Deteção e Classificação de Doenças da Batata

Jeffri Murrugarra[1], **Mac Kevin Cabanillas**[1]

[1] Universidade de São Paulo – Campus de São Carlos (SP) – –ICMC
São Carlos – Brazil

ju.jeffri.v@usp.br, mac.kevin.c.e@usp.br

Abstract. *The objective of this project is to classify the data of a potato leaf between 3 classes (leaf in good condition, leaf affected by Alternaria solani, leaf affected by Phytophthora infestans), this to detect pests and to avoid the loss of potato crops, from a set of data that are descriptors of the images of the potato leaf, also wiil be do a reduction in the dimensionality of the data to evaluate the characteristics of the space.*

Resumo. *O objetivo deste projeto é classificar os dados de uma folha da batata entre 3 classes (folha em bom estado, folha afetada por Alternaria solani, folha afetada por Phytophthora infestans), isso para detectar pragas e evitar a perda de colheitas de batata,a partir de um conjunto de dados que são descritores das imagens da folha de batata,tamben foi fieto uma redução na dimensionalidade dos dados para avaliar as características do espaço.*

1. Introdução

A agricultura familiar na área andina desempenha um papel estratégico na produção de alimentos, especialmente aquelas que fazem parte da base alimenticia, como é o caso da batata, que é considerada o quarto alimento mais consumido no mundo. No entanto, a disponibilidade deste alimento fica comprometida quando as pragas e doenças que afetam este cultivo causam perdas nos rendimentos e na qualidade dos produtos antes e depois da colheita.

O cultivo de batatas é afetado por numerosos organismos que, sob certas condições, causam danos econômicos. Patógenos de batata afetam o rendimento e a qualidade das culturas e são insetos, fungos, bactérias, nematóides e vírus que danificam as folhas; alterar o crescimento das plantas; eles causam apodrecimento ou malformação e afetam a aparência comercial e a qualidade culinária dos tubérculos. Cada ano o mercado de produção de batata sofre perda devido a infestação de pragas, uma estimativa da Organização das Nações Unidas para Alimentação e Agricultura (FAO), do 20% até 40% de perda na produção. Portanto, o gerenciamento oportuno deles é vital para uma produção mais eficiente, nesse sentido, a prevenção é altamente recomendada. O reconhecimento de pragas é um dos princípios fundamentais dessa abordagem que determina o controle que deve ser adotado.

Tradicionalmente, a detecção dessas pragas é feita por um especialista humano, mas nem todos os produtores podem cobrir as despesas. Assim, o objetivo deste projeto e a detecção e classificação de doenças da batata causadas por pragas. Por estas razões, para reduzir danos causados por pragas e doenças do cultivo da batata, é necessário implementar este programa que classifica folhas de batata detecção de pragas a tempo, aumentando a produtividade e sem organismos causadores de pragas e doenças tornar mais agressivo.

2. Fundamentação Teórica

2.1. BackPropagation

2.1.1. Principio do BackPropagation

BackPropagation é um algoritmo de rede de aprendizado supervisionado, que emprega um ciclo de propagação-adaptação de duas fases. Uma vez aplicado um padrão à entrada da rede como um estímulo, ele se propaga da primeira camada através das camadas superiores da rede, até gerar uma saída. O sinal de saída é comparado com a saída desejada e um sinal de erro é calculado para cada uma das saídas.

As saídas de erro são propagadas para trás, a partir da camada de saída, para todos os neurônios na camada oculta que contribuem diretamente para a saída. No entanto, os neurônios da camada oculta recebem apenas uma fração do sinal total do erro com base aproximadamente na contribuição relativa que cada neurônio contribuiu para a saída original. Esse processo é repetido, camada por camada, até que todos os neurônios da rede recebam um sinal de erro que descreva sua contribuição relativa ao erro total. Com base no sinal de erro percebido, onde os pesos de conexão de cada neurônio são atualizados, fazer a rede convergir para um estado que permita classificar corretamente todos os padrões de treinamento.[de Mello 2018]

A importância deste processo consiste em sua capacidade de adaptar os pesos dos neurônios intermediários para aprender a relação existente entre um conjunto de padrões dados como exemplo e suas saídas correspondentes. Após o treinamento, quandoé apresentado um padrão de entrada arbitrário que contém ruído ou está incompleto, os neurônios na camada oculta da rede responderão com uma saída ativa se a nova entrada contiver um padrão que se assemelhe ao recurso que os neurônios individuais aprenderam a reconhecer durante o treinamento. Inversamente, as unidades das camadas ocultas tendem a inhibir sua saída se o padrão de entrada não contiver a característica a reconhecer, para a qual foram treinadas.

