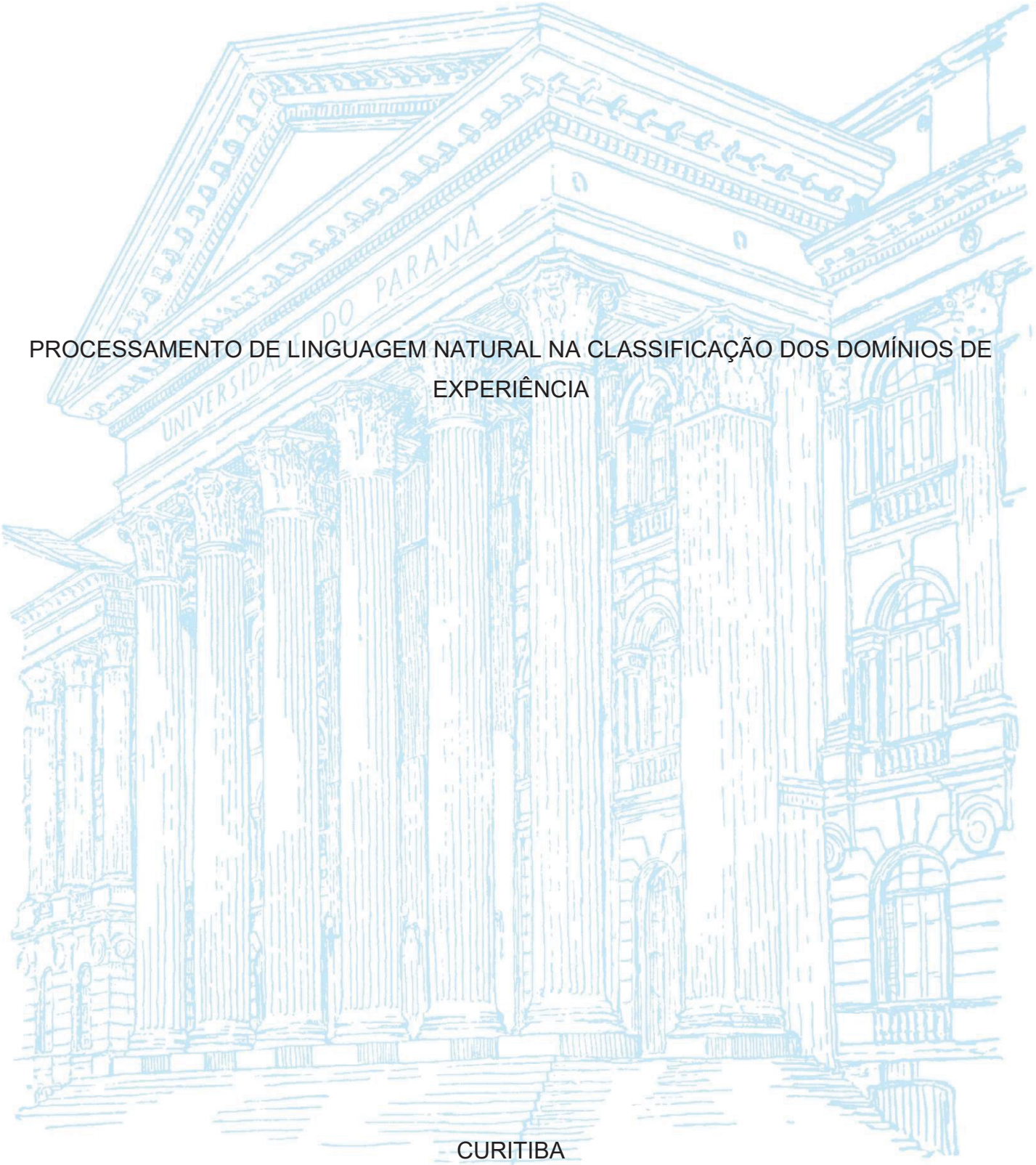
UNIVERSIDADE FEDERAL DO PARANÁ

JOHANN BELENDA MASCHIO



PROCESSAMENTO DE LINGUAGEM NATURAL NA CLASSIFICAÇÃO DOS DOMÍNIOS DE EXPERIÊNCIA

CURITIBA

2022

JOHANN BELENDA MASCHIO

PROCESSAMENTO DE LINGUAGEM NATURAL NA CLASSIFICAÇÃO DOS DOMÍNIOS DE EXPERIÊNCIA

Trabalho de conclusão de curso apresentado ao curso de Pós-Graduação em Inteligência Artificial Aplicada, Setor de Educação Profissional e Tecnológica, Universidade Federal do Paraná, como requisito parcial à obtenção do título de Especialista em Inteligência Artificial.

Orientador: Prof. Dr. João Eugenio Marynowski

CURITIBA

2022

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação INTELIGÊNCIA ARTIFICIAL APLICADA da Universidade Federal do Paraná foram convocados para realizar a arguição da Monografia de Especialização de **JOHANN BELENDA MASCHIO** intitulada: **Processamento de Linguagem Natural na Classificação dos Domínios de Experiência**, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de especialista está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 23 de Novembro de 2022.

