# ESTUDO DE METODOLOGIAS DE RECONHECIMENTO DE CARACTERES MANUSCRITOS

Ana L. C. F. de Oliveira, Augusto dos S. G. Vaz, Gabrielly C. Guimarães, Vitor E. A. Costa
*Departamento de Computação, Universidade Federal de São Carlos*
*São Carlos, Brasil*

## I. INTRODUÇÃO

O reconhecimento de caracteres é uma extensão da área de reconhecimento de padrões, apesar disso, foi a necessidade do reconhecimento de caracteres que motivou os estudos em reconhecimentos de padrões e análises de imagens. Suas origens datam de 1870, porém foi 30 anos depois, em 1900, que as primeiras tentativas bem-sucedidas foram realizadas pelo cientista russo Tyurin.

Com o desenvolvimento dos primeiros computadores, a versão moderna do *Optical Character Recognition* (OCR) apareceu em meados de 1940. Assim, pela primeira vez a OCR foi utilizada como uma forma de processamento de dados.

### *A. Objetivo*

Este trabalho tem por objetivo realizar um estudo generalizado sobre o Reconhecimento Óptico de Caractere, além disso, será realizado um estudo de caso na utilização do algoritmo de k-Nearest-Neighbors e Redes Neurais para o reconhecimento de padrões, com foco na análise de caracteres manuscritos.

A motivação para esse relatório é entender melhor as metodologias envolvidas no reconhecimento de padrões e também comparar os resultados obtidos por diversos trabalhos que abordam a temática.

### *B. Semântica*

Reconhecimento de caracteres é mais conhecido como OCR, devido ao fato de lidar com o reconhecimento de dados processados opticamente ao invés de magneticamente.

A Figura 1 [1] apresenta os diferentes esquemas que existem dentro do termo reconhecimento de caracteres.

As definições dos principais esquemas de OCR são:

*i) Reconhecimento de caractere de fonte fixa*: reconhecimento de fontes específicas (Times New Roman, Arial, Courier) em caracteres digitados;

*ii) Reconhecimento de caractere on-line:* reconhecimento de caracteres desenhados à mão, porém, além da imagem, também é fornecida a informação de tempo de cada risco;

*iii) Reconhecimento de caractere manuscrito*: reconhecimento de caracteres desenhados à mão, não-conectados e sem caligrafia;

*iv) Reconhecimento de Script*: reconhecimento de caracteres sem restrição que podem, ou não, estarem conectados e em letra cursiva.

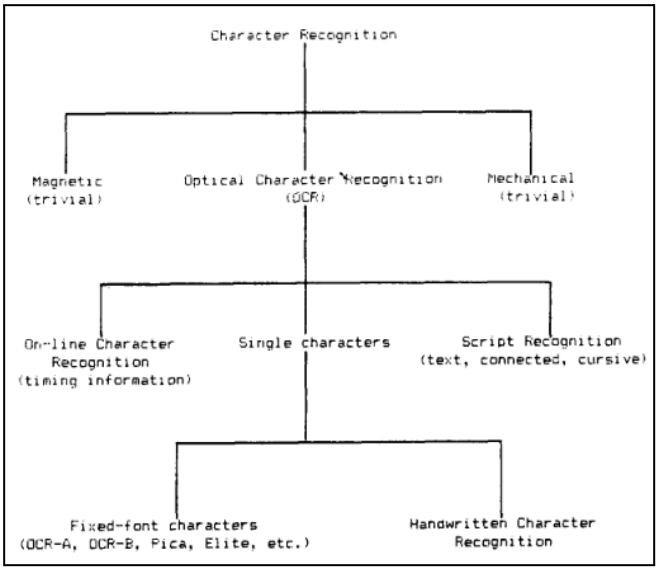


Fig. 1 Diagrama dos esquemas de OCR

## II. RECONHECIMENTO DE CARACTERE MANUSCRITO

Apesar do OCR já ser um problema resolvido e possuir ótimas soluções quando se trata de *Handwritten Character Recognition* (HCR), ao abordar tópicos mais complexos como textos multilíngues, ainda há avanços e descobertas a serem feitas. Nesse trabalho haverá uma ênfase melhor no reconhecimento *offline*, isto é, manuscrito.