2.1.2. Regra de Aprendizagem

O algoritmo Backpropagation para redes multicamadas é uma generalização do algoritmo LMS, ambos algoritmos realizam sua tarefa de atualização de pesos e ganhos com base no erro quadrático médio.

Para realizar este processo, a topologia de rede deve primeiro ser definida, isto é: número de neurônios na camada de entrada, que depende do número de componentes do vetor de entrada, número de camadas ocultas e número de neurônios em cada um deles. eles, número de neurônios na camada de saída que depende do número de componentes do vetor de saída ou padrões de destino e funções de transferência necessárias em cada camada.

Cada padrão de treinamento é propagado através da rede e seus parâmetros para produzir uma resposta na camada de saída, que é comparada com os padrões de destino ou saídas desejadas para calcular o erro na aprendizagem, esse erro marca a maneira mais apropriada para atualizar os pesos e ganhos que no final do treinamento produzirão uma

resposta satisfatória a todos os padrões de treinamento, isso é alcançado minimizando o erro quadrático médio em cada iteração do processo de aprendizagem.

A rede Backpropagation trabalha sob aprendizado supervisionado e, portanto, precisa de um conjunto de treinamento que descreva cada saída e seu valor de saída.[de Mello 2018]

2.1.3. Algoritmo backpropagation

- **Etapa de operação**

  Quando um padrão é apresentado p entrada $X^p : x_1^p, ..., x_i^p, ..., x_N^p$, ele é transmitido através dos pesos w_{ij} da camada de entrada para a camada oculta. Os neurônios nesta camada intermediária transformam os sinais recebidos aplicando uma função de ativação, fornecendo assim um valor de saída. Isto é transmitido através dos pesos v_{kj} para a camada de saída, onde aplicando a mesma operação como no caso anterior, os neurônios desta última camada fornecem a saída da rede. Este processo é resumido a seguir: A entrada total ou líquida que um neurônio oculto recebe j, net_j^p, é:

  $$net_j^p = \sum_{i=1}^{N} w_{ji} x_i^p + \theta_j$$

  Onde θ_j é o limiar do neurônio que é considerado como um peso associado a um neurônio fictício com um valor de saída igual a 1.
  O valor de saída do neurônio oculto j, y_j^p é obtido pela aplicação de uma função $f(\cdot)$ em sua entrada líquida:

  $$y_j^p = f(net_j^p)$$

  Da mesma forma, a entrada prevista recebida por um neurônio de saída k, net_k^p é:

  $$net_k^p = \sum_{j=1}^{H} v_{kj} y_j^p + \theta_k$$

  Finalmente, o valor de saída do neurônio de saída k, y_k^p, é:

  $$y_k^p = f(net_k^p)$$

- **Fase de aprendizagem**
  Na etapa de aprendizado, o objetivo é minimizar o erro entre a saída obtida pela rede e a saída desejada pelo usuário antes da apresentação de um conjunto de padrões chamado de grupo de treinamento.
  Assim, o aprendizado em redes de retropropagação é supervisionado. A função de erro que se destina a minimizar para cada padrão p é dada por:

  $$E^p = \frac{1}{2} \sum_{k=1}^{M} (d_k^p - y_k^p)^2$$

Onde d_k^p é a saída desejada para o neurônio de saída xxxx antes da apresentação do padrão k. A partir desta expressão, uma medida de erro geral pode ser obtida por:

$$E = \sum_{p=1}^{P} E^p$$

A base do algoritmo de retropropagação para a modificação de pesos é a técnica conhecida como gradiente decrescente.
Como E^p é uma função de todos os pesos da rede, gradiente E^p é um vector igual a derivada parcial de E^p respeito a cada um dos pesos. O gradiente toma a direção que determina o aumento mais rápido do erro, enquanto a direção oposta, ou seja, a direção negativa, determina a diminuição mais rápida do erro. Portanto, o erro pode ser reduzido ajustando cada peso na direção:

$$-\sum_{p=1}^{p} \frac{\partial E^p}{\partial w_{ji}}$$

Um perigo que pode surgir ao usar o método gradiente gradiente é que o aprendizado converge para um mínimo local. No entanto, o problema potencial de mínimos locais raramente ocorre em dados reais.
Em um nível prático, a maneira de modificar os pesos de uma maneira iterativa é aplicar a regra da cadeia à expressão de gradiente e adicionar uma taxa de aprendizado η. Assim, em um neurônio de saída:

$$\Delta v_{kj}(n+1) = -\eta \frac{\partial E^p}{\partial v_{kj}} = \eta \sum_{p=1}^{P} \delta_k^p y_j^p$$

Onde

$$\delta_k^p = (d_k^p - y_k^p) f'(net_k^p)$$

E n indica a iteração.
Em um neurônio oculto:

$$\Delta w_{ji}(n+1) = \eta \sum_{p=1}^{P} \delta_j^p x_i^p$$

Onde

$$\delta_j^p = f(net_j^p) \sum_{k=1}^{M} \delta_k^p v_j^k$$

Pode ser visto que o valor de erro ou delta associado a um neurônio escondido j, é determinada pela soma dos erros cometidos em neurônios de saída k recebendo como entrada a saída do neurônio escondido j. Portanto, o algoritmo também é chamado de propagação do erro para trás.

Para a modificação dos pesos, a atualização é feita após a apresentação de todos os padrões de treinamento. Essa é a maneira usual de proceder e é chamada de aprendizado em lote ou em lote ou padrão.

Para acelerar o processo de convergência de pesos, Rumelhart et al. (1986) sugeriu adicionar um fator de momento, α, que leva em conta a direção do incremento obtido na iteração anterior:

$$\Delta v_{kj}(n+1) = \eta \left(\sum_{p=1}^{P} \delta_k^p y_j^p \right) + \alpha \Delta v_{kj}(n)$$

2.2. Principal component analysis(PCA)

2.2.1. Descripción del PCA

A técnica de análise de componentes principais consiste em analisar um conjunto de dados de entrada, que contém diferentes observações descritas por múltiplas variáveis independentes ou dependentes e cujas relações entre si não precisam ser conhecidas. O objetivo principal é o de reduzir o tamanho do conjunto de dados de entrada tentaram manter o máximo de informação possível para ser analisado mais facilmente e nas fases posteriores, como classificadores ou regressores, pode simplificar critérios de decisão.[Sanchez 2012]

O PCA realiza uma transformação linear dos dados em um novo sistema de coordenadas ortogonais. Os vetores de projeção de dados no novo espaço são as direções de variância máxima dos dados de entrada. Enquanto as novas variáveis resultantes da projeção dos dados de entrada nos vetores de projeção serão chamadas de componentes principais (”Main Component”, PC). Nesse novo sistema de coordenadas, os componentes principais são ordenados automaticamente de acordo com a variação da projeção de dados, ou seja, de acordo com a quantidade de informações que eles contêm. Finalmente, a dimensão dos dados resultantes no novo espaço pode ser reduzida eliminando os componentes principais que possuem uma variância menor, ou seja, que fornecem menos informações.[Sanchez 2012]

Os dados de entrada serão organizados em uma matriz $X \in \mathbb{R}^{\mathrm{m}\times\mathrm{n}}$, onde m é o número de variáveis de entrada e n é o número de observações. Enquanto os dados são projectados numa matriz $Y \in \mathbb{R}^{\mathrm{p}\times\mathrm{n}}$, com p saída variável e n observações, em que o número de variáveis de saída é menos do que o $(p < m)$ entrada. Cada vetor $x_i = [x_1, x_2, ..., x_m]^T$ conterá todas as variáveis de entrada associadas a uma observação i e $y_i = [y_1, y_2, ..., y_p]^T$ conterá as variáveis de saída.