JOÃO EUGENIO MARYNOWSKI
Presidente da Banca Examinadora

RAFAELA MANTOVANI FONTANA
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

Processamento de Linguagem Natural na Classificação dos Domínios de Experiência

Johann Belenda Maschio
SEPT - Setor de Educação Profissional e Tecnológica
UFPR - Universidade Federal do Paraná
Curitiba, Brasil
jbmaschio@gmail.com

João Eugenio Marynowski
SEPT - Setor de Educação Profissional e Tecnológica
UFPR - Universidade Federal do Paraná
Curitiba, Brasil
jeugenio@ufpr.br

***Resumo*—Utilizando Processamento de Linguagem Natural esse trabalho apresenta a comparação dos modelos *Random Forest*, *Support Vector Machine*, *Stochastic Gradient Descent*, *Multi-layer Perceptron* e Redes Neurais Artificiais, na classificação de quadrigramas nos quatro domínios da experiência de um turista, a partir dos comentários realizados no site *Tripadvisor* sobre seis atrativos do Parque Estadual do Jalapão. Os dois melhores desempenhos foram da Rede Neural Artificial com 77% de acurácia, utilizando *Tokenizer* do *Keras* e o *Stochastic Gradient Descent* com 76% de acurácia utilizando TF-IDF (*Term Frequency–Inverse Document Frequency*).**

***Palavras-chave*—Parque estadual do Jalapão, RNA, SVM, SGD, MLP, random forest**

Abstract*—*Using Natural Language Processing this work presents a comparison of the models Random Forest, Support Vector Machine, Stochastic Gradient Descent, Multi-layer Perceptron and Artificial Neural Networks, in the classification of quadrigrams in the four domains of a tourist's experience, based on the comments made on the website Tripadvisor about six attractions in the Jalapão State Park. The two best performances were the Artificial Neural Network with 77% accuracy, using Tokenizer from Keras and the Stochastic Gradient Descent with 76% accuracy using TF-IDF (Term Frequency–Inverse Document Frequency).

***Index Terms*—Jalapão State Park, ANN, SVM, SGD, MLP, random forest**

I. DESENVOLVIMENTO

Um texto é a linguagem natural dos seres humanos. A linguagem natural é a forma como nós seres humanos nos comunicamos, conforme a região temos nossos idiomas assim como os computadores têm suas linguagens de programação. É nesse contexto que nasce o processamento de linguagem natural.

Processamento de Linguagem Natural (PLN), é um campo da inteligência artificial interessado em interpretar a linguagem humana para os computadores. Um dos objetivos da PLN é reconhecer textos, identificar a intenção das palavras usadas, contexto e entender o texto da mesma forma que um humano faria, para gerar informação e utilizar em sistemas ou aplicações. Dada a grande dificuldade e das variadas formas que humanos podem entender um mesmo texto, PLN usa técnicas de Aprendizado de Máquina para entregar o significado [1].

Com o intuito de classificar a experiência de um turista, foi criado por Pine e Gilmore [2], um modelo chamado de estágios da estruturação de uma experiência, consistindo em dois eixos, um vertical e outro horizontal, conforme pode ser observado na Figura 1. O eixo horizontal representa a participação ativa (*active participation*) e passiva (*passive participation*), já o eixo vertical representa a absorção (*absorption*) e a imersão (*immersion*) do turista.

O cruzamento dos eixos gera os quatro domínios da experiência ou quatro dimensões: entretenimento (*entertainment*), educação (*educational*), estética (*esthetic*) e evasão (*escapist*).

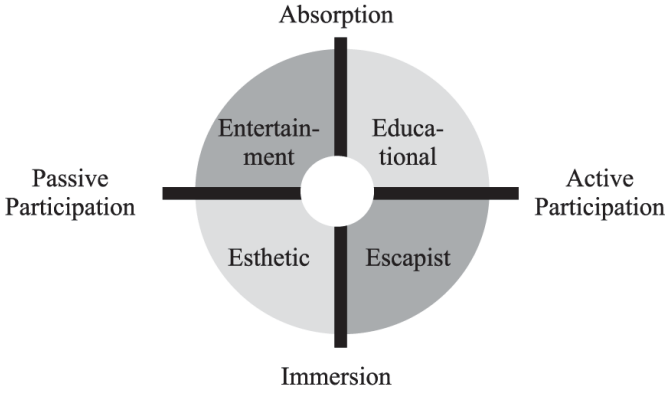


Figura 1. Eixos da estruturação da experiência [2].

Com o objetivo de classificar os comentários [3] realizaram a contagem de quatro palavras que aparecem em sequência (quadrigrama) e quantas vezes esse mesmo quadrigrama aparece no conjunto inteiro. Os quadrigramas são resultado da análise de frequência dos conjuntos de quatro palavras mais comuns dentro dos comentários.

A escolha de utilizar quadrigramas foi baseada na agilidade e na facilidade do tratamento dos dados, sendo assim facilitando também a utilização neste trabalho.