No reconhecimento de caractere offline, o indivíduo escreve no papel e após escaneada ou digitalizada, a imagem é salva. Um dos obstáculos nesse tipo de reconhecimento de padrões é a necessidade de imagens com boas resoluções, o que nem sempre é possível devido à natureza do escaneamento, e um bom pré-processamento. Caso haja muitos *scripts* — quando se trata de OCR, *scripts* são os símbolos inerentes a cada linguagem e alfabeto, além da escrita individual de cada pessoa — em um único arquivo, isso acaba se tornando um problema.

O processo de reconhecimento de caractere manuscrito é dividido em 6 partes, sendo essas: aquisição da imagem, pré-processamento, segmentação, extração de *features*, classificação e pós-processamento.

A aquisição da imagem é basicamente o modo como o texto a ser reconhecido será obtido. Para isso, vários métodos podem ser usados, como o uso de dispositivos *touchscreen* ou mesas digitalizadoras, que devolvem um *input* já em formato digital ou uso de imagens propriamente ditas, como fotos e escaneamento do texto.

O pré-processamento é a etapa de tratamento do *input* bruto. Nele, são aplicadas técnicas de optimização que visam normalizar

e reduzir ruído para aumentar a eficiência e eficácia da IA. Dentre essas técnicas, as mais usadas são a remoção de sombras, alinhamento de caracteres com inclinação, normalização do tamanho da letra além da centralização de cada caractere.

A segmentação, por sua vez, é a divisão do texto em caracteres individuais. Normalmente isso é feito dividindo cada linha do texto em uma *row*, depois cada palavra dessa *row* é dividida em uma *column* e então, finalmente, cada letra é extraída.

A extração da *feature* desmembra cada letra em pequenos pedaços unitários, como "laços", "retas" e "pontos". Dessa maneira, cada letra é composta por uma combinação de várias features.

A classificação recebe cada caractere dividido em várias features e usa um classificador para estimar com uma certa confiança que letra é essa. Vários algoritmos podem ser empregados nessa etapa, porém neste trabalho será aprofundado o KNN..

O pós-processamento é uma fase de aperfeiçoamento do algoritmo. Ela verifica a corretude da classificação através do estudo da linguagem propriamente dita. Dessa maneira, letras mal classificadas que geraram um erro de ortografia (*typo*, em inglês) podem ser corrigidas através de mecanismos semelhantes ao de corretores de texto usados nos teclados de celulares.

## III. TRABALHOS RELACIONADOS

O trabalho de Ghosh et al [3] apresenta uma tabela de acurácia de diferentes esquemas de reconhecimento *offline*. Como citado anteriormente, esses trabalhos apresentam altas acurácias, alguns chegando a 100%, como por exemplo o trabalho de Gokapumar et al. [6].

| Trabalho | Língua | Formato | Acurácia (%) |
|---|---|---|---|
| Pal e Chaudhuri [4] | Romano, Chinês, Árabe, Devanágari, Bengali | Texto | 97,33 |
| Benjelil et al. [5] | Árabe e Latim | Palavra | 97,5 |
| Gokapumar et al. [6] | Canarim, Telugo, Tâmil-Malaiala, Hindi, Inglês | Texto | 100 |
| Engammalan and Ismail [7] | Árabe, Latim | Linha | 96, 8 |
| Moalia et al. [8] | Árabe, Latim | Palavra | 100 |
| Jawahar et all. [9] | Devanágari, Telugo | Palavra | 92,3 a 99,86 |

*Tab. 1 Acurácia de reconhecimento de diferentes esquemas de HCR offline*

| **Trabalho** | **Língua** | **Formato dos dados** | **Acurácia (%)** |
|---|---|---|---|
| KNN [2] | Algarismos ocidentais | Dígitos | Max: 100% Min: 83,3% *Figura 2 |
| CNN [10] | Algarismos ocidentais | Letras e Dígitos | Max: 92,91% Min: 65,32% |
| SVM [11] | Algarismos ocidentais | Letras | Max: 88,46% Min: 73,44% |
| HMM [12] | Algarismos ocidentais | Palavras | Em média 98% |

*Tab. 2 Trabalhos relacionados à OCR*

### *A. Um Método Para Classificação De Dados Baseado Nos K-Vizinhos Mais Próximos Para O Reconhecimento De Caracteres*

Nesse artigo é explorado o reconhecimento de caracteres utilizando o algoritmo KNN como base, que classifica as amostras não classificadas a partir de sua similaridade com outras amostras já classificadas. Para a análise de similaridade utiliza-se métricas tais como Distância de Manhattan e distância Euclidiana.