A tarefa do PCA é encontrar uma matriz $U \in \mathbb{R}^{\mathrm{m}\times\mathrm{p}}$ que transforme linearmente o espaço dos dados de entrada em X em outra matriz Y com um número menor de variáveis. Este processo é realizado por meio de uma combinação linear das variáveis originais para que sejam projetadas nas direções de máxima variância dos dados, e assim a quantidade máxima de informação é conservada. Essas direções máximas de variação serão definidas pelos vetores de projeção p, u_k, que serão encontrados nas colunas de U.

$$X = \begin{bmatrix} x_1 \\ x_2 \\ ... \\ x_m \end{bmatrix} \xrightarrow[Y=U^T X]{PCA} Y = \begin{bmatrix} y_1 \\ y_2 \\ ... \\ y_m \end{bmatrix}, (m > p)$$

Durante este processo, as variáveis de saída $y_k = u_k^T X$ serão chamadas de "componentes principais", enquanto os vetores de projeção u_k serão os coeficientes ou cargas do componente principal k.

Em princípio, U pode ter até $p = m$ vetores de projeção, um para cada variável. No entanto, essa dimensão geralmente será reduzida mantendo apenas as projeções com a maior variação, o que resulta em perda de informações. Para minimizar essa perda, o algoritmo PCA consegue concentrar a maior parte da variação em um número reduzido de variáveis do novo espaço. Dessa maneira, você pode eliminar essas variáveis do espaço de saída que contém menos variação e, assim, perder a menor quantidade possível de informações. A premissa é que a maior quantidade de informação está contida no menor número possível de variáveis.[Sanchez 2012]

2.2.2. Formulação do PCA como método para maximizar a variância

Em termos gerais operação PCA está focada em encontrar, a partir de um conjunto de dados X entrada com variáveis m, um vetor de pesos u_1 capaz de projetar este conjunto de dados sobre o rumo da X máxima variância, onde u_1 é um vetor formado por variáveis m. Quando tivermos u_1, o próximo passo é encontrar outro vetor u_2 ortogonal com u_1 que retenha a variação máxima possível. Isso continuará até obter o p-ésimo vetor com a variância máxima sendo ortogonal com u_1,u_2,...,u_{p-1}. A projeção dos dados de entrada nestes vetores dá origem aos componentes principais: $y_1 = u_1^T X$,$y_2 = u_2^T X$,...,$y_k = u_k^T X$. Embora seja possível continuar até a obtenção do m-ésimo componente principal, normalmente só será necessário calcular o primeiro p onde a maior parte da variância será projetada. O resto das variáveis será descartado, o que causa uma perda de informação, mas será mínima (em termos de variação).

Seja U a matriz composta dos vetores de projeção p, X a matriz com os dados de entrada, então a matriz Y formada pelos componentes principais p, você obterá:

$$Y = U^T X$$

Se quer preservar a quantidade máxima de informações nas variáveis de Y, enquanto elas são ortogonais. Uma maneira de atingir esse objetivo é maximizar a covariância dos principais componentes, ou seja:

$$\max_U \ C_{yy}$$

$$sujeto \ a \ U^T U = I$$

Para realizar a concentração da informação no menor número possível de componentes e que a informação não é repetido, a transformação tem de assegurar que a variável resultando no novo espaço são ortogonais, porque a força de restrição $U^T U = I$. Dessa forma, quando eliminarmos os atributos menos significativos, perderemos a menor quantidade de informação. Para encontrar a matriz U que resolve o problema dado, primeiro devemos desenvolver sua expressão de C_{YY} da seguinte maneira:

$$C_{YY} = \frac{1}{n} Y Y^T = \frac{1}{n} (U^T X)(U^T X)^T = \frac{1}{n} U^T X X^T U = U^T (\frac{1}{n} X X^T) U = U^T C_{XX} U$$

A partir dessa expressão, podemos redefinir o objetivo do PCA como:

$$\max_U \quad tr\{U^T C_{XX} U\}$$

$$sujeto \ \ a \ \ U^T U = I$$

Agora temos um problema de otimização com restrições, portanto, para resolver essa expressão vamos aplicar multiplicadores de Lagrange. Primeiro de tudo, é definido uma matriz de multiplicadores de Lagrange e temos que

$$L_p = Tr\{U^T C_{XX} U\} - (I - U^T U)\Lambda$$

$$\frac{d\, L_p}{d\, U} = 2 C_{XX} U + (-2U)\Lambda = 0$$

$$C_{XX} U = U \Lambda$$

Chegamos a um problema de autovalores e autovetores sobre a matriz de covariância dos dados.

