IA353 – Redes Neurais (1s2021)
Exercícios de Fixação de Conceitos 1 – EF 1
Atividade Individual – Peso 5
Data de entrega dos resultados solicitados: 27/04/2021

Questão 1) Síntese de modelos lineares para classificação de padrões

Caso de estudo inicial:

Base de dados MNIST (http://yann.lecun.com/exdb/mnist/), ver Figura 1 abaixo, a qual contém dígitos manuscritos rotulados em 10 classes (são os dígitos de '0' a '9'), sendo 60.000 amostras para treinamento e 10.000 amostras para teste (os dados de teste não devem ser empregados em nenhuma fase do processo de síntese do classificador). Cada imagem de entrada contém 784 pixels (no intervalo [0,255], correspondente a níveis de cinza), visto que a dimensão é 28 × 28 pixels. Considere que a classe 10 corresponde ao dígito '0'.

Dados disponibilizados: Visando facilitar a implementação em GNU Octave / Matlab, o professor está fornecendo arquivos *.mat para os dados de treinamento e teste, contendo 2 matrizes cada.

Observação: Está subentendido um classificador com uma saída por classe, de tal modo que a classe indicada é aquela associada à saída de maior valor numérico.

Arquivo de treinamento [data.mat]: Uma matriz *X* de dimensão 60.000 × 784, com a imagem de um dígito por linha da matriz, **com cada pixel normalizado no intervalo [0,1]** e uma matriz *S* de dimensão 60.000 × 10, com as saídas desejadas, sendo '1' para a classe do dígito da linha correspondente da matriz *X* e '0' para as demais classes.

Arquivo de teste [test.mat]: Mesma configuração, mas para 10.000 amostras e matrizes *Xt* e *St*.

O professor também disponibiliza um programa [visual_MNIST.m] para visualização da imagem dos dígitos.

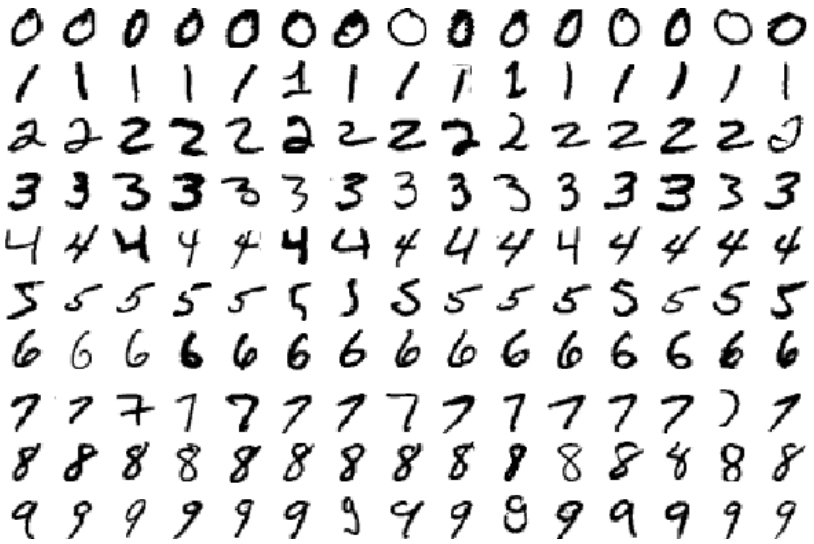


Figura 1 – Exemplos de imagens do conjunto de dados MNIST. Nos dados, a classe '0' é a última, e não a primeira, como na figura acima.

Parte prática – Etapa 1 (código disponível):

Como primeiro passo, inclua o seu RA na linha 11 do código fornecido pelo professor: programa [LC_MNIST.m]. Estando no ambiente de comando do GNU Octave / Matlab, você pode editar o código digitando: open('LC_MNIST.m')
Executando o programa [LC_MNIST.m], você vai obter um modelo linear de classificação para as 10 classes existentes, de tal modo que a saída para cada classe seja produzida como segue, já supondo que os 784 pixels foram empilhados formando um vetor de entrada, **contendo uma entrada fixa de polarização (*offset*) como primeira coluna da matriz *X***:

$$c_j = w_{0j} + w_{1j}x_1 + w_{2j}x_2 + \cdots + w_{784\,j}x_{784},\ j \in \{1,\cdots,10\}$$