Para tornar o algoritmo mais robusto, o autor propõe uma etapa de binarização da imagem e enfatiza a importância do pré-processamento para o bom desempenho do algoritmo.

| Base de Dados | Acurácia (%) | | | |
|---|---|---|---|---|
| | Método proposto k = 3 | | | |
| | 112 atributos | 64 atributos | 32 atributos | 16 atributos |
| optdigits-orig.tra | 97,8 | 98,0 | 98,0 | 97,8 |
| optdigits-orig.cv | 96,8 | 97,1 | 97,1 | 87,4 |
| optdigits-orig.wdep | 97,5 | 97,9 | 98,0 | 96,8 |
| optdigits-orig.windep | 98,9 | 98,9 | 98,9 | 98,4 |
| Artificial-numeros | 100,0 | 100,0 | 100,0 | 100,0 |
| Real-numeros1 | 93,3 | 93,3 | 96,7 | 83,3 |
| Real-numeros2 | 90,0 | 90,0 | 90,0 | 90,0 |

| | | | | |
|---|---|---|---|---|
| Real-numeros3 | 100,0 | 100,0 | 100,0 | 96,7 |
| Real-numeros1,2e 3 | 97,8 | 97,8 | 97,8 | 97,8 |
| Artificial-Real1,2e 3 | 97,5 | 97,5 | 97,5 | 97,5 |

*Tab. 3 Resultado de validação das bases de dados [2]*

### *B. Reconhecimento de Escrita Manual Utilizando Rede Neural Convolucional*

A abordagem de Redes Neurais Artificiais é considerada uma das melhores para reconhecer manuscrito, elas ajudam a simular como o cérebro humano funciona quando lendo um manuscrito de forma mais simplificada.

No estudo foi apresentado o sistema OCR (Optical Character Recognition) utilizando Rede Neural Convolucional, que é composto por 5 fases para o reconhecimento:

1. Aquisição da imagem e digitalização
2. Pré-processamento
3. Segmentação
4. Extração de features
5. Classificação e Reconhecimento

| No. of Training Images | No. of Testing Images | Average Accuracy (%) |
|---|---|---|
| 200 | 200 | 65.32% |
| 300 | 200 | 74.43% |
| 500 | 200 | 80.84% |
| 600 | 200 | 85.21% |
| 800 | 200 | 87.65% |
| 1000 | 200 | 92.91% |

*Fig. 2 Tabela de resultados da CNN*

### *C. Support Vector Machine (SVM) Para Reconhecimento de Manuscrito em Inglês*

No trabalho é estudado a utilização do SVM para classificar caracteres, porém antes de chegar ao estágio de classificação, a imagem precisa ser tratada, seguindo as etapas de pré-processamento e extração de features. Então, inicia a fase onde acontece de fato o reconhecimento do caractere e, portanto, sua classificação.

O SVM foi treinado com conjuntos separados, sendo um composto por letras maiúsculas e o outro por letras minúsculas. Foram utilizados os *datasets* NIST, considerando 189.411 amostras de letras minúsculas, 217.812 para letras maiśuculas e 407.223 para a combinação de ambas.

Em conclusão, pôde-se observar que a maior acurácia foi alcançada com o conjunto de letras maiúsculas utilizando 53.662 support vectors, como mostra a tabela a seguir:

| Data Set | Samples | Epsilon set tolerance of termination criteria | Number Support Vector (SV) | Accuracy |
|---|---|---|---|---|
| TD1 (Lowercase) | 189,411 | 0.07 | 58,563 | 86.0077% (32578/37878) |
| TD2 (Uppercase) | 217,812 | 0.02 | 53,627 | 88.4671% (43095/48713) |
| | | 0.13 | 53,662 | 88.4671% (43095/48713) |
| TD3 (Lowercase+Uppercase) | 407,223 | 0.004 | 179,943 | 73.4464% (63598/86591) |

*Fig. 3 Tabela de resultados do SVM*

### *D. Reconhecimento Off-Line de Escrita Cursiva Utilizando Modelos Ocultos de Markov*

Os HMMs (Hidden Markov Models) têm sido amplamente utilizados na área de reconhecimento de voz, reconhecimento de padrões de imagens, como a classificação de formas e reconhecimento facial. Assim, esse modelo se encaixa no objetivo de reconhecer manuscritos cursivos.

Os autores estudaram duas diferentes topologias do HMM, uma para grandes e outra para pequenos vocabulários. A taxa de reconhecimento deste sistema fica entre 79% e 91%, dependendo do tamanho do vocabulário e da qualidade dos dados de entrada.