Este artigo tem como objetivo a utilização de um modelo de Inteligência Artificial (IA) para a classificação automatizada de quadrigramas dentro dos quatro domínios da experiência de um turista propostos por [2]: entretenimento, educação/aprendizagem, evasão/fuga e estética/contemplação.

A. Descrição dos dados

Os dados utilizados são 1257 quadrigramas classificados manualmente dentro dos quatro domínios da experiência, gerados a partir de 2.104 comentários feitos no site *Tripadvisor*[1] sobre seis atrativos do Parque Estadual do Jalapão (PEJ) [4], localizado no estado do Tocantins.

B. Métodos

Esta seção descreve como os métodos aplicados no tratamento e análise dos dados coletados. Os dados irão passar por cinco etapas: 1) composição da base de dados; 2) pré-processamento dos dados;

[1]https://www.tripadvisor.com.br/Attractions-g2441392-Activities-Jalapao_State_Park_State_of_Tocantins.html

3) preparação do treinamento; 4) treinamento e 5) predição. Por fim o estudo gera um modelo de classificação, baseado em IA, que classifica quadrigramas nos quatro domínios da experiência.

1) Composição da base de dados: Os quadrigramas coletados e classificados foram unificados, pois estavam separados por atrativo. Existiam as colunas "Domínio", "Quadrigramas", "Frequência"e "Palavras", sendo utilizadas neste trabalho as colunas Quadrigramas e Domínio. A Tabela I mostra alguns exemplos de quadrigramas e seus domínios.

Tabela I
EXEMPLO DE QUADRIGRAMAS

Quadrigrama	Domínio
Pedra solta portanto recomendo	Aprendizagem
Contrate guia local assim	Aprendizagem
Entrada gratuita você pode	Aprendizagem
Piquenique deliciosa falar presteza	Entretenimento
Atração não pode deixa	Entretenimento
Boa caminhada chegar duna	Entretenimento
Cristalina temperatura super agradável	Estética
Algo tão lindo assim	Estética
Cachoeira formiga pequena queda	Estética
Recomendo entusiasmo não deixar	Evasão
Não dá vontade sair	Evasão
Jalapão vale pena passar	Evasão

2) Pré-processamento dos dados: A base de dados foi dividida em duas, uma com todos os quadrigramas e outra sem os quadrigramas que não tinham classificação. Isso foi feito como prova de conceito para avaliar e confirmar o apresentado por Acuña Rodriguez [5], que afirmaram ser necessário retirar os valores ausentes para que não hava ruído no treinamento, afetando o desempenho do modelo obtido.

Foram rodados testes como prova de conceito incluindo os quadrigramas sem classificação para confirmar o descrito em Acuña Rodriguez [5].

A Tabela II mostra o número de quadrigramas de cada domínio da experiência em seu respectivo atrativo e domínios da experiência, incluindo a quantidade sem classificação.

Utilizando a função *unidecode*[2] do Python todos os quadrigramas foram processados e todas as letras foram passadas para sua versão sem acentos, a fim do classificador não fazer distinção.

Todas as letras foram colocadas em minúsculo e também foi necessário a correção gramatical manual de algumas palavras.

3) Preparação do treinamento: Na Tabela III podemos observar a diversidade das divisões de treinamento e teste nos trabalhos da área e de bases de dados que já são disponibilizadas com os dados de treinamento e teste separados, que é o caso do MNIST[3], CIFAR-100 e CIFAR-10[4].

Como não existe um consenso da melhor proporção para a divisão da base de dados para o treinamento, inicialmente foi escolhido dividir a base em 80% para treinamento e 20% para validação, mantendo-se a proporção para cada domínio. Então, na etapa 5) predições, foram incluídos testes para avaliar a melhor divisão para os dados de treino e teste para este trabalho.

Para realizar a divisão foi utilizada a função *train_test_split*[5] que podemos conseguir a partir do pacote *Scikit-learn*[6]. Resultando na divisão que pode ser observada na Tabela IV, onde podemos ver a quantidade de quadrigramas separados para o treinamento e teste conforme os domínios da experiência.

[2]https://pypi.org/project/Unidecode/

[3]https://keras.io/api/datasets/mnist/

[4]https://www.cs.toronto.edu/~kriz/cifar.html

[5]https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[6]https://scikit-learn.org/stable/

Também como etapa da preparação para o treinamento todas as palavras contidas nos quadrigramas foram transformadas em números utilizando a função *tokenizer*[7] contida dentro da biblioteca *Keras*[8] utilizada junto com o *TensorFlow*[9], para que o modelo de IA gerado pelo *Keras* possa trabalhar de maneira mais rápida e facilitada. O resultado da função é um dicionário de dados chamado *word_index*, a Tabela V mostra alguns exemplos de palavras e seus respectivos números.

