospedado, há a necessidade de distribuir as requisições que chegam para os serviços de maneira eficiente, visto que a distribuição de tarefas e requisições feita de forma eficiente leva a melhora no tempo de resposta, economia de energia e diminui o risco de falhas.

Com base nesse requisito, o balanceamento de carga tem por objetivo receber as requisições e distribuí-las para os nós do serviço de nuvem. Isto pode ser feito de

diversas maneiras e a classificação dos algoritmos que realizam o balanceamento de carga é dividida, geralmente, em dois tipos: algoritmos estáticos e algoritmos dinâmicos [K. A. Nuaimi 2012].

Tendo isto em vista, técnicas de reconhecimento de padrões também podem ser viáveis neste contexto. Seu objetivo é aprender uma função que melhor separe amostras de classes diferentes no espaço de características [Duda et al. 2000]. Esse processo de aprendizado e reconhecimento pode ser feito por meio de Redes Neurais Artificiais (RNA) - *Artificial Neural Networks (ANN)* [Haykin 2007], cuja arquitetura baseia-se na estrutura de neurônios, que por meio de redes, são capazes de realizar as mais complexas atividades.

O presente trabalho propõe uma aplicação de balanceamento de cargas em sistemas de nuvem que utiliza uma RNA como técnica de distribuição de pesos entre os servidores configurados em uma nuvem, comparando seu comportamento com o algoritmo Round Robin, que apresenta como principal vantagem a estabilidade e a simplicidade de implementação e execução [NGINX 2017].

2. Arquitetura do Protótipo

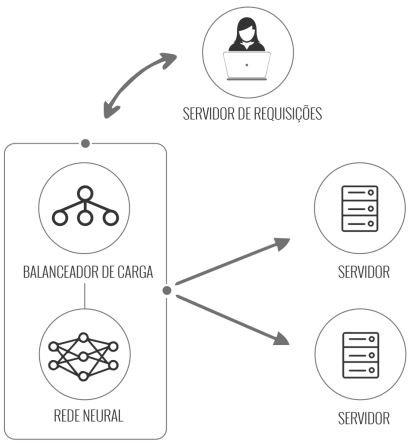

Figure 1. Arquitetura do protótipo

A arquitetura implementada para o módulo de balanceamento segue a proposta apresentada na Figura 1, dividindo-se em módulos: Balanceador de Carga, Rede Neural, Servidor de Requisições e Servidores, sendo os dois primeiros parte integrante do protótipo de balanceamento de carga e os dois últimos utilizados para teste do protótipo.

O módulo que contempla o balanceador de cargas e a rede neural é responsável por distribuir as requisições geradas pelo servidor de requisições. A partir das características de cada servidor, a rede neural atribui dinamicamente pesos a cada um dos servidores presentes na lista de servidores ativos. Então, o algoritmo de balanceamento de cargas envia a requisição ao servidor com maior peso. Este processo se repete a cada requisição recebida.

Considerando que a RNA escolhida seria incorporada ao módulo de balanceamento e que, por ser uma aplicação de balanceamento de carga de requisições - onde o

tempo de resposta é importantíssimo - o modelo de rede neural escolhido é de simples implementação.

Implementação da Rede Neural

A rede contem duas camadas: uma de entrada e uma camada de saída. Por se tratar de uma rede neural aplicada à balanceamento de carga, a forma de aprendizado ocorre de maneira dinâmica, considerando as mudanças de estado dos servidores.

A Figura 2 apresenta a arquitetura da rede, na qual a camada de entrada recebe as características em um vetor x com n características, w representa a matriz de pesos, com n linhas (número de entradas) e m colunas (quantidade de saídas). Na camada de saída, os valores propagados da primeira camada passam por uma sigmoide que tem como saída y_i o valor de peso que é atribuído ao vetor de pesos utilizado pelo algoritmo de balanceamento de carga para distribuir as requisições.