Foram feitos 2 experimentos, no primeiro um dicionário foi criado, composto por 150 palavras aleatórias em inglês e 5 pessoas diferentes escreveram cada palavra 4 vezes. Assim foram obtidos 4 *datasets* com 750 palavras cada. A, B e D foram usados para treino e C para teste.

No segundo experimento, as palavras eram nomes de cidades e vilarejos, portanto haviam letras maiúsculas. Assim como no primeiro experimento, o dicionário possuía o mesmo número de palavras, o mesmo número de pessoas e palavras para testes e treinos.

Pode-se concluir que a taxa de reconhecimento nos dois experimentos foi acima de 98%, e que, sob os cenários descritos anteriormente é possível atingir uma taxa de reconhecimento de manuscritos cursivos *off-line* próximo ao atingido com caracteres manuscritos isolados.

Table 4. Results of experiment 1

| Writer | Test set A | Test set B | Test set C | Test set D | Average |
|---|---|---|---|---|---|
| 1 | 99.33 | 96.67 | 98.67 | 98.00 | 98.17 |
| 2 | 97.33 | 96.67 | 95.33 | 97.33 | 96.67 |
| 3 | 100 | 99.33 | 100 | 100 | 99.83 |
| 4 | 96.67 | 96.00 | 99.33 | 98.00 | 97.50 |
| 5 | 100 | 100 | 100 | 99.33 | 99.83 |
| Average | 98.67 | 97.73 | 98.67 | 98.53 | 98.40 |

*Fig. 4 Resultados do primeiro experimento com HMM*

Table 5. Results of experiment 2

| Writer | Test set A | Test set B | Test set C | Test set D | Average |
|---|---|---|---|---|---|
| 1 | 98.67 | 94.67 | 96.00 | 98.00 | 96.83 |
| 2 | 98.67 | 99.33 | 98.00 | 99.33 | 98.83 |
| 3 | 96.67 | 96.67 | 96.67 | 95.33 | 96.33 |
| 4 | 100 | 100 | 99.33 | 100 | 99.83 |
| 5 | 99.33 | 100 | 98.67 | 100 | 99.50 |
| Average | 98.67 | 98.13 | 97.73 | 98.53 | 98.27 |

*Fig. 5 Resultados do segundo experimento com HMM*

## VI. METODOLOGÍA CIENTÍFICA

Dado os resultados da comparação entre os trabalhos analisados, o algoritmo KNN se mostra o mais adequado para os fins de nossa pesquisa, tendo acurácia média superior aos demais e sendo um algoritmo popular e já estudado na matéria de Inteligência Artificial. Desse modo, iremos estudar mais a fundo o reconhecimento de caracteres manuscritos usando este algoritmo.

### *A. Como funciona o k-NN*

O algoritmo KNN classifica uma amostra de um conjunto de dados, no nosso caso, uma imagem de um caractere, baseando-se na classificação dos k vizinhos mais próximos a ela, isto é, aquelas

quais os atributos mais se assemelham. Desse modo, usamos como métrica de distância entre as amostras a distância euclidiana e, ou, distância de Manhattan.

Como exemplos, imagine um dataset em que cada amostra contém dois atributos que a descreve e pode fazer parte de duas classes, triângulo ou bola. Desse modo, teremos um gráfico bidimensional contendo as amostras desta maneira

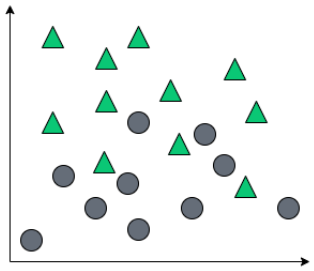

*Fig. 6 Distribuição de amostras*

Assim, o algoritmo knn analisa os k vizinhos (sendo k um parâmetro escolhido arbitrariamente) e suas distâncias para a amostra ainda não classificada.

Abaixo segue um pseudocódigo do algoritmo.

**Algoritmo 1:** Algoritmo k-NN

```
Seja D conjunto de objetos, z o objeto a classificar e k vizinhos prox.
Para cada objeto d_i de D faça
  Calcule distancia(d_i,z)
Fim para
Ordenar distâncias obtidas
Conjunto Dz recebe os k vizinhos mais próximos de z
Exemplo z recebe o rótulo de maior frequência em Dz
```

*Fig. 7 Pseudocódigo do kNN*

## *B. Extração de atributos*

Há diversas formas de se extrair atributos de um um caractere, entretanto, decidimos explorar mais a fundo o método de zoning utilizado no artigo KNN[2] devido seu alto desempenho. O método usado consiste em, primeiramente, preencher uma matriz intermediária de inteiros I com 32 colunas e 32 linhas. Assim, cada célula dessa matriz conterá a porcentagem de pixels brancos ou pretos contidos na área dessa célula. Posteriormente, convertemos essas porcentagens em 1 caso o número de pixels brancos seja superior a 50% e 0 caso seja menor, resultando em uma matriz binária semelhante à apresentada.