Os demais modelos utilizaram uma abordagem semelhante ao *tokenizer* que é a medida TF-IDF (*Term Frequency–Inverse Document Frequency*) por meio da função *TfidfVectorizer* da biblioteca *scikit-learn*. Que é dada por meio da Equação (1).

$$W_{x,y} = tf_{x,y} * \log(\frac{N}{df_x}) \quad (1)$$

Onde $tf_{x,y}$ é a frequência de x em y, df_x número de documentos em x e N o número total de documentos.

Devido ao objetivo desse trabalho ser a classificação automatizada dos quadrigramas nos domínios da experiência como medida de desempenho será utilizada a acurácia baseada na matriz de confusão [10].

A Tabela VI apresenta a estrutura de uma matriz de confusão genérica, onde é possível observar os valores preditos e os valores de referência de cada classe. P representa a classe positiva, N a classe negativa. A diagonal principal VP e VN significam, verdadeiro positivo, quando os valores são classificados corretamente como positivos e verdadeiro negativo quando os valores são classificados corretamente como negativos. FP significa falso positivo, quando o modelo classifica um dado como positivo mas ele é negativo. FN é a classificação de um dado positivo como negativo.

A partir da matriz de confusão é calculada a acurácia que o modelo atingiu. A acurácia calcula o percentual de acertos por meio da Equação (2).

$$Acurácia = \frac{VP + VN}{VP + VN + FN + VP} \quad (2)$$

4) Treinamento: As técnicas de IA utilizadas foram *Random Forest* (RF), *Support Vector Machine* (SVM), *Stochastic Gradient Descent* (SGD), Redes Neurais Artificiais (RNA) e *Multi-layer Perceptron* (MLP). Os hiperparâmetros de cada uma das redes serão descritos a seguir.

Para o treinamento dos modelos e a verificação dos melhores hiperparâmetros será utilizado o método *GridSearchCV*[10] contida na biblioteca *Scikit-learn* [11].

Baseado em árvore de decisão o RF trabalha com nós, podendo ser um nó folha ou um nó de decisão, para cada decisão existe uma subárvore com estrutura semelhante da árvore [12].

Para *Random Forest* os hiperparâmetros treinados são, o número de árvores treinadas (`n_estimators`), o número de características consideradas quando se está procurando a melhor divisão (`max_features`), a profundidade máxima da árvore (`max_depth`) e o parâmetro para medir a qualidade da árvore (`criterion`).

Para o `n_estimators` foram testados os seguintes valores 50, 100, 200, 500 e 1000. O `max_features` pode ser *sqrt* quanto *log2*. Os valores testados para `max_depth` foram de 4, 5, 6, 7, 8, 9 e10. Já `criterion` pode ser *gini* ou *entropy*.

Desenvolvido baseado na teoria de aprendizado estatístico (TAE) e através de diversos princípios que devem ser seguidos, como o limite

[7]https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer

[8]https://www.tensorflow.org/api_docs/python/tf/keras

[9]https://www.tensorflow.org/

[10]https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

Tabela II
COMPOSIÇÃO DO BANCO DE DADOS POR ATRATIVO E DOMÍNIO

Local	Aprendizagem	Entretenimento	Estética	Evasão	Sem Classe
Cachoeira da Formiga	49	52	66	8	25
Cachoeira da Velha	49	76	39	9	1
Serra Espírito Santo	60	53	58	12	15
Dunas	49	48	73	14	16
Mumbuca	38	129	5	0	21
Fervedouro	37	93	33	16	8
PEJ (Geral)	39	65	54	32	10

Tabela III
TRABALHOS DIVISÕES DE TREINAMENTO E TESTE

Trabalho	Treino & Teste
MNIST	85% / 15%
CIFAR-100	83.5% / 16.5%
CIFAR-10	83.5% / 16.5%
Comparação de técnicas de word embedding na análise de sentimentos [6]	80% / 20%
Autoregressive Convolutional Neural Networks for Asynchronous Time Series [7]	80% / 20%
Um processo de classificação de texto: análise de sentimento das opiniões no tripadvisor sobre a atração Oktoberfest Blumenau [8]	75% / 25%
Computer-aided diagnosis for breast cancer classification using deep neural networks and transfer learning [9]	65% / 35%

Tabela IV
BANCO DE DADOS SEPARADOS EM TREINO E TESTE

Domínio	Treino	Teste
Aprendizagem	257	64
Entretenimento	413	103
Estética	263	65
Evasão	73	18

Tabela V
QUADRIGRAMA TRANSFORMADO PELO TOKENIZER

Palavra	Valor
Pedra	210
Solta	474
Portanto	694
Recomendo	312

Tabela VI
MATRIZ DE CONFUSÃO

		Referência	
		P	N
Predição	P	VP	FP
	N	FN	VN

de risco, o SVM busca classificadores com uma boa capacidade de generalização utilizando vetores lineares e não lineares [13].

Para *Support Vector Machine*, o `kernel` utilizado foi o RBF (*radial basis function*), que permite o treinamento de dados não-lineares. Dessa forma, os dois hiperparâmetros testados foram o custo *C* e o *Gamma* (Coeficiente do *Kernel*).

