Rede Neural Artifical

Gustavo dos Santos Vieira Lemes
Centro Universitário Alves Faria
Marcos Paulo da Costa Mendes
Centro Universitário Alves Faria

***Resumo* — Este artigo tem como objetivo relatar experiências ao treinar uma Rede Neural Artificial (RNA) para a classificação de cogumelos como comestíveis ou venenosos.**

I. INTRODUÇÃO

Rede Neural Artificial (RNA) pode ser definida como uma estrutura complexa interligada por elementos de processamento simples (neurônios), que possuem a capacidade de realizar operações para processamento de dados e representação de conhecimento. Uma RNA é inspirada no sistema nervoso central e seu primeiro conceito foi introduzido em 1943, ganhando bastante popularidade algumas décadas depois com a introdução de algoritmos de treinamento como o *backpropagation*, que permite a realização de um treinamento posterior para aperfeiçoar os resultados do modelo. O artigo em questão, pretende exibir a implementação de uma RNA capaz de determinar se cogumelos são comestíveis ou venenosos, bem como o desempenho e os resultados obtidos através de diferentes testes.

II. MATERIAL E MÉTODOS

A base de dados utilizada foi retirada da *UCM Machine Learning Repository*, do doador Jeff Schlimmer. Os dados foram retirados de *The Audubon Society Field Guide to North American Mushrooms (1981)* (Guia de Campo da Sociedade Audubon para Cogumelos da América do Norte).

O conjunto de dados inclui descrições de 8124 amostras hipotéticas, sendo 4208 classificadas como comestíveis e 3916 classificadas como venenosas, correspondentes a 23 espécies de cogumelos cultivados na família Agaricus e Lepiota. Cada espécie é identificada como definitivamente comestível, definitivamente venenosa ou de comestibilidade desconhecida e não é recomendada. Está última classe foi combinada com a venenosa. O guia afirma claramente que não existe uma regra simples para determinar a comestibilidade de um cogumelo. Os dados de entrada utilizados foram: forma do chapéu (*cap shape*), superfície do chapéu (*cap surface*), cor do chapéu (*cap color*), contusões (*bruises*), odor (odor), brânquia anexada (*gill attached*), espaçamento entre as brânquias (*gill spacing*), tamanho da brânquia (*gill size*), cor da brânquia (*gill color*), forma do caule (*stalk shape*), raiz do caule (*stalk root*), superfície do caule acima do anel (*stalk surface above ring*), superfície do caule abaixo do anel (*stalk surface below ring*), cor do caule acima do anel (*stalk color above ring*), cor do caule abaixo do anel (*stalk color below ring*), tipo de véu (*veil type*), cor do véu (veil color), número de anel (*ring number*), tipo de anel (*ring type*), cor do esporo impresso (*spore print color*), população (*population*) e *habitat*.

Utilizou-se uma RNA do tipo Perceptron de multicamadas para o desenvolvimento do projeto, bem como o framework Keras, que utiliza algoritmo *de backpropagation*, juntamente com Tensorflow, usado para ser o backend. A função de ativação adota foi a sigmoide e o modelo possui três camadas, sendo duas ocultas e uma de saída. Nas camadas ocultas há seis neurônios e na camada de saída apenas um. Para o treinamento do modelo, 75% das amostras estão sendo utilizada e os outras 25% estão sendo utilizadas para testes.

Para que haja um maior conhecimento no desempenho do modelo, utiliza-se uma matriz de confusão.

		Valores Previstos	
		Positivo	**Negativo**
Valores Reais	**Positivo**	**TP**	**FP**
	Negativo	**FN**	**TN**

Figura 1 – Matriz de confusão.

TP (True Positive) – Representa os valores positivos que o modelo determinou como positivo.
FP (False Positive) – Representa os valores positivos que o modelo determinou como negativo.
FN (False Negative) – Representa os valores negativos que o modelo determinou como positivo.
TN (True Negative) – Representa os valores negativos que o modelo determinou como negativo.

