EA072 – Inteligência Artificial em Aplicações Industriais (2s2020)
Exercícios Computacionais 2 – ECp2
Atividade Individual – Peso 4 – Data de Entrega: 27/11/2020

Questão 6)

Esta questão aborda conceitos de interpretabilidade em aprendizado de máquina, evidenciando dois paradigmas de grande relevância e focando num subconjunto de técnicas, dentre várias possibilidades que vêm sendo propostas na literatura. A busca por modelos de aprendizado interpretáveis sempre esteve na alça de mira das pesquisas em aprendizado de máquina, mas até recentemente havia uma predominância de foco em acurácia e se defendia a existência de um compromisso entre acurácia e interpretabilidade, de modo que a busca por mais acurácia tenderia a gerar modelos menos interpretáveis, da mesma forma que modelos mais interpretáveis não conseguiriam atingir os mesmos níveis de acurácia que modelos "caixa-preta". As últimas conquistas voltadas para interpretabilidade em aprendizado de máquina, no entanto, demonstram que acurácia e interpretabilidade podem caminhar juntas, trazendo maior confiabilidade para aplicações de inteligência artificial.

Tomando o mesmo problema de classificação de dados da base MNIST, o objetivo desta questão é analisar a interpretabilidade de uma RNA treinada. Para tanto, vamos utilizar o código a seguir para treinar uma RNA.

```
import keras

mnist = keras.datasets.mnist
(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
x_train, x_test = x_train / 255.0, x_test / 255.0

model = keras.models.Sequential()
model.add(keras.layers.Conv2D(32, kernel_size=(3, 3),
 activation='relu',input_shape=(28, 28, 1)))
model.add(keras.layers.Conv2D(64, (3, 3), activation='relu'))
model.add(keras.layers.MaxPooling2D(pool_size=(2, 2)))
model.add(keras.layers.Dropout(0.25))
model.add(keras.layers.Flatten())
model.add(keras.layers.Dense(128, activation='relu'))
model.add(keras.layers.Dropout(0.5))
model.add(keras.layers.Dense(10, activation='softmax'))

model.get_config()

model.compile(optimizer='adam',
 loss='sparse_categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train, epochs=5)
evaluation = model.evaluate(x_test, y_test)

model.save('mnist_model.h5')
```

6a)

Utilize a biblioteca *innvestigate* [1] para analisar as predições da RNA e comparar diferentes métodos de explicação [3]. Uma explicação é a coleção de características do domínio interpretável, que contribuíram para que uma dada amostra produzisse uma decisão (de classificação ou regressão, por exemplo) [3]. Utilize os seguintes métodos da biblioteca *innvestigate*: *Gradient*, *SmoothGrad*, *DeepTaylor*, *LRPAlphaBeta*, *LRPEpsilon*, *LRPZ*. Um código que pode servir de ponto de partida é fornecido a seguir. Escolha 6 imagens aleatórias de 3 classes distintas para analisar, portanto teremos 2 imagens por classe. Escolha parâmetros adequados para os métodos *LRPAlphaBeta* e *LRPEpsilon* (use os mesmos parâmetros para as 6 imagens escolhidas). A Figura 1 apresenta um modelo de como os resultados devem ser exibidos (não é necessário incluir os letreiros). Analise o resultado obtido.

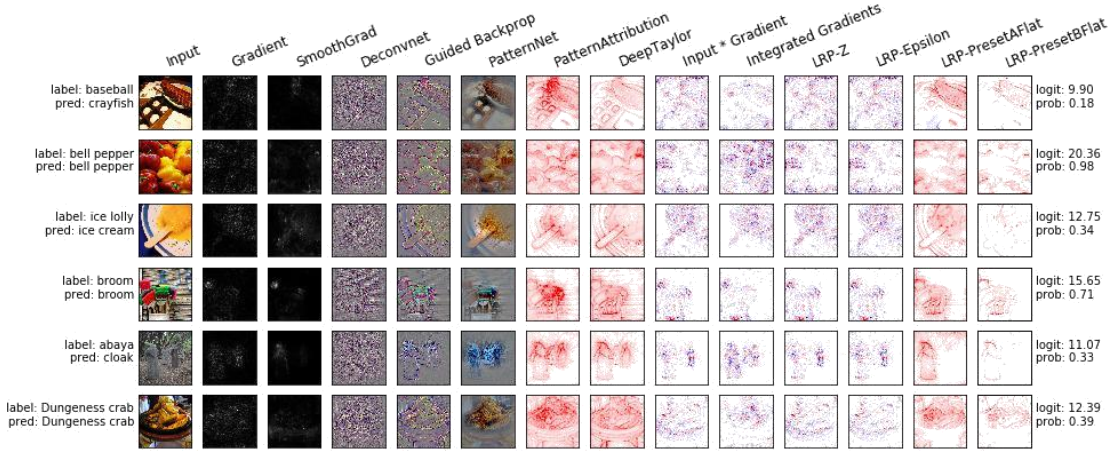

Figura 1 – Exemplo de como exibir os resultados da Questão 7a. Fonte: *https://github.com/albermax/innvestigate*

```
import keras
import innvestigate
import matplotlib.pyplot as plot

mnist = keras.datasets.mnist
(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
x_train, x_test = x_train / 255.0, x_test / 255.0

model = keras.models.load_model('mnist_model.h5')
model_wo_sm = innvestigate.utils.keras.graph.model_wo_softmax(model)

imagem = x_test[0:1]
plot.imshow(imagem.squeeze(), cmap='gray', interpolation='nearest')

analyzer = innvestigate.analyzer.LRPEpsilon(model=model_wo_sm,
 epsilon=1)
analysis = analyzer.analyze(imagem)
plot.imshow(analysis.squeeze(), cmap='seismic',
 interpolation='nearest')
```

