Universidade Federal de Minas Gerais
Departamento de Ciência da Computação

DCC057 Mineração de Dados

Trabalho Prático 3

Alexander Thomas Mol Holmquist

17 de março de 2021

1 Introdução

Como dito na proposta, este trabalho tem como objetivo principal de negócio encontrar um modelo que classifica as figuras de regiões de câncer com alta precisão ($F - SCORE \geq 95\%$) como "benigno"ou "maligno". Sendo assim, é um problema de classificação binário de imagens.

Já no início do projeto, percebeu-se a ampla falta de experiência por parte do autor com imagens. Muitas das propostas e recomendações delineadas na proposta do trabalho se mostraram espúrias ou mesmo estúpidas. Por exemplo, o plano inicial do projeto constava uma recomendação de extrair métricas interessantes, como média, variância, assimetria e curtose dos dados, como meio de descrição. Mas isto é claramente impraticável em um banco de dados de imagens, que contém milhares e milhares de atributos (no presente projeto, especificamente 150528 atributos).

Assim, este projeto é nada mais que a história de um tremendo fracasso por parte do autor. Sendo assim, contudo, tem-se que o aprendizado foi tanto maior, pois é com erros deste tipo que se adapta à diversidade da disciplina de mineração de dados. Cada categoria de dados requer um tratamento diferente, devido a uma grande gama de características que podem apresentar. Este fato não é evidente para um iniciante na área, e não se torna evidente a não ser por meio de prática com cada uma das categorias existentes de dados.

Este projeto, como delineado pela proposta, tem seu curso altamente guiado pelo CRISP-DM. Um plano de projeto inicial foi desenvolvido, e foi seguido durante o projeto. Ao seguir o plano, porém, percebeu-se que a especialização das propostas do CRISP-DM para o caso presente se deu de forma muito deformada, novamente por conta da inexperiência do autor. Isto é, não foi bem sucedido em montar um plano inicial a partir das recomendações do CRISP-DM. Mesmo assim, ter esse direcionamento foi de grande ajuda, até mesmo para tornar o mais claro possível as falhas cometidas, e para que estas não venham a ser repetidas.

A seguir é dado um passo-a-passo da execução do plano inicial do projeto, com reflexões e resultados, para cada etapa.

2 Execução do Plano

2.1 Fase 1 — Carregamento dos dados

2.1.1 Passo 1 — Carregar os dados em um jupyter notebook

Jupyter notebook é a ferramenta utilizada em grande parte deste trabalho. Este primeiro passo já levou a alguns empecilhos quanto aos dados colhidos [1]. Surgiram perguntas da forma: "como decodificar os arquivos de imagens?", "qual o tipo em python3 dos dados que se deve utilizar?", "como lidar com o fato de que as imagens são compostas por três dimensões de atributos. Isto é, como transformá-los em uma matriz 2D?", "como lidar com a alta dimensionalidade dos dados?".

Por fim, a maioria dos problemas foi resolvida simplesmente utilizando a biblioteca *skimage* [2] (posteriormente se verificou que com a biblioteca matplotlib obter-se-ia resultados semelhantes). Além disso, a observação de que o valor de cada pixel deveria ser armazenado utilizando somente um byte foi essencial, já que a matriz utilizando os 8 bytes (comportamento padrão em Numpy) ocuparia três gigabytes de memória, desnecessariamente.

Vale notar que no fim do projeto foi necessário utilizar uma ferramenta ferramenta alternativa, o "Google Colab" [3], devido à falta de poder de computação da máquina local (ver

Seção 2.4.

2.1.2 Passo 2 — Verificar pressupostos fundamentais

Este passo se refere principalmente a fatos como corrupção dos dados. Isto é verificado automaticamente pela biblioteca utilizada no carregamento das imagens.

2.1.3 Passo 3 — Verificar a qualidade dos dados

Aqui, a preocupação principal são valores faltantes. Novamente, a biblioteca selecionada para carregamento das imagens lida com esse problema automaticamente, de tal forma que as etapas posteriores não são afetadas.