A faixa de valores testados de *C* foram de 0.1, 1, 10, 100 e 1000. Já a faixa de valores para *gamma* foi definida como 1, 0.1, 0.01, 0.001 e 0.0001. Os algoritmos de *Stochastic Gradient* tem o objetivo de diminuir os riscos de cada modelo através do cálculo de cada parâmetros, dos pesos e da função de perda a cada rodada de treinamento fazendo assim os ajustes necessários para os pesos e iniciando uma nova rodada [14].

Para *Stochastic Gradient Descent* os hiperparâmetros disponíveis são o `loss`, que é a função de perda, `penalty` que é a função de penalidade ou regularização, `alpha` constante que é multiplicada pelo termo de regularização e o número máximo de iterações ou também conhecido como épocas `max_iter`.

Os parâmetros treinados para `loss` foram *hinge*, *log*, *modified_huber*, *perceptron* e *squared_hinge*, para `penalty` foram *l2*, *l1*, *elasticnet* e *none*. Para o treinamento de `alpha` foram utilizados os valores 0.0001, 0.001, 0.01, 0.1, 1 e 10 e valores de `max_iter` foram de 100, 500, 1000 e 2000.

Com o intuito de imitar o comportamento do nosso sistema nervoso, surgiram as RNAs, utilizando uma densa camada de neurônios. Como em nosso cérebro esses neurônios têm a capacidade de aprender e passar o conhecimento para a próxima camada [13].

Para Redes Neurais Artificiais, foi utilizada a biblioteca *Keras*. Os hiperparâmetros utilizados foram, o tamanho da camada de *Embedding*, a função de ativação, a função de otimização, `batch size` e o número de épocas.

Os valores treinados para a camada de *Embedding* foram 100, 200 e 500. As funções de ativação escolhidas para o treinamento são *softmax*, *relu* e *sigmoid*. Para o número de neurônios os valores foram 64, 128 e 256. As funções de otimização testadas *SGD*, *RMSprop* e *Adam*. Para o `batch_size` foram testados os valores 8, 16, 32 e 40 e o os valores para o número de épocas foram 10, 30, 100 e 200.

O modelo final da Rede Neural artificial ficou com uma camada de *embedding*, uma camada *Global Average Pooling* e uma camada densa conforme a Figura 2.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 4, 200)            208000

global_average_pooling1d_1   (None, 200)               0
(GlobalAveragePooling1D)