6b)
Utilize a biblioteca keras-vis [2] para interpretar as predições da RNA para cada uma das 10 classes. Neste caso, procuramos por um padrão de entrada que produza uma resposta máxima do modelo para uma métrica de interesse [3]. Um código que pode servir de ponto de partida é fornecido a seguir. Escolha valores adequados para os parâmetros que determinam os pesos da *Total variation* e *L-p norm*. Veja dicas em:

https://github.com/raghakot/keras-vis/blob/master/examples/mnist/activation_maximization.ipynb

Plote os gráficos das imagens obtidas para cada uma das 10 classes e analise o resultado.

```
import keras
from vis.visualization import visualize_activation
from vis.utils import utils
import matplotlib.pyplot as plot

model = keras.models.load_model('mnist_model.h5')
layer_idx = utils.find_layer_idx(model, 'dense_2')
model.layers[layer_idx].activation = keras.activations.linear
model = utils.apply_modifications(model)

filter_idx = 9
img = visualize_activation(model, layer_idx,
 filter_indices=filter_idx, input_range=(0., 1.), verbose=True,
 max_iter=1000, tv_weight=1., lp_norm_weight=0.)
plot.imshow(img.squeeze(), cmap='seismic', interpolation='nearest')
```

Referências bibliográficas

[1] M. Alber, S. Lapuschkin, P. Seegerer, M. Hägele, K. T. Schütt, G. Montavon, W. Samek, K.-R. Müller, S. Dähne, P.-J. Kindermans, **innvestigate neural networks!**, preprint arXiv:1808.04260, 2018.

[2] R. Kotikalapudi and contributors, **keras-vis**. https://github.com/raghakot/keras-vis, 2017.

[3] G. Montavon, W. Samek, K.-R. Müller, **Methods for interpreting and understanding deep neural networks**, Digital Signal Processing, vol. 73, pp. 1-15, 2018.

Questão 7)
Aprendizado por reforço do tipo *Deep Q-Learning* para definir percursos em labirintos.
Estude o notebook **Q7.ipynb**, procurando compreender o que está sendo feito em cada trecho de código. Para tanto, é muito relevante acompanhar atentamente as explicações em https://www.samyzaf.com/ML/rl/qmaze.html. Atividades práticas:

1. Execute o notebook para os dois labirintos propostos (execuções independentes, usando um de cada vez), apresentando os resultados do treinamento até a convergência, além de 2 percursos para cada caso de estudo (com condições iniciais distintas), obtidos com a rede neural treinada.
2. Para esses dois casos de estudo, apresente os 4 valores da função Q-valor para 3 estados representativos do ambiente, produzidos pela rede neural treinada.

Questão 8)
O treinamento de redes neurais adversárias foi proposto em 2014:
I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, **Generative adversarial nets**, Advances in Neural Information Processing Systems, pp. 2672-2680, 2014.
no contexto de síntese de máquinas generativas, e foi considerada a ideia mais interessante em aprendizado de máquina da última década, tendo já recebido inúmeras contribuições e extensões na literatura. As GANs conseguem conduzir o processo de treinamento a desempenhos nunca antes vislumbrados em aprendizado de máquina, permitindo afirmar que as redes neurais generativas podem capturar "qualquer" distribuição exibida pelos dados de treinamento, habilidade esta que torna "ilimitado" o potencial de aplicação dessas máquinas de aprendizado. Evidentemente, a exploração de todo este potencial requer muita memória e muito processamento, algo que não será exigido neste curso. Inicialmente, esta atividade envolve a execução completa do treinamento de uma GAN para reproduzir dígitos manuscritos, a partir da base MNIST. Para isso, vamos utilizar a biblioteca Keras-GAN disponível em:

https://github.com/eriklindernoren/Keras-GAN

O objetivo é conseguir executar o exemplo disponível em:

https://github.com/eriklindernoren/Keras-GAN/blob/master/gan/gan.py

que já foi organizado no notebook **Q8.ipynb**. O código executa o treinamento por 30.000 épocas e salva exemplos de imagens a cada 1000 épocas. Execute o código e exiba: a primeira imagem obtida, e imagens obtidas após 1000, 10000, 20000 e 30000 épocas. Por fim, adapte o código ou use algum outro código capaz de treinar uma máquina generativa para alguma outra base de imagens da literatura, apresentando os resultados de forma similar ao caso da base MNIST.

Questão 9)
Processamento de Linguagem Natural – *Word Embedding*, usando a implementação GENSIM do Word2Vec e uma base de reviews de hotéis pelo mundo, chamada OpinRank (https://kavita-ganesan.com/entity-ranking-data/#.XwbHbOWSlPY).
O primeiro passo é instalar o GENSIM:

conda install -c conda-forge gensim

A seguir, copie o arquivo GZIP fornecido pelo professor em algum diretório de trabalho. O *default* no notebook é C:/EA072.
Explique como funcionam Word2Vec e t-SNE (*t-Distributed Stochastic Neighbor Embedding*).
Execute o notebook **Q9.ipynb** e comente / interprete os resultados obtidos.

Observações gerais:

- Os entregáveis devem ser enviados para o e-mail [vonzuben@dca.fee.unicamp.br] com Assunto [EA072 - ECp1] até 23h59 do prazo final. Aguarde um retorno por e-mail com a confirmação de recebimento.