2.1.4 Passo 4 — Verificar necessidade de estrutura especial

É necessário utilizar alguma estrutura de dados especial para os dados? A resposta simples encontrada foi não. No fim do projeto, constatou-se que os dados poderiam ser utilizados no mesmo formato entregue *skimage*, isto é, cada imagem como um vetor *Numpy* de formato (224, 224, 3).

2.1.5 Passo 5 — Produzir relatório

Como reflexão, é importante relatar aqui, que inicialmente várias técnicas foram buscadas, até o ponto em que uma péssima decisão foi tomada. A implementação de amostrar as imagens pela média de janelas de 4 x 4 pixels foi realizada. Em estágios subsequentes, esta decisão foi obviamente revogada, pois desconsidera a morfologia das imagens, e também o caráter do problema. Um fato que se aprendeu durante o projeto é que, na detecção de câncer de pele, são exatamente os detalhes que mais importam, como pontilhados vermelhos ou negros. Portanto, tais detalhes não devem ser ignorados.

Outro problema encontrado foi a tentação à redução da dimensionalidade dos dados. Com dimensionalidade tão grande quanto 150528, o autor se viu constrangido a tentar métodos como o PCA, que se baseiam na decomposição espectral da matriz de dados. Como tais métodos precisam calcular os vetores singulares esquerdos, a computação leva tempo demasiadamente longo. Além disso, devido à heterogeneidade das imagens, seria necessário extrair mais de cem componentes principais para preservar 95% da variância original. Portanto, não foi realizada redução de dimensionalidade.

O relatório de carregamento de dados se encontra no caminho "doc/relatorios/fase-1/relatorio-carregamento-dados.pdf" do repositório do projeto [4].

2.2 Fase 2 — Descrição dos dados

2.2.1 Passo 1 — Obter estatísticas

Como mencionado na introdução, esta recomendação por parte do plano inicial não é realista. Não se faz isto para bancos de imagens, simplesmente porque não é sensato focar em um atributo específico, por exemplo a cor vermelha do pixel de número 10234, e analisar sua variância, etc.

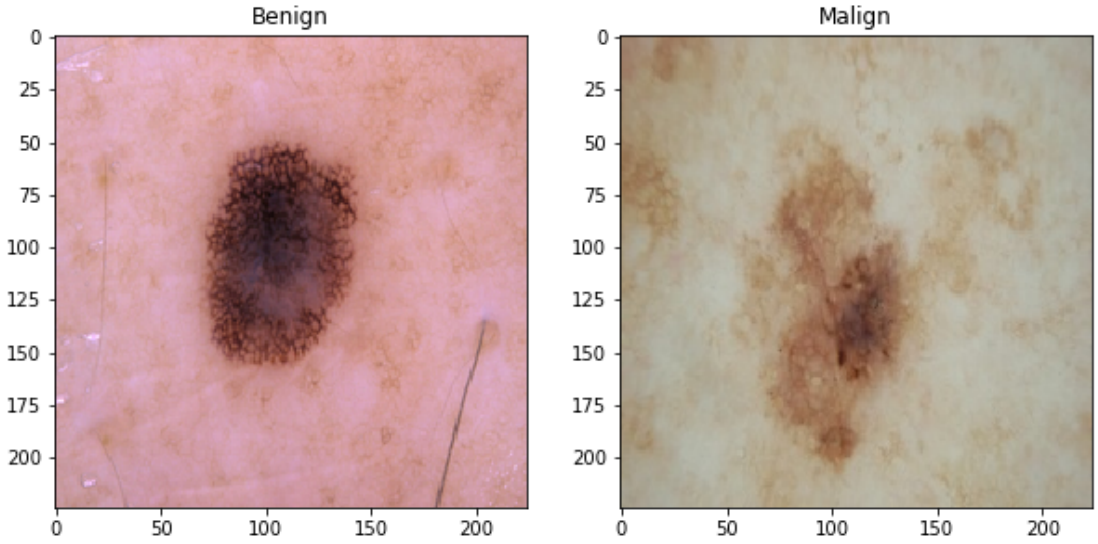

Figura 1: Um exemplo de câncer benigno e maligno

2.2.2 Passo 2 — Exibir gráficos

Também não é realista. A dimensionalidade não foi reduzida e portanto não foram gerados tais gráficos. O autor havia tomado como pressuposto, na construção do plano inicial, que é sempre uma boa ideia reduzir a dimensionalidade dos dados para visualização, mas estava errado.