Baseado nos dados que se pode retirar da matriz de confusão, pode-se determinar algumas informações importante para melhor compreensão do modelo feito:

Acurácia – É a taxa dos valores testados que modelo conseguiu acertar.

$$\frac{TP + TN}{TP + TN + FP + FN}$$

Figura 2 – Fórmula para calcular a acurácia do modelo.

Recall – É a taxa dos valores positivos que o modelo acertou no total de positivos das amostras testadas.

$$\frac{TP}{TP + FN}$$

Figura 3 – Fórmula para calcular o recall do modelo.

Precisão – É a taxa dos valores positivos que o modelo acertou no total de positivos que ele indicou.

$$\frac{TP}{TP + FP}$$

Figura 4 – Fórmula para calcular a precisão do modelo.

F-Score – Média harmônica do Recall e da Precisão, quanto maior, maior a confiabilidade da acurácia.

$$2 * \frac{\text{Precisão} * \text{Recall}}{\text{Precisão} + \text{Recall}}$$

Figura 5 – Fórmula para calcular o F-score do modelo.

III. RESULTADOS E DISCUSSÃO

Utilizando-se 80 épocas para treinar a RNA, pode-se obter diferentes resultados para cada vez que se executava o que fora implementado. Abaixo há resultados de diferentes execuções da implementação, utilizando-se os mesmos dados para treinamento e teste de desempenho do modelo.

```
loss: 0.0277 - acc: 0.9719

             Comestivel(Predito)  Envenenado(Predito)
Comestivel                  1326                  465
Envenado                       0                  240

Acuracia: 77.10 %
Recall: 100.00 %
Precisao: 74.04 %
FScore: 85.08 %
```

Figura 6 – Resultados de teste (a) da RNA.

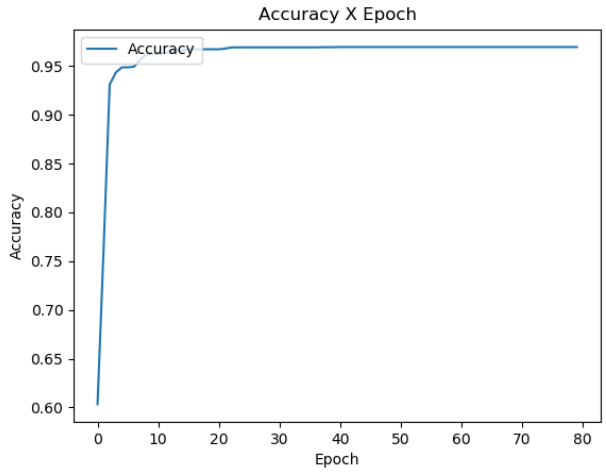

Figura 7 – Gráfico da acurácia x época do modelo referente ao teste (a).

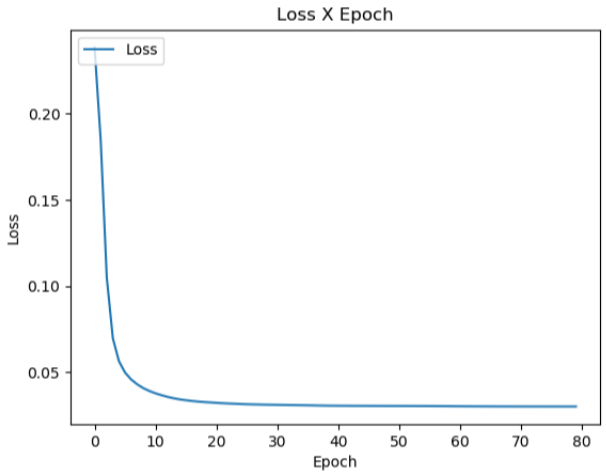

Figura 8 – Gráfico da perda x época do modelo referente ao teste (a).

```
loss: 0.0218 - acc: 0.9778

             Comestivel(Predito)  Envenenado(Predito)
Comestivel                  1695                   96
Envenado                       0                  240

Acuracia: 95.27 %
Recall: 100.00 %
Precisao: 94.64 %
FScore: 97.25 %
```

Figura 9 – Resultados de teste (b) da RNA.