dense_1 (Dense)              (None, 4)                 804

=================================================================
Total params: 208,804
Trainable params: 208,804
Non-trainable params: 0
```

Figura 2. Modelo da estrutura das camadas da Rede Neural Artificial.

Sendo semelhante ao funcionamento de uma RNA, a MLP adiciona mais camadas de neurônios intermediarias e uma camada

de saída. Todos os neurônios são completamente conectados, os neurônios da primeira cama estão conectados à todos da segunda camada e assim por diante [13].

Para *Multi-layer Perceptron* os parâmetros utilizados foram `solver` para a otimização dos pesos, `activation` função de ativação da camada oculta, `alpha` como termo de regularização, `hidden_layer_sizes` que representa o número de neurônios na camada oculta e `max_iter` que representa o número máximo de iterações.

Os valores treinados para o `solver` foram *lbfgs*, *sgd* e *adam*, já para `activation` foram utilizados *identity*, *logistic*, *tanh* e *relu*. Os valores de `Alpha` 0.0001, 0.001, 0.01, 0.1, 1 e 10, para o `hidden_layer_sizes` foram testados os seguintes valores (100,), (100,2) e (100,3). Para o `max_iter` foram escolhidos os valores 10, 100, 200, 500, 1000 e 2000.

A Tabela VII mostra o tempo de execução do treinamento de cada modelo utilizando a base dados utilizando a divisão de 80% treinamento e 20% teste. O parâmetro `n_jobs` do *GridSearchCV* tem a função de definir o número de núcleos utilizados durante a execução do script, como padrão ele utiliza apenas um, se for atribuído o número -1 ele utiliza todos os núcleos do processador.

O primeiro modelo testado foi o MLP, com o `n_jobs` padrão, o tempo de execução foi de 2 horas 40 minutos e 7 segundos e após foi testado com o número máximo de núcleos do processador, com tempo de 1 hora 59 minutos e 11 segundos. Por tanto todos os outros modelos foram testados com o `n_jobs` com valor de -1.

Tabela VII
TEMPO DE EXECUÇÃO DOS MODELOS

Modelo	Tempo de Execução
RF	00:00:39
SVM	00:00:03
SGD	00:00:01
MLP	01:59:11
RNA	04:37:48

5) Predições: As predições foram iniciadas identificando os melhores hiperparâmetros a partir da utilização do *GridSearchCV*, conforme apresentado na Tabela VIII. São apresentados os Modelos, os hiperparâmetros e a Acurácia obtida, cujo melhor método foi o RNA com 74,21% de Acurácia.

Utilizando os mesmos hiperparâmetros e o modelo com melhor acurácia (RNA), foi realizada uma nova bateria de testes utilizando a base de dados contendo os quadrigramas sem classificação, cujos resultados são apresentados na próxima seção.

Como a última etapa dos testes foi realizado o experimento da melhor divisão da base de treinamento e teste, nessa etapa foi utilizado os mesmos hiperparâmetros já descritos para o modelo que apresenta melhor acurácia e a base de dados apenas com os quadrigramas classificados dentro dos quatro domínios da experiência.

A Tabela IX mostra a divisão da base em diversas proporções para o treinamento e teste dos modelos. O primeiro teste foi com uma divisão de 60% para treinamento e 40% para teste atingindo uma acurácia de 67.20%. Dividindo a base em 70% para treinamento e 30% para teste a acurácia obtida foi de 67.64%.

A divisão de 80% para treinamento e 20% para teste foi a utilizada na primeira rodada de treinamento dos modelos e obteve 74.21% de acurácia, sendo a segunda melhor divisão para a base utilizada nesse trabalho.

Atingindo 76.19% de acurácia a divisão de 85% para treinamento e 15% para teste obteve a melhor acurácia dos testes realizados. A última divisão testada foi de 90% para treinamento e 10% para teste e chegou a uma acurácia de 69.84%.

Sendo assim, foi realizada uma nova rodada de treinamento, agora com a base mais bem ajustada com a divisão de 85% para treinamento e 15% para teste, e cujos resultados são apresentados na próxima seção.

C. Tecnologias

O script foi desenvolvido em um computador pessoal com processador Ryzen 7 3700X de 8-núcleos a 3,59 GHz, com sistema operacional Windows 10 Pro 21H2. Foram utilizadas as seguintes aplicações e bibliotecas:

- TensorFlow, versão 2.9.0
- Linguagem Python, versão 3.9.2
- Spyder, versão 5.2.2
- Scikit-learn, versão 1.1.2

II. RESULTADOS E DISCUSSÕES

Na Tabela VIII, observamos os melhores hiperparâmetros e a acurácia dos modelos propostos nesse trabalho e na Tabela IX o resultado do experimento feito com as divisões da base de dados, chegando à conclusão de que a melhor divisão para o treinamento é 85% treinamento e 15% para teste. Dessa forma os treinamentos foram refeitos e o resultado pode ser conferido a seguir.

A. Medida de qualidade dos modelos

A métrica de desempenho escolhida para ser utilizada neste trabalho foi a acurácia. O melhor resultado para as técnicas apresentadas foi de 77.78% para a Rede Neural Artificial, seguido do *Stochastic Gradient Descent*, *Support Vector Machine*, *Multi-layer Perceptron* e pelo *Random Forest*.