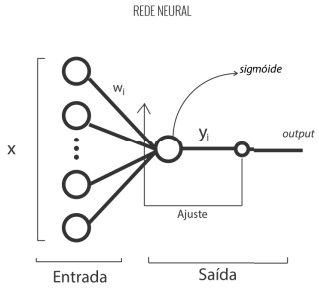

Figure 2. Rede Neural implementada

Tendo isto em vista, a rede neural escolhida possui poucas camadas, como ilustrado pela Figura 2, com o objetivo de aumentar a velocidade de processamento e garantir que o tempo de resposta do balanceador não seja tão afetado pelo tempo de processamento da RNA. Ao fim do processo de distribuição das requisições, os pesos de w são ajustados.

3. Resultados

Os experimentos realizados para verificar o desempenho do balanceador foram realizados em ambiente real, de maneira dinâmica através da execução dos três módulos da aplicação. Foram alocados servidores na *Amazon EC2* [Services 2016] sendo três servidores do tipo t2.micro como servidores, um servidor t2.medium para alocar a aplicação de balanceamento e um computador para executar a aplicação para executar as requisições, no qual é armazenado o log das requisições.

A avaliação do desempenho da aplicação de balanceamento foi comparada ao balanceamento realizado com Round Robin puro, através da análise dos arquivos de log gerados em ambos os casos de teste, com o mesmo número de requisições com o qual foi testada a aplicação de RNA. É fato que a aplicação de balanceamento teve desempenho satisfatório, visto que conseguiu distribuir a carga das requisições entre os servidores com sucesso, sem apresentar uma diferença significativa em relação ao algoritmo de Round Robin puro.

Durante o processo foi possível notar que a característica que mais influenciou o processo de escolha foi a quantidade de requisições associadas ao servidor. Percebe-se

Table 1. Comparação entre distribuição de requisições entre servidores

Servidores	Round Robin	RNA	RNA - treinada
Servidor 1	34	36	37
Servidor 2	35	35	37
Servidor 3	34	36	32

Elaborado pela autora.

também que servidores do mesmo tipo tendem a deixar o balanceamento mais equilibrado. A Tabela 1 apresenta a distribuição das requisições atribuídas a cada um dos servidores alocados após a execução dos testes.

4. Conclusão

Os testes não demonstram ganho significativo no uso da RNA. Portanto, nas condições em que os testes foram realizados, pode-se considerar o Round Robin como melhor alternativa, se levarmos em conta a simplicidade de sua implementação. Considerando o ambiente em que foram realizados os testes, que pode ser considerado estável e com servidores do mesmo tipo, o algoritmo de balanceamento com RNA não se destacou. Porém, foi possível perceber que a característica que mais influenciou a classificação foi o tempo de resposta dos servidores.

O modelo proposto neste trabalho leva em consideração os estados mais recentes dos servidores, levando em conta seu último estado, o que o torna mais dinâmico se comparado com o Round Robin. Esta implementação pode apresentar melhor desempenho em ambientes mais instáveis, visto que a distribuição das cargas se dá pela avaliação de estados como falhas de conexão e estado do servidor, por exemplo e seria interessante simular ambientes deste tipo em trabalhos futuros, para avaliar o desempenho da abordagem proposta.

References

Duda, R. O., Hart, P. E., and Stork, D. G. (2000). *Pattern Classification (2nd Edition)*. Wiley-Interscience.

Haykin, S. (2007). *Neural Networks: A Comprehensive Foundation (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

K. A. Nuaimi, N. Mohamed, M. A. N. J. A.-J. (2012). A survey of load balancing in cloud computing: Challenges and algorithms. *Second Symposium on Network Cloud Computing and Applications*, pages 137 – 142.

NGINX (2017). What is round-robin load balancing?

Services, A. W. (2016). Ec2.

Taleb-Bendiab, M. R. . D. L. . A. (2010). A comparative study into distributed load balancing algorithms for cloud computing. *Advanced Information Networking and Applications Workshops (WAINA)*, pages 551 – 556.