2.2.3 Passo 3 — Explorar os dados mais a fundos

Esta recomendação foi muito feliz, devido a sua generalidade. Antes de mostrar alguns aspectos dos dados, é bom notar que, durante a elaboração de um plano para um projeto de mineração de dados, é necessário que as minucias dos passos propostos sejam descritas, mas também não pode faltar espaço para caminhos inesperados. Caso contrário, os executores do plano se verão forçados a ignorá-lo, fronte à realidade enfrentada.

A seguir, exporemos algumas características dos dados que ajudarão o leitor a se familiarizar.

- Quantidade de imagens de câncer benigno: 1800
- Quantidade de imagens de câncer maligno: 1497
- Particionamento K-dobras: 5 dobras utilizadas (3 para treino, 1 para validação, 1 para teste).

A Figura 1 mostra um exemplo de câncer benigno, ao lado de um caso maligno.

2.2.4 Passo 4 — Produção do relatório

O relatório de descrição de dados pode ser encontrado no caminho “doc/relatorios/fase-2/relatorio-descricao-dados.pdf” do repositório.

2.3 Fase 3 — Preparação dos dados

Esta fase foi em grande parte ignorada, porque se percebeu que não seria necessária muita preparação para a modelagem. Além disso, aqui já estava decidido que o único modelo a ser considerado seria o de rede neural convolutiva, que requer somente manipulações básicas. Esta decisão será explicada com mais detalhes na Seção 2.4.

2.3.1 Passo 1 — Solucionar problemas de qualidade dos dados

Não foi necessário (ver Seção 2.1.3.)

2.3.2 Passo 2 — Comprimir os dados

Veja Seção 2.1.5 para uma tentativa de compressão de dados. Posteriormente, decidiu-se que não seria feita qualquer tipo de compressão.

2.3.3 Passo 3 — Criação de atributos derivados

Isso foi tentado com rotação, translação, *zoom*, e outras transformações nos dados. Porém, como só é possível verificar quais transformações são beneficiais após o treinamento, este passo não deveria ter sido colocado na seção 3. Aqui cabe a reflexão: talvez toda a fase três deveria ter sido integrada com a fase 4, já que diversas premissas aqui consideradas só podem ser testadas uma vez que se tem um ambiente de modelagem pronto. A preparação dos dados para o modelo, e a aplicação do modelo em si são fortemente interligadas.

2.3.4 Passo 4 — Normalização

Realizou-se a normalização dos dados para o intervalo de zero a um. Ou seja, divide-se os valores dos pixels por 255.

2.3.5 Passo 5 — Verificar se está tudo pronto

Novamente, não é possível determinar isto durante esta fase do projeto. Talvez este passo seria melhor colocado na Seção 2.4.

2.3.6 Passo 6 — Produção de relatório

Não foi produzido o relatório de preparação dos dados, por conta da falta de tempo (isto é uma consideração real, mesmo em ambientes de negócio).

2.4 Fase 4 — Modelagem

Os modelos de árvores de decisão e árvores aleatórias foi rejeitado. Não parece ser um bom classificador para imagens. O modelo de rede neural, por outro lado, é comumente utilizado neste meio. Inicialmente, havia um grande receio na utilização deste modelo, devido a limitações de poder computacional da máquina local utilizada para este trabalho. Todavia, foi descoberto o recurso "Google Colab" [3], que oferece gratuitamente poder de computação para alguns processos em máquinas remotas, com performance muito maior que a referida máquina local.

Desta forma, o modelo de rede neural foi viabilizado pela disposição deste recurso. Outro recurso utilizado foi a biblioteca Keras [5] da linguagem Python, que possibilitou que o trabalho de implementação dos algoritmos de redes neurais, que levaria talvez dias a mais, não precisasse ser feito. Outra vantagem desta biblioteca é que o código é adaptado para computação altamente paralela.