Desta forma, o PCA pode ser resolvido como um problema de autovetores e autovalores. Os autovetores de C_{XX} fornecem as direções de desvio máximas de X. Portanto, se u_k for escolhido como o autovetor k de C_{XX}, então $u_k^T X$ será o componente principal k. Os autovetores são ordenados de maior a menor variância e o autovalor correspondente a cada autovetor indicará a variância do componente principal calculado a partir dele. Usualmente, no cálculo dos autovetores e autovalores, utiliza-se a técnica matemática da decomposição em valores singulares (”Singular Value Decomposition”, SVD). A solução do SVD de X é:

$$X = U \Sigma V^T$$

onde se sabe que U e V são matrizes ortonormais, e $U = [u_1 u_2 ... u_m]$ tem os autovetores de $X X^T \times \Sigma$ em suas colunas é uma matriz diagonal com valores singulares de $X X^T$. Cada valor singular é definido como a raiz quadrada do autovalor correspondente ($\sigma_i \equiv \sqrt{\lambda_i}$).[Sanchez 2012]

3. Desenvolvimento

3.1. Preprocesamento

Os dados da batata estão em três arquivos diferentes con nome(BatataH.dat, batataE.dat , batataH.dat) , cada arquivo representa uma classe.

Os dados contidos nos arquivos são descritores de imagens representando as folhas de batata, os descritores foram: (Contraste, Correlação, Energia, Homogeneidade, Mídia, Desvio Padrão, Assimetria, Entropia, Curtose).

Existem 152 amostras da classe Healthy potato, 182 amostras da classe da batata com peste moderada, 290 batatas com peste tardia no momento da leitura das amostras foram balançadas com 152,depois todas foram agrupadas em uma única matriz de dados e foram normalizadas ter média 0 e variância 1

3.2. PCA

Com os dados normalizados,se trabalhou - se com os dados do espaço X,e foram encontrados os vetores que maximizam a variância, aplicando-se a matriz de covariância, depois extraímos os autovalores e autovetores, para ter uma representação de como cada vetor influência na variância. , com isso podemos escolher um subconjunto K de componentes que maximiza a variância do espaço, para minimizar a perda das características originais do espaço, e com esses componentes produzir uma nova matriz que representa nosso espaço reduzido a K dimensões

3.3. Rede Multi - Camada

3.3.1. Arquitectura

A arquitectura usada foi:

- Camadas de entrada: 8
- Camadas Intermedias: 5
- Camadas de saídaa: 3
- Função de ativação: Sigmoide

3.3.2. Treinamento

- Termo de Aprendizado: 0.1
- Termo Momentum: 0.0
- Numero de Ciclos: 5 - 100
- Algoritmo de Aprendizado: BackPropagation
- Forma de Treinamento: Padrão

3.4. Metricas - Matrix de confusão

Formulas:

- $Precisao = \frac{TP}{TP+FP}$
- $Recall = \frac{TP}{TP+FN}$
- $F1 - medida = \frac{2xprecisaoxrecall}{precision+recall}$
- $accuracy = \frac{TP+TN}{TP+FN+TN+FP}$

		Predicción	
		Positivos	**Negativos**
Observación	**Positivos**	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Figura 1. Confusion Matrix

4. Conclusão

4.1. Resultados do PCA

PCA	Autovalores	Responsabilidade na variância
PCA1	8.09	89.38%
PCA2	0.62	7.05%
PCA3	0.22	2.48%
PCA4	0.07	0.77%
PCA5	0.01	0.19%
PCA6	0.66 x 10^{-2}	0.07%
PCA7	0.20 x 10^{-2}	0.02%
PCA8	0.03 x 10^{-2}	0.04 x 10^{-1}%