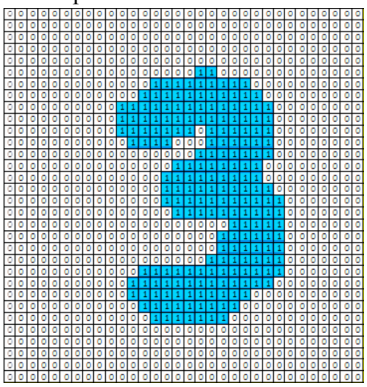

*Fig. 8 Matriz de pixels*

Por fim, definimos 112 atributos deste modo:

*i)* atributos 1 ao 32: quantidade de bits 1 por coluna

*ii)* atributos 33 ao 64: quantidade de bits 1 por coluna

*iii)* atributos 65 ao 96: $\sum_{i=0}^{64} atributo\,i \; + \; atributo\,i+1$

*iv)* atributos 97 ao 112: $\sum_{i=65}^{96} atributo\,i \; + \; atributo\,i+1$

## *C. IMPLEMENTAÇÃO*

O algoritmo foi implementado em Python e foi aplicado o classificador KNN da biblioteca *sklearn* em um conjunto de dados [13] de caracteres segmentados, de formato ".csv", que contém 394120 imagens de caracteres manuscritos. O conjunto é desbalanceado, como pode ser visualizado na *Fig. 6*.

# IV.CONCLUSÕES

O algoritmo de k-Vizinhos Mais Próximos apresentou uma acurácia de 95,19%, um F1-score de 93,44%, um *recall* 92,25% e uma precisão de 94,89%.

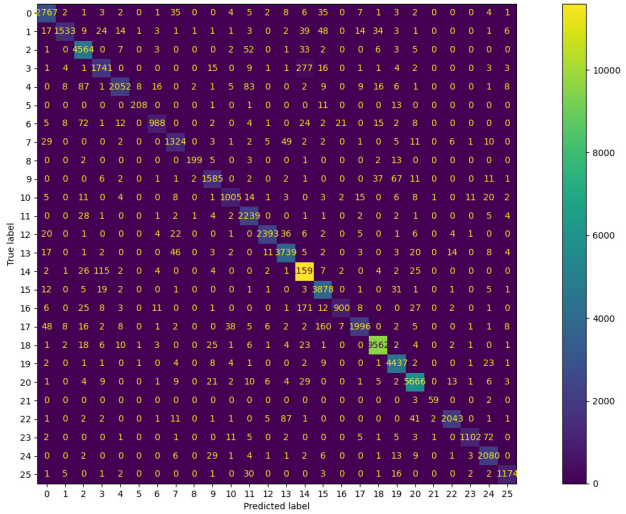


*Fig. 9 Matriz de confusão*

A Figura 5 apresenta a matriz de confusão resultante da predição do algoritmo de KNN. A primeira observação a ser feita é a discrepância de coloração entre a letra "O" das demais letras. Isso deve-se à natureza desbalanceada do conjunto de dados, enquanto que a letra "O" possui 58 mil imagens, algumas chegam a ter pouco mais de 300, como demonstra a Figura 6. Além disso, percebe-se que o algoritmo, muito raramente, confundiu essa letra com as demais, o que é justificável pois a letra "O" é bem simples e icônica.

Por outro lado, entre todas as confusões, as que ocorreram com maior frequência foi as entre a letra "D", predita como "O" pelo algoritmo, resultando em 301 predições erradas.

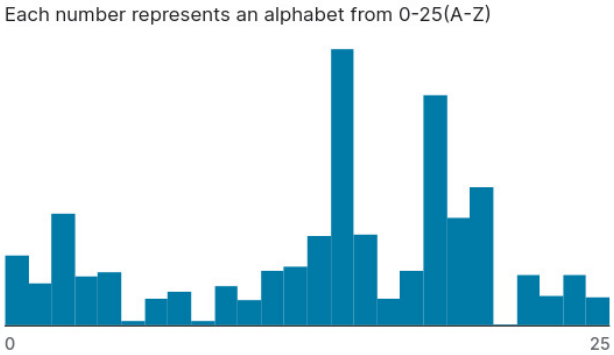


*Fig. 10 Gráfico representando a distribuição das letras*

# REFERÊNCIAS

[1] MANTAS, J. An Overview of Character Recognition

Methodologies. Pattern Recognition, Vol. 19. No. 6. pp 425 - 430, 1986. Disponível em: <https://www.sciencedirect.com/science/article/pii/0031320386900403>. Acesso em: 15 mar. 2023.