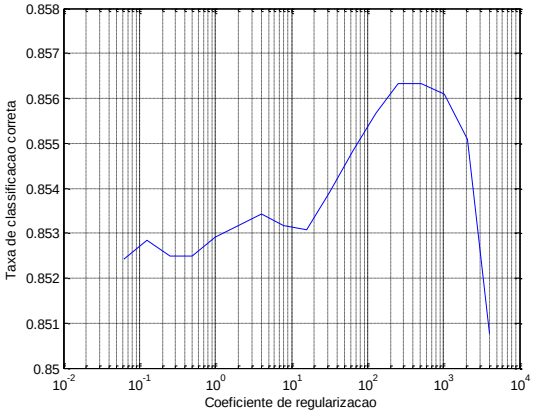
sendo que os parâmetros do modelo linear devem compor uma matriz *W* de dimensão 785 × 10 e devem ser obtidos de forma fechada, a partir de uma única expressão algébrica. Com isso, o coeficiente de regularização deve ser único para as 10 classes. Deve-se buscar um bom coeficiente de regularização, o qual tem que ser maior do que zero, pois a matriz de dados de entrada *X* não tem posto completo. Para tanto, tomar parte dos dados de treinamento como validação (Sugestão: 48.000 amostras para treinamento e 12.000 amostras para validação), adotando a técnica *holdout*, para poder implementar esta busca pelo melhor coeficiente de regularização, considerando como critério de desempenho para o classificador a taxa de erro de classificação. O coeficiente de regularização deve ser buscado iniciando pelo seguinte conjunto de valores candidatos:

$$\left\{2^{-10}, 2^{-8}, \cdots, 2^{0}, 2^{+2}, \cdots, 2^{+16}\right\}$$

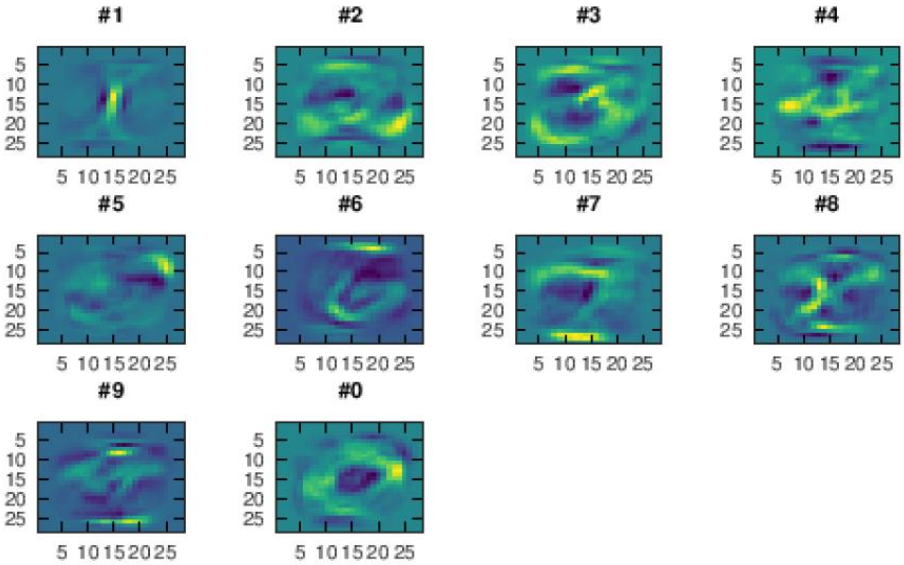
Este intervalo de busca é amplo, mas pode não ser adequado para todas as partições a serem produzidas. Um intervalo de busca adequado é aquele em que o gráfico da Figura 2, correspondente ao erro de classificação para os dados de validação, tem o formato de um arco côncavo, com o erro de classificação subindo, atingindo um pico e depois caindo.
Uma busca refinada é realizada automaticamente, após concluída a busca grossa. Ela faz uma busca linear no intervalo cujos extremos são os coeficientes imediatamente anterior e posterior àquele sugerido pela busca grossa.
Uma vez encontrado um valor adequado para o coeficiente de regularização (após terminadas as buscas grossa e refinada), o programa usa todas as 60.000 amostras de treinamento para sintetizar o classificador linear. Repare que o conjunto de dados de teste não foi usado em nenhum momento até este ponto.

Em seu relatório, a ser entregue até a data limite especificada, o(a) aluno(a) deve nomear o arquivo PDF como EF1_RA_do(a)_aluno(a).pdf, e deve incluir uma tabela com os valores dos 2 coeficientes de regularização encontrados na busca, antes e após o refinamento. Incluir também no relatório os dois gráficos semilog do desempenho dos classificadores junto aos dados de treinamento e os dois gráficos com escala linear junto aos dados de validação. Em seguida, o(a) aluno(a) deve incluir também os 10 mapas de calor (*heatmaps*) para os parâmetros de cada um dos 10 classificadores lineares.
Até este ponto, basta coletar as saídas produzidas. A partir de agora, é necessário que o(a) aluno(a) desenvolva algum código adicional, faça análises pertinentes e responda as perguntas a seguir, justificando suas respostas.