Tabela 1. Análisis das componentes principais

A partir do PCA, observamos que os dois primeiros componentes representam juntos 96% da variância dos dados.Os dados foram reconstruídos a partir dos dois componentes mais representativas

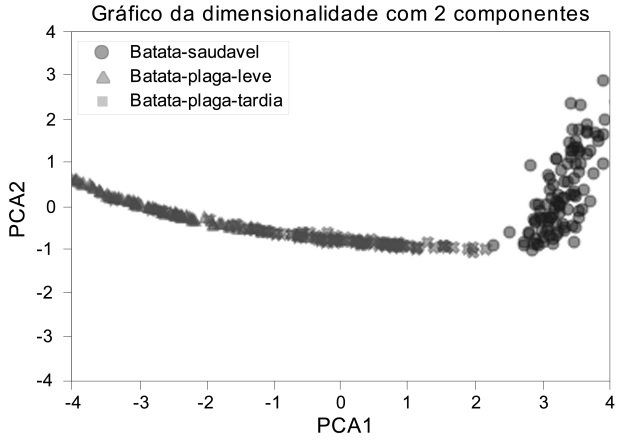

Figura 2. PCA com 2 componentes

É observável que a classe da batata saudável é linearmente separável das outras 2, isto pode ser útil no caso em que o propósito seja apenas distinguir entre uma batata em bom estado e outra em má estado,para fazer esta tarefa só precisamos de um perceptron.

4.2. Resultados de Teste

Para o teste o conjunto de dados particionado com a tecnica HoldOut 70% (319 instâncias) conjunto de treinamento e 30 % (137 instâncias) conjunto de teste.

	Clase 1	Clase 2	Clase 3	Promedio
Precisión	100%	96.07%	76.08%	90.72%
Recall	86.95%	90.74%	94.59%	90.76%
f1	93.02%	93.33%	84.33%	90.23%

Tabela 2. Metricas de Avaliação

Tasa de acerto no treinamento 87.46% — Tasa de acerto no test 90.51%

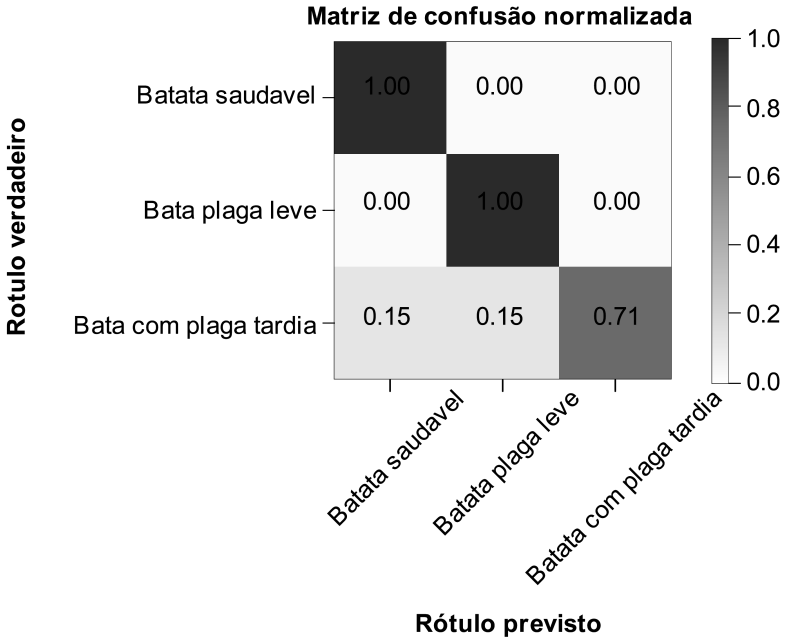

Figura 3. Matrix de confusão

5. References

Referências

de Mello, R. F. (2018). *A Practical Approach on the Statistical Learning Theory*. Springer, 1th edition.

Sanchez, A. (2012). Analisis de componentes principales: Versiones dispersas y robustas al ruido impulsivo. *Thesis*.