2.4.1 Passo 1 — Montar esquema de testes

O conjunto de dados, que originalmente estava particionado entre treino e teste, seguindo o modelo 5-dobras, foi incrementalmente particionado, para incluir um conjunto de validação. O resultado foi, portanto, uma tripartição de proporções (3, 1, 1) para treino, validação e teste, respectivamente.

2.4.2 Passo 2 — Selecionar uma técnica de modelagem

Isto foi feito antes mesmo de chegar a este passo, porque ficou óbvio que árvores de decisão não seriam uma boa escolha, enquanto redes neurais se mostraram mais promissoras. Além disso, o trabalho da universidade de Stanford mencionado na proposta [6] utiliza precisamente um modelo de redes neurais convolutivas para atingir resultados impressionantes.

2.4.3 Passo 3 — Decidir metaparametros

Alguns metaparâmetros merecem atenção. Em geral se seguiu o modelo fornecido em [7].

- **Estrutura da rede neural**, melhor explicada através do código abaixo, escrito em Python:

```
img_input = layers.Input(shape=(224, 224, 3))

# Camadas internas. Sao utilizados filtros 3x3 de
#   convolucao, e selecao dos valores maximos em
#   uma janela 2x2(funcao MaxPooling2D).
x = layers.Conv2D(16, 3, activation='relu')(img_input)
x = layers.MaxPooling2D(2)(x)
x = layers.Conv2D(32, 3, activation='relu')(x)
x = layers.MaxPooling2D(2)(x)
x = layers.Conv2D(64, 3, activation='relu')(x)
x = layers.MaxPooling2D(2)(x)
x = layers.Flatten()(x)
x = layers.Dense(512, activation='relu')(x)

# A classe com maior probabilidade na funcao
#   'sigmoid' eh selecionada.
output = layers.Dense(1, activation='sigmoid')(x)

model = Model(img_input, output) # Modelo final
```

- **Algoritmo de otimização**: RMSProp [8], para decidir automaticamente a taxa de aprendizado.

- **Tamanho de lote** para o algoritmo de otimização: 20 instâncias (para permitir parelização).

2.4.4 Passo 4 — Descrever o modelo, e relatar reflexões

O modelo de redes neurais aprende especificidades dos dados a cada camada, construindo uma compreensão das características essenciais do banco de dados. O algoritmo de otimização faz várias iterações sobre os dados. Cada passagem pelo banco tomou de 100 a 124 segundos. Devido a este processo repetitivo, o algoritmo pode tentar otimizar a função de perda de acordo com uma mesma imagem várias vezes, e portanto existe o risco de *overfitting*.

Para contornar o problema de *overfitting*, foi tentada a criação de instâncias artificiais para os dados de treinamento, com distorções, translações, etc. Porém, devido à alta variabilidade dos dados gerados, o algoritmo RMSProp não conseguiu chegar a convergência.

Em ambos os casos - com instâncias artificiais e sem, a acurácia máxima do modelo no conjunto de validação foi cerca de 82%. A Figura 2 mostra a variação da acurácia de acordo com o andamento do treino da rede neural.

2.4.5 Passo 5 — Argumentar a qualidade do modelo

Os parâmetros do modelo foram ajustados de acordo com recomendações para a área. O modelo parece ser suficientemente atrativo, considerando o tamanho do banco de dados disponível.

Certamente, seria possível melhorar os resultados sem precisar trocar os dados. Mesmo assim, é provável que não seja possível passar de 90% de acurácia, a não ser que se use um modelo pre-treinado em um banco de dados maior, como entrada (*stacking*).

2.4.6 Passo 6 — Analisar sucesso do modelo

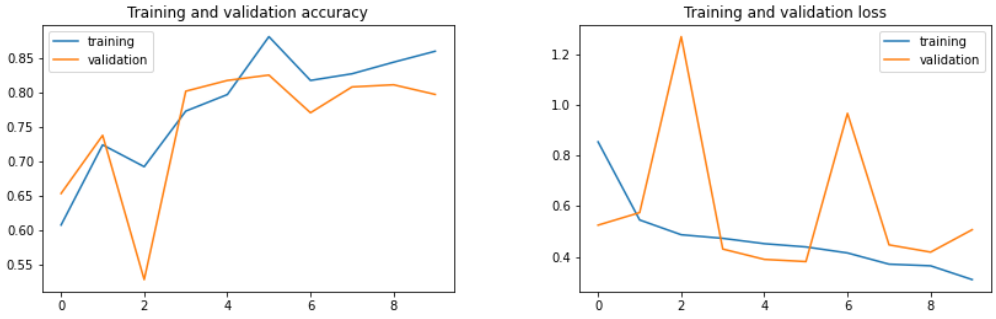

Figura 2: Acurácia e perda do modelo (sem instâncias artificiais) de redes neurais convolutivas em função do número de “epochs”.

