# Máquina de Vetores de Suporte Aplicada à Base de Dados MNIST

Vítor Gabriel Reis Caitité, Augusto Vilaça Alves Machado

***Abstract*—Este relatório tem o intuito de apresentar um estudo feito sobre uma base de dados fornecida pelos professores da disciplina de Reconhecimento de Padrões, lecionada na UFMG. O objetivo deste é utilizar os conhecimentos adquiridos ao longo do semestre para resolver um problema prático de reconhecimento de imagens. Neste relatório, serão apresentados resultados obtidos utilizando-se duas técnicas que foram aprendidas durante a disciplina. A primeira delas foi o *PCA (Principal Component Analysis)*, que é utilizado com o intuito de diminuir as dimensões do problema dado, definindo-se suas componentes principais que possuem maior impacto no problema de classificação. Basicamente, essa técnica foi utilizada para encontrar um meio de condensar a informação contida em várias variáveis originais (*pixels*) em um conjunto menor de variáveis estatísticas (componentes) com uma perda mínima de informação. Após isso, foi utilizada uma *SVM (Support Vector Machine*. Tal técnica foi utilizada sobre os dados pré-processados pelo *PCA*, para gerar os resultados do problema proposto no trabalho, ou seja, classificar qual número está desenhado na imagem com base nos valores dos seus *pixels*.**

***Index Terms*—PCA, SVM, Classificador, Reconhecimento de Padrões**

## I. Descrição do Problema

O Problema proposto, consiste em gerar um modelo preditivo que seja capaz de identificar padrões nos dados fornecidos pelos professores. Os dados se tratam de imagens contendo algarismos numéricos, escritos por diversas pessoas diferentes. A partir desses dados, o modelo gerado deve ser treinado, e ser capaz de inferir, para novas entradas, qual algarismo é encontrado na imagem. Para gerar esse resultado, devem ser aplicadas, uma ou mais técnicas, que foram aprendidas ao longo do semestre na disciplina de Reconhecimento de Padrões.

## II. Revisão Bibliográfica

Os dados do problema resolvido neste artigo foram retirados do *dataset* MNIST. De acordo com [1] o banco de dados MNIST contém 60.000 dígitos manuscritos no conjunto de treinamento e 10.000 dígitos manuscritos no conjunto de teste.

Como esta base é mundialmente conhecida e amplamente utilizada para *benchmark* no campo de inteligência artificial, é possível encontrar diversos trabalhos que resultaram em classificadores para o problema. Alguns do diferentes classificadores comprovados nesta base de dados apresentaram taxa de erro de 1,00 a 0,42%, como pode ser visto na Tabela 1.

TABLE I
Taxa de reconhecimento de diferentes classificadores

| Métodos | % de erro | Referêcias |
|---|---|---|
| Reduced set SVM poly 5 | 1.0 | [2] |
| LeNet-5 (neural net) | 0.95 | [2] |
| Virtual SVM poly 9 [distortions] | 0.8 | [2] |
| LeNet-5 [distortions] (neural net) | 0.8 | [2] |
| Boosted LeNet-4 [distortions] (neural net) | 0.7 | [2] |
| Shape matching + 3-NN | 0.63 | [3] , [4] |
| Proposed classifier LIRA_grayscale (neural net) | 0.61 | [1] |
| SVC-RBF_grayscale | 0.42 | [5] |

## III. Metodologia

Para resolução do problema foram utilizados os métodos *PCA (Principal Component Analysis)* e *SVM (Support Vector Machine)*.

Assim como visto em sala de aula, o PCA é um método muito utilizado para identificar padrões em dados, visto que ele coloca em evidência suas similaridades e diferenças. Foi decidido por utilizar esse método como uma estratégia de pré-processamento, para redução da dimensionalidade do problema. Ao se utilizar o PCA, o número de componentes principais se torna o número de variáveis consideradas na análise, contudo as primeiras componentes são as mais importantes já que explicam a maior parte da variação total. Com isso, geralmente é possível utilizar apenas estas componentes de maior importância para resolução do problema. Assim consegue-se reduzir a dimensão dos dados de entrada, o que é algo importante para diminuir a complexidade computacional do problema e facilitar o desenvolvimento de um classificador eficiente.

Para aplicá-lo, foi utilizada a função PCA da biblioteca sklearn.decomposition [6] da linguagem Python. A princípio, não era de conhecimento da dupla quantas componentes eram realmente importantes para a resolução do problema, devido a isso foi plotado o gráfico *Explained Variance vs PCs*, mostrado na Figura 1, para todas as 784 componentes dos dados.