Na Figura 3, os resultados são apresentados por modelo e dentro de cada modelo estão organizados por base de dados. Em que o primeiro valor de cada modelo é o treinamento realizado com a base de dados contendo os quadrigramas classificados nos quatro domínios da experiência mais os sem classificação. A segunda base de dados apenas com os quadrigramas classificados e divisão de 85% para treinamento e 15% para testes e a última base com os quadrigramas classificados e divisão de 80% para treinamento e 20% para testes.

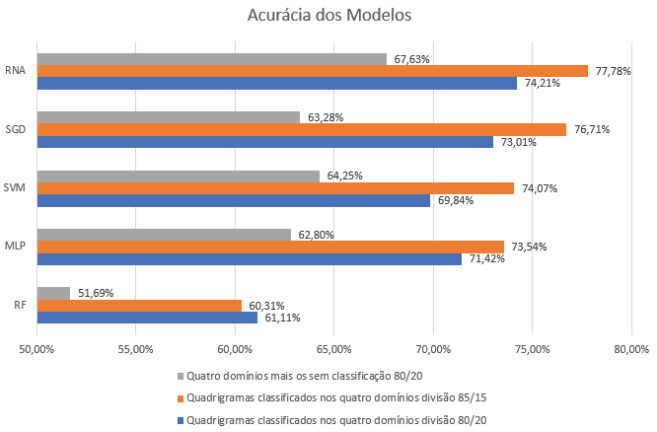


Figura 3. Gráfico comparativo das acurácias dos modelos aplicados em diferentes bases de dados.

Analisando o gráfico da Figura 3 podemos observar uma média de melhora de 3.4% na acurácia dos primeiros quatro modelos utilizando a base com divisão 85% para treinamento e 15% para testes, assim como uma performance semelhante nos modelos Rede Neural Artificial e *Stochastic Gradient Descent* com 77.78% e 76.71% de acurácia respectivamente, e por conseguinte uma matriz de confusão distribuída de maneira similar que serão mostradas na próxima seção.

B. Matrizes de Confusão

A matriz de confusão do modelo Rede Neural Artificial é apresentada em todas as tabelas ao lado da matriz de confusão dos demais modelos. A do modelo *Stochastic Gradient Descent* é apresentada na

Tabela VIII
RESULTADO DOS EXPERIMENTOS GRIDSEARCHCV

Algoritmo	Hiperparâmetros	Acurácia
RF	n_estimators=50, max_features='sqrt', max_depth=10, criterion='gini'	61.11%
SVM	C=1000, gamma=0.001	69.84%
SGD	loss='hinge', alpha=0.001, penalty='elasticnet', max_iter=100	73.01%
MLP	solver='SGD', activation='tanh', alpha=0.01, hidden_layer_sizes=(100, 3), max_iter=2000	71.42%
RNA	embedding= 200, activation='softmax', batch_size=40, optimizer='adam', epochs=200	**74.21%**

Tabela IX
RESULTADO DAS DIVISÕES DA BASE DE TREINAMENTO E TESTE

Treinamento	Teste	Acurácia
90	10	69.84%
85	**15**	**76.19%**
80	20	74.21%
70	30	67.64%
60	40	67.20%

Tabela X. A do modelo *Support Vector Machine* é apresentada na Tabela XI. A do modelo *Multi-layer Perceptron* é apresentada na Tabela XII, e a matriz de confusão do *Random Forest* é apresentada na Tabela XIII. Em todas as tabelas AP significa "Aprendizagem", EN significa "Entretenimento", ES significa "Estética" e EV significa "Evasão", referente a classificação dentro dos domínios da experiência, as linhas horizontais são referente a predição e as colunas são a referência.

Tabela X
MATRIZ DE CONFUSÃO – RNA x SGD

		RNA				SGD			
		AP	EN	ES	EV	AP	EN	ES	EV
Predição	AP	**28**	11	4	1	**22**	7	3	0
	EN	13	**64**	3	3	15	**67**	4	3
	ES	0	2	**49**	1	4	6	**50**	2
	EV	0	3	1	**6**	0	0	0	**6**

O RNA se mostrou melhor na predição da classe Aprendizagem com 28 classificados corretamente contra 22 do SGD, resultado igual de acerto na classe Evasão ambos com 6 e resultado inferior nas classes Entretenimento e Estética, tendo predito 64 e 49 respectivamente contra 67 e 50 do SGD. O RNA mostrou um erro maior na predição das classes Entretenimento e Estética, tendo errado 16 e 8, o SGD errou 13 e 7.

Tabela XI
MATRIZ DE CONFUSÃO – RNA x SVM

		RNA				SVM			
		AP	EN	ES	EV	AP	EN	ES	EV
Predição	AP	**28**	11	4	1	**24**	13	5	1
	EN	13	**64**	3	3	15	**62**	3	2
	ES	0	2	**49**	1	2	2	**48**	2
	EV	0	3	1	**6**	0	3	1	**6**

Olhando as colunas de referências o SVM se mostrou igualmente capaz na classificação da classe Evasão ambos com seis acertos. Nas demais classes teve resultado minimamente inferior, tendo uma diferença na predição de quatro na classe Aprendizado, dois na classe Entretenimento e um na classe Estética.

A Matriz de confusão do MLP mostra a capacidade de predição do modelo na classe Entretenimento onde obteve resultado semelhante ao RNA, ambos com 64 acertos. Nas demais classes o RNA se mostrou superior acertando seis a mais na classe Aprendizado, um na classe Estética e dois na classe Evasão.

Tabela XII
MATRIZ DE CONFUSÃO – RNA x MLP

		RNA				MLP			
		AP	EN	ES	EV	AP	EN	ES	EV
Predição	AP	**28**	11	4	1	**22**	12	3	0
	EN	13	**64**	3	3	15	**64**	6	5
	ES	0	2	**49**	1	3	3	**48**	2
	EV	0	3	1	**6**	0	1	0	**4**

Tabela XIII
MATRIZ DE CONFUSÃO – RNA x RF

		RNA				RF			
		AP	EN	ES	EV	AP	EN	ES	EV
Predição	AP	**28**	11	4	1	**3**	1	0	0
	EN	13	**64**	3	3	37	**78**	25	10
	ES	0	2	**49**	1	1	1	**32**	0
	EV	0	3	1	**6**	0	0	0	**1**

Por fim a matriz de confusão do RF mostra uma grande vantagem na predição da classe Entretenimento, onde o modelo acerta 14 a mais que o RNA. Entretanto nas demais classes o RF erra muito mais, com apenas três acertos na classe Aprendizagem, 32 na classe Estética e um na classe Evasão.

C. Discussões

Como validação o modelo de redes neurais artificias foi exposto à uma nova base de dados para classificação dos quadrigramas. Utilizando as mesmas técnicas de coleta e de preparação de dados, foram obtidos 5114 quadrigramas feitos a partir de comentários a respeito da ilha de Superagui no estado do Paraná. A Tabela XIV mostra alguns exemplos dos quadrigramas classificados.