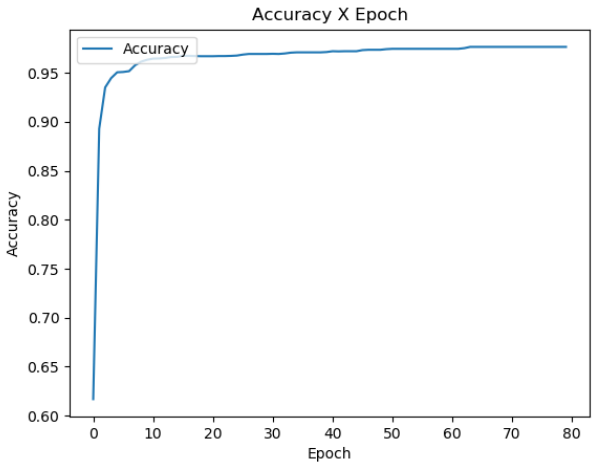

Figura 10 – Gráfico da acurácia x época do modelo referente ao teste (b).

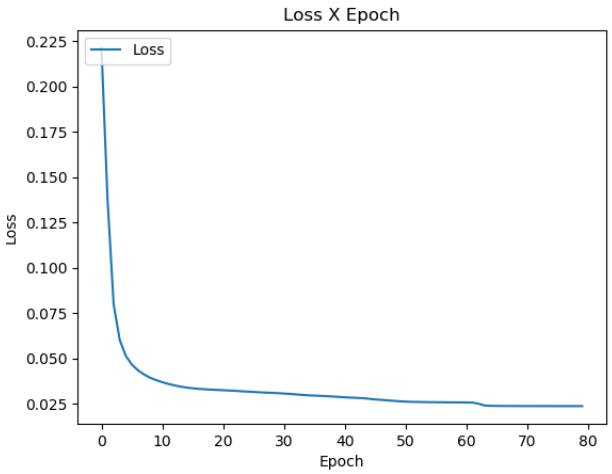

Figura 11 – Gráfico da perda x época do modelo referente ao teste (b).

Analisando os gráficos de ambos os testes, não é possível notar muita diferença em relação ao aprendizado da RNA, em relação a sua acurácia e perda. Porém, quando se verifica a matriz de confusão e alguns parâmetros que se pode retirar dela, percebe-se que a RNA interpreta os dados de maneira diferente a cada vez que se executa. No teste 'a', notasse que a acurácia determinada pela matriz foi cerca de 18% menor que no teste 'b' e isso implicou num aumento no número de falso positivo no modelo e uma queda de cerca de 20% na precisão do mesmo. Outro fator importante de se notar, é o F-score de ambos os resultados, há uma diferença considerável entre eles e o do resultado 'b' alcançou quase os 100%, indicando, assim, que pode-se

confiar mais em seus resultados.

IV. CONCLUSÃO

Uma RNA apesar de ser bastante complexa pode ser bastante otimizada de acordo com a complexidade do problema a ser resolvido. Inicialmente começamos com uma camada bem densa com muitos neurônios e isso só acarretou em problemas, com realização de vários testes percebemos que não seria necessários muitos neurônios para o nosso problema mas a variação da resposta estava bastante grande. Resolvemos adicionar outra camada oculta e notamos que a variação dos resultados diminuiu bastante. Terminamos o projeto configurando o de forma a ter 6 neurônios em cada camada oculta e 1 na camada de saída.

Com os testes realizados, podemos notar a eficácia de nossa RNA sendo segura ao classificar um cogumelo como comestível pois diante dos testes não fora classificado nenhum falso negativo.

REFERÊNCIA

[1] SANTANA, Rodrigo. **Análise de Sentimentos – Aprenda de uma vez por todas como funciona utilizando dados do Twitter**.15 mar. 2017. Disponível em: http://minerandodados.com.br/index.php/2017/03/15/analise-de-sentimentos-twitter-como-fazer/#avaliando-modelo. Acesso em: 13 maio 2019.

[2] GORGENS, Eric Bastos *et al*. **Estimação do volume de árvores utilizando redes neurais artificiais**. 2009. Disponível em: https://www.redalyc.org/html/488/48815855016/. Acesso em: 13 maio 2019.