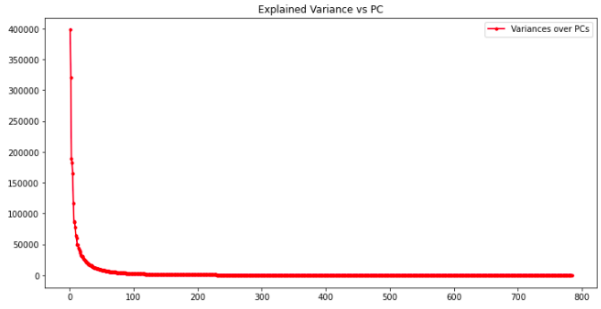


Fig. 1. Gráfico de Variância vs Componentes Principais para todas as componentes do problema

Com esse gráfico plotado, foi possível verificar que para os

dados de entrada, por volta de apenas 30 a 90 componentes dos 784 presentes nos dados de entrada realmente tinham uma significância maior no cálculo de saída do modelo, fazendo assim com que fosse possível diminuir as dimensões de entrada em mais de 10 vezes. Então foi gerado um novo gráfico com o número de componentes variando somente até 60 (valor escolhido para a dimensão reduzida do problema). Tal gráfico é mostrado na Figura 2.

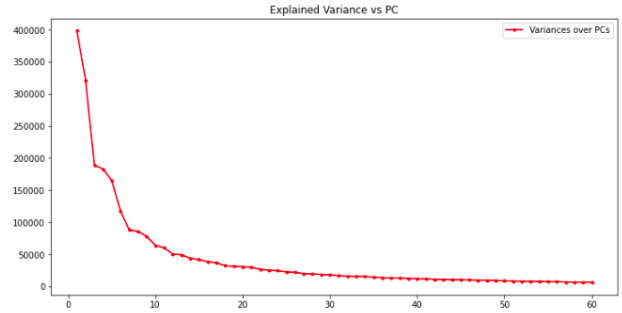


Fig. 2. Gráfico de Variância x Componentes Principais para 60 primeiras componentes do problema

Como já visto no primeiro gráfico, podemos perceber que realmente após as 60 principais componentes escolhidas, as demais não causam grande impacto nos cálculos de saída do modelo.

Com os dados já pré-processados (com redução das dimensões), partiu-se para a aplicação do modelo da SVM, que retornaria o vetor com os pesos que poderia propriamente calcular a saída do modelo e ser capaz de prever os números presentes em cada imagem de entrada.

Como visto na teoria, para aplicar o modelo SVM em nossos dados, precisamos definir à priori, o parâmetro de penalização C e a função de kernel a ser utilizada no cálculo de seus pesos. Para a função de kernel, foi escolhida a função RBF, em vista de seus resultados nos trabalhos desenvolvidos ao longo da disciplina e também em vista dos resultados que foram aferidos em alguns artigos que foram estudados pela dupla em problemas semelhantes ao proposto no Trabalho Prático.

Já para definir o melhor C possível a ser aplicado, foi utilizada a função GridSearchCV da biblioteca sklearn.model. Essa função recebe possíveis valores de C e para cada valor gera-se um modelo e calcula a acurácia média de saída utilizando a técnica de validação cruzada (mais especificamente foi utilizado a técnica *10-fold cross validation*). Foi definido variar o C de 1 a 100 para aferição dos resultados. Ao final, bastou-se plotar o gráfico, mostrado na Figura 3, com as médias encontradas em cada C. A partir dele pôde-se descobrir qual seria o melhor valor de C para se escolher na aplicação do modelo aos dados e gerar os resultados para a entrega final.

É importante apontar que, todos esses testes com a SVM foram feitos particionando-se os dados do arquivo trainReduzido.csv em amostras de teste e treino, de acordo com a técnica de validação cruzada. Para a entrega dos resultados finais, todos os dados deste arquivo foram utilizados para treino, visto que os dados a serem "testados" seriam propriamente os a serem entregados.

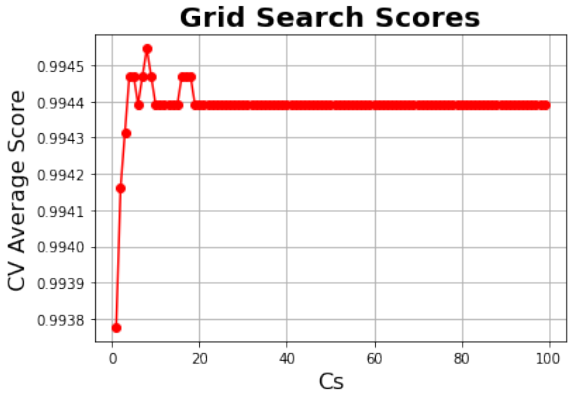


Fig. 3. Gráfico com as acurácias médias para cada parâmetro C.

## IV. Descrição dos Dados

Foram disponibilizados alguns arquivos que foram retirados do dataset MNIST. Essa base de dados contêm imagens em escala de cinza de algarismos numéricos desenhados à mão por diversas pessoas diferentes, de zero a nove. Um exemplo de imagem contida nessa base de dados pode ser vista na Figura 4. O arquivo disponibilizado para treino e validação se encontra no formato .csv e possui cerca de 13000 linhas. Cada linha possui 786 colunas. A primeira coluna é referente ao identificador de cada uma das imagens dentro do *dataset* completo. A segunda coluna representa o *label* daquela imagem, ou seja, qual algarismo numérico ela ilustra. Por fim, as últimas 784 colunas, representam os *pixels* que definem tal imagem, de tal forma que essas colunas podem ser enxergadas como uma matriz 28 x 28 que define a imagem final. Basicamente, uma notação matemática para essas colunas pode ser definida pela expressão abaixo:

$$x = i * 28 + j \tag{1}$$

Onde i e j, são números inteiros entre 0 e 27, de tal forma que os pixels possam ser definidos nas linhas e colunas dentro da matriz. Ou seja, se pegarmos como exemplo o pixel da coluna 40, sabemos que ele se encontra na segunda linha e décima terceira coluna da matriz 28 x 28.