[2] SEMAAN, G. S. et al. Um método para classificação de dados baseado nos k-vizinhos mais próximos para o reconhecimento de caracteres. A pesquisa operacional como ferramenta de governança em projetos estratégicos. Anais... In: XIX SIMPÓSIO DE PESQUISA OPERACIONAL E LOGÍSTICA DA MARINHA. Brasil: Marinha, 6 nov. 2019. Disponível em: <https://www.marinha.mil.br/spolm/sites/www.marinha.mil.br.spolm/files/UM%20M%C3%89TODO%20PARA%20CLASSIFICA%C3%87%C3%83O%20DE%20DADOS%20BASEADO%20NOS%20K-VIZINHOS%20MAIS%20PR%C3%93XIMOS%20PARA%20O%20RECONHECIMENTO%20DE%20CARACTERES_0.pdf>. Acesso em: 15 mar. 2023.

[3] GHOSH, T. et al. Advances in online handwritten recognition in the last decades. Computer Science Review, Vol. 46. 2022. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1574013722000491>. Acesso em: 15 mar. 2023.

[4] PAL, U. CHAUDHURI, B. B. Identification of different scripts lines from multi-script documents. Image and Vision Computing Vol. 21. No. 11. pp 945 - 954, 2003.

[5] BENJELIL, M. KANOUN, S. MULLOT, R. A.M. Alimi, Arabic and Latin Script Identification in Printed and Handwritten Types Based on Steerable Pyramid Features, in: Proc. of International Conference on Document Analysis and Recognition, 2009, pp. 591 - 595.

[6] GOPAKUMAR, R. SUBBAREDDY, N. V. MAKKITHAYA, K.K. U.D. Acharya, Zone-based structural feature extraction for script identification from Indian documents, in: Proc. of International Conference on Industrial and Information Systems, 2010, pp. 420 - 425.

[7] ELGAMMAL, A. ISMAIL, M. A. Techniques for Language Identification for Hybrid Arabic-English Document Images, in: Proc. of International Conference on Document Analysis and Recognition. 2001, pp. 1100 - 1104.

[8] MOALLA, I. ELBAATI, A. ALIMI, A. A. BENHAMADOU, A. Extraction of Arabic Text from Multilingual Documents, in: Proc. of International Conference on Systems, Man and Cybernetics, 2002.

[9] JAWAHAR, C. V. KUMAR, M. N. S. S. K. P. KIRAN, S. S. R. A Bilingual OCR for Hindi–Telugu Documents and Its Applications, in: Proc. of International Conference on Document Analysis and Recognition. 2003. pp. 1 - 5.

[10] AQAB, S. TARIQ, M. U. Handwriting Recognition using Artificial Intelligence Neural Network and Image Processing. International Journal of Advanced Computer Science and Applications. Vol. 11. No. 7. pp 137-146. 2020. Disponível em: <https://pdfs.semanticscholar.org/2590/ccf9445b96ef6ec17de8adae603f420517e2.pdf>. Acesso em: 15 mar. 2023.

[11] SHAMIM, S. M. et al. Handwritten Digit Recognition using Machine Learning Algorithms. Global Journal of Computer Science and Technology: DNeural & Artificial Intelligence. Vol. 18. No. 1. 2018. Disponível em: <https://computerresearch.org/index.php/computer/article/view/1685/1669>. Acesso em: 15 mar. 2023.

[12] BUNKE, H. ROTH, M. SCHUKAT-TALAMAZZINI, E. G. Off-line cursive handwriting recognition using hidden markov models. Pattern Recognition, Vol. 28. No. 9. pp 1399 - 1413. 1995. Disponível em: <https://www.sciencedirect.com/science/article/pii/003132039500013P>. Acesso em: 15 mar. 2023.

[13] GUPTA, A. Handwritten A-Z. Kaggle, 2018. Disponível em: <https://www.kaggle.com/datasets/ashishguptajiit/handwritten-az>. Acesso em: 15 mar. 2023.