Tabela XIV
EXEMPLO DE QUADRIGRAMAS CLASSIFICADOS

Quadrigrama	Domínio
Isolada alta temporada levar	Aprendizagem
Pouco restaurante abrem somente	Aprendizagem
Serem descoberta alugar bicicleta	Entretenimento
Visual maravilhoso apena casa	Entretenimento
Incrível sensação sair alma	Estética
Nome linda exótica selvagem	Estética
Uau lugar linda dá	Evasão
Todo lugar esconder mundo	Evasão

A Tabela XV mostra a quantidade de quadrigramas classificados pelo modelo de redes neurais artificiais. Dos 5114 quadrigramas disponíveis, foram classificados 1251 como aprendizagem, representando 24.46% do total, 3244 como entretenimento, sendo 63.43%

da base, 500 classificados como estética, 9.77% do total e por fim 119 quadrigramas classificados como evasão, representando 2.32% da quantidade total de quadrigramas.

Tabela XV
QUANTIDADE DE QUADRIGRAMAS CLASSIFICADOS DE SUPERAGUI

Domínio	Quantidade
Aprendizagem	1251
Entretenimento	3244
Estética	500
Evasão	119

Os quadrigramas classificados serão enviados a um especialista do turismo para validação e aprovação do modelo gerado.

Como trabalhos futuros pode-se seguir diversas abordagens diferentes, como a obtenção de uma base de dados maior para o treinamento do modelo, especialmente com uma quantidade de quadrigramas maior na classe evasão. A aplicação do modelo em uma API de classificação de quadrigramas nos domínios da experiência. Além do desenvolvimento de um novo modelo de IA melhor ajustado com abordagens não realizadas neste trabalho como a implementação de CNN realizada por [15] e o testes de diversas técnicas de *word embedding* conforme [6].

Trabalhando com um classificador baseado em uma rede LSTM e também com o intuito de classificar a experiência de um turista porém apenas como positiva ou negativa, [8] conseguiram 93% de acurácia em seu modelo de IA.

Sendo assim, podemos concluir um resultado satisfatório com os modelos RNA e SGD na classificação dos quadrigramas contudo com uma possibilidade grande de melhoria.

REFERÊNCIAS

[1] R. Weiscbedel et al., "White Paper on Natural Language Processing," in Speech and Natural Language, Proceedings of a Workshop, 1989, pp. 481–493. doi: 10.3115/1075434.1075526.

[2] B. Joseph and J. H. Gilmore, "Work Is Theatre f Every Business a Stage."

[3] E. F.; KAIZER, M. F. A.; CARACRISTI, J. E.; FEGER, J. E.; MARYNOWSKI, and T. M. SILVA, "Análise da experiência relatada pelos turistas ao visitar o Parque Estadual do Jalapão (PEJ) – TO, Brasil," Ateliê do Turismo, vol. 5, no. 1, pp. 183–204, 2021, [Online]. Available: https://periodicos.ufms.br/index.php/adturismo/article/view/12354.

[4] M. de F. de A. Caracristi, J. E. Feger, T. M. da Silva, and J. E. Marynowski, "Uma Viagem pelo Jalapão, Brasil: análise das experiências turísticas," Revista Paranaense de Desenvolvimento (RPD), vol. 41, no. 138, pp. 89–110, 2021.

[5] E. Acuña and C. Rodriguez, "The treatment of missing values and its effect in the classifier accuracy."

[6] G. Ziger and F. Roberto Pereira, "Comparação de técnicas de word embedding na análise de sentimentos."

[7] M. Bińkowski, G. Marti, and P. Donnat, "Autoregressive Convolutional Neural Networks for Asynchronous Time Series," Mar. 2017, [Online]. Available: http://arxiv.org/abs/1703.04122

[8] M. Crescencio, ; Alexandre, L. Gonçalves, and J. L. Todesco, "Um processo de classificação de texto: Análise de sentimento das opiniões no Tripadvisor sobre a atração Oktoberfest Blumenau."

[9] H. Aljuaid, N. Alturki, N. Alsubaie, L. Cavallaro, and A. Liotta, "Computer-aided diagnosis for breast cancer classification using deep neural networks and transfer learning," Comput Methods Programs Biomed, vol. 223, Aug. 2022, doi: 10.1016/j.cmpb.2022.106951.

[10] K. Faceli, A. C. Lorena, J. Gama, and A. C. P. L. F. Carvalho, Inteligencia artificial: uma abordagem de aprendizado de maquina. 2011.

[11] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E. Scikit-learn: Machine learning in Python. J Mach Learn Res. 2011;12:2825–30.

[12] REZENDE, Solange O., MONARD, Maria Carolina, CARVALHO, André C. P. L.. Sistemas Inteligentes para Engenharia: Pesquisa e Desenvolvimento. Anais III Workshop de Sistemas Inteligentes para Engenharia. Belo Horizonte: Editora UFMG, 1999.

[13] K. Faceli, A. C. Lorena, J. Gama, and A. C. P. L. F. Carvalho, Inteligencia artificial: uma abordagem de aprendizado de maquina. 2011.

[14] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," 2016.

[15] S. Silva and A. Serapiao, "Ensaio em deep learning com aplicações em imagens e textos Formal models View project Ensaio em Deep Learning com aplicações em imagens e textos View project." [Online]. Available: https://www.researchgate.net/publication/335675987.