Um detalhe importante é que apesar de o *dataset* MNIST possuir algarismos de 0 a 9, o subgrupo de imagens separadas para este problema contém apenas os algarismos 1, 5, 6 e 7.

## V. Experimentos e Resultados

Utilizando os dados de treinamento e aplicando o modelo SVM (com kernel RBF e C=8) aos dados pré-processados pelos PCA (com redução de dimensionalidade para 60), foi realizado um experimento utilizando a técnica *10-fold cross validation*. Nesse ensaio foi possível obter uma acurácia média de 99,46%. Além disso, o excelente resultado obtido pelo classificador também pode ser notado na matriz de confusão mostrada na Figura 5. Para geração dessa matriz de confusão foram separados 30% dos dados de treinamento para validação.

A partir dos resultados acima, conseguiu-se validar a eficiência do modelo. A partir de então ele foi aplicado aos

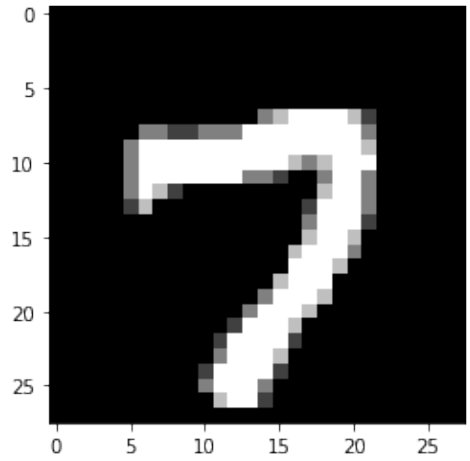

Fig. 4. Exemplo de imagem presente na base de dados MNIST.

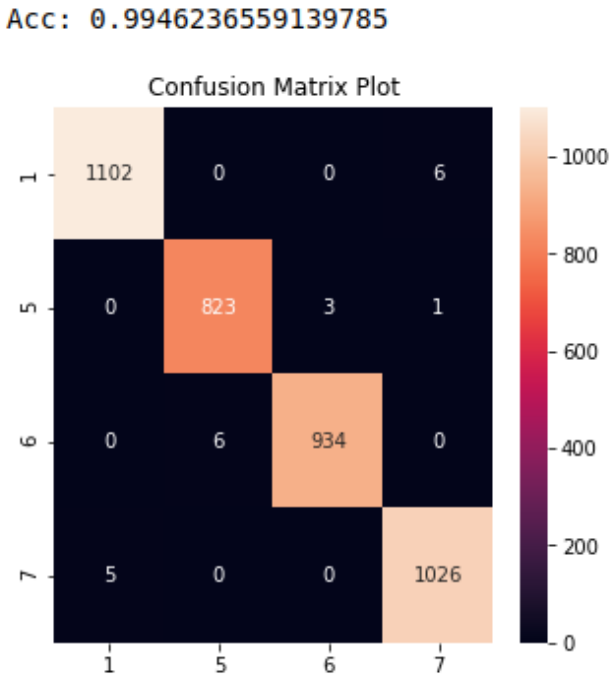


Fig. 5. Matriz de confusão do modelo SVM aplicado aos dados de validação.

dados de teste disponibilizados e os resultados de classificação encontrados foram enviados para avaliação no formato especificado.

## VI. Conclusão

Com este trabalho foi possível aplicar 2 dos principais algoritmos vistos furante a disciplina de Reconhecimento de Padrões. O primeiro deles, o PCA, foi utilizado no intuito de redução da dimensionalidade do problema. Já o segundo, um classificador SVM com kernel RBF, foi utilizado para a classificação das imagens da base de dados MNIST. Os resultados encontrados com a aplicação desses métodos ao problema foram considerados positivos, visto que se obteve acurácias médias de teste maiores que 99% ao se aplicar a validação cruzada nos dados oferecidos para treino e validação.

## Acknowledgment

Os autores gostariam de agradecer aos professores Antônio de Pádua Braga e Frederico Gualberto Ferreira Coelho por todo conteúdo ensinado durante a disciplina Reconhecimento de Padrões.

## References

[1] E. Kussul and T. Baidyk, "Improved method of handwritten digit recognition tested on mnist database," *Image and Vision Computing*, vol. 22, no. 12, pp. 971–981, 2004, proceedings from the 15th International Conference on Vision Interface. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0262885604000721

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2344, 1998.

[3] S. Belongie, J. Malik, and J. Puzicha, "Matching shapes," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 1. IEEE, 2001, pp. 454–461.

[4] ——, "Shape matching and object recognition using shape contexts," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[5] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition using state-of-the-art techniques," in *Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition*. IEEE, 2002, pp. 320–325.

[6] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.