O plano inicial incluía o objetivo de alcançar um F-SCORE de 95%. Isto não foi possível (com acurácia de 82%, não é nem necessário calcular a revocação). Se o autor tivesse uma compreensão sequer básica da área aprendizado de máquina (classificação) para imagens enquanto elaborava o plano, certamente teria estabelecido um limite menos estrito. Tem-se, de qualquer forma, o resultado: acurácia de 82%.

Não é satisfatório, nem de acordo com o objetivo principal do projeto, nem do ponto de vista da aplicabilidade. Para conseguir atingir o objetivo inicial, seria necessário um modelo mais bem ajustado em seus metaparâmetros, e certamente um banco de dados maior (talvez duas ordens de grandeza maior).

2.5 Fase 5 — Avaliação

2.5.1 Passo 1 — Sucesso dos objetivos de negócio

O plano inicial do projeto havia estabelecido dois objetivos. O principal já foi tratado na Seção 2.4.6. O secundário, que é distinguir quais são as características determinantes que diferenciam um câncer benigno de um maligno, falhou por conta do limite de tempo.

Para continuar o presente trabalho, uma possível alternativa é obter mais dados para construir um modelo mais preciso. Isto, porém, não teria muita utilidade científica, considerando que já existem modelos praticamente perfeitos para o mesmo propósito [6].

2.5.2 Passo 2 e 3 — Revisão do processo de mineração de dados

Uma das motivações deste projeto, apesar de não constar como objetivo de negócio, foi explorar a utilidade de, em uma ambiente profissional, se ter um pequeno projeto como este, para fins de treinamento.

Durante o processo de construção do modelo, através da descoberta de fatos específicos da área (delineados anteriormente neste relatório), o autor se deparou com a veracidade da hipótese. É muito útil passar por um processo de treinamento prático em uma área semelhante (neste caso aprendizado de máquina com imagens), antes de entrar em um projeto maior.

De fato, foi descoberto que este processo é amplamente conhecido na indústria, e tem o nome "experimentation cycle" [9]. Acontece que mineração de dados, geralmente relacionada com a disciplina Ciência de Dados, tem muita semelhança com o método científico, e portanto requer exploração e descoberta empíricas de maneira iterativa.

Finalmente, ficou claro durante este projeto, que o plano inicial foi mal formulado. Como ele depende fortemente da experiência prévia dos seus elaboradores, e o autor não tinha qualquer, não é estranho que isso tenha acontecido. Um ensinamento prático a ser guardado para projetos futuros é que a fase 1 talvez poderia ter sido integrada com a fase 2, e o mesmo vale para as fases 3 e 4.

Referências

[1] https://www.kaggle.com/fanconic/skin-cancer-malignant-vs-benign. Acessado em 6 de março de 2021.

[2] https://scikit-image.org/. Acessado em 17 de março de 2021.

[3] https://colab.research.google.com/. Acessado em 17 de março de 2021.

[4] https://github.com/Yowgf/MD-TP3. Acessado em 17 de março de 2021.

[5] https://keras.io/. Acessado em 17 de março de 2021.

[6] https://www.nature.com/articles/nature21056. Acessado em 7 de março de 2021.

[7] https://developers.google.com/machine-learning/practica/image-classification. Acessado em 17 de março de 2021.

[8] https://en.wikipedia.org/wiki/Stochastic_gradient_descentRMSProp. Acessado em 17 de março de 2021.

[9] https://towardsdatascience.com/ai-ml-practicalities-the-cycle-of-experimentation-fd46fc1f3835. Acessado em 17 de março de 2021.