1. Apresente a matriz de confusão (https://en.wikipedia.org/wiki/Confusion_matrix) e também exemplos de dígitos classificados equivocadamente, ao menos de três classes diferentes. Quais são os dois dígitos mais desafiadores para o classificador linear?
2. Recorra ao conteúdo da seção 4 deste roteiro e responda: A adoção de um índice de regularização distinto para cada classe (em lugar de um único índice para todas as classes) pode conduzir a um ganho de desempenho?
3. A seguir, é apresentado um possível comportamento para o erro junto aos dados de validação, em função do coeficiente de regularização. Apresente o motivo pelo qual o gráfico não é côncavo, como se deveria esperar, apresentando algumas oscilações. Essas oscilações estão ilustradas mais à esquerda do gráfico, mas podem ocorrer em qualquer ponto.



4. No seu relatório, você vai incluir mapas de calor similares àqueles apresentados a seguir. Interprete esses resultados, indicando o que se pode extrair desses mapas de calor.



5. O professor também fornece um código adaptado para a base CIFAR10 (https://www.cs.toronto.edu/~kriz/cifar.html). Rode este código e apresente os resultados. São 5 *batches* que foram usados para treinamento (4) e validação (1). Se você tiver problemas de memória, trabalhe com menos *batches*. Em seguida, procure interpretar os mapas de cores apresentados ao final e associe esta interpretação com o baixo desempenho produzido pelos classificadores lineares.

Observação: Para resolver esta questão, sugere-se o emprego do ambiente de programação GNU Octave (https://www.gnu.org/software/octave/download.html), ou então o Matlab. No entanto, outras linguagens de programação podem ser adotadas, após realizar adaptações e reaproveitamento de código.

As seções a seguir apresentam conceitos fundamentais que servem de base para esta Questão 1 do EF1.

1 Regressão de quadrados mínimos

- Considere que você tenha à disposição um conjunto de *N* amostras de treinamento na forma: $\left\{\left(\mathbf{x}_i, y_i\right)\right\}_{i=1}^{N}$, onde $\mathbf{x}_i \in \Re^n$, *i*=1,...,*N*. Suponha também que $N > n$.
- A regressão de quadrados mínimos busca um vetor $\mathbf{w} \in \Re^n$ que minimiza:

$$J\left(\mathbf{w}\right) = \sum_{i=1}^{N}\left(\mathbf{x}_i^T\mathbf{w} - y_i\right)^2 .$$

- Fazendo $X = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1n} \\ 1 & x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{N1} & x_{N2} & \cdots & x_{Nn} \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$ e acrescentando um elemento de *offset* ao vetor **w**, constata-se que a regressão de quadrados mínimos requer a solução de um sistema linear sobredeterminado, na forma:

$$\min_{\mathbf{w}} \left\|X\mathbf{w} - \mathbf{y}\right\|_2^2 .$$

- Caso a matriz *X* tenha posto completo, a Seção 2 fornecerá a solução para este problema de otimização, na forma:

$$\mathbf{w} = \left(X^T X\right)^{-1} X^T\mathbf{y} .$$

- Caso a matriz *X* não tenha posto completo, a Seção 3 fornecerá a solução para este problema de otimização, na forma:

$$\mathbf{w} = \left(X^T X + \lambda I\right)^{-1} X^T\mathbf{y} \text{, com } \lambda > 0 .$$

2 Resolvendo sistemas lineares sobredeterminados

- O sistema $X\mathbf{w} = \mathbf{y}$, com $X \in \Re^{N\times(n+1)}$, $\mathbf{w} \in \Re^{(n+1)\times 1}$, $\mathbf{y} \in \Re^{N\times 1}$ e $N \geq (n+1)$, sendo *X* uma matriz de posto completo, tem garantia de solução exata apenas quando $N = \left(n+1\right)$. Na situação em que $N > \left(n+1\right)$, há mais equações do que incógnitas, criando a possibilidade de inconsistência entre algumas equações (regidas pelas linhas da matriz *X*), que não podem ser satisfeitas simultaneamente.
- A presença de inconsistência impede, portanto, que exista **w** tal que $X\mathbf{w} = \mathbf{y}$, mas não impede que se busque encontrar **w** que resolva o seguinte problema de programação quadrática:

$$\min_{\mathbf{w}} \sum_{i=1}^{N}\left(\mathbf{x}_i^T\mathbf{w} - y_i\right)^2 = \min_{\mathbf{w}} \left\|X\mathbf{w} - \mathbf{y}\right\|_2^2 = \min_{\mathbf{w}} \left(X\mathbf{w} - \mathbf{y}\right)^T\left(X\mathbf{w} - \mathbf{y}\right)$$

- A função-objetivo fica:

$$J(\mathbf{w}) = \left(X\mathbf{w} - \mathbf{y}\right)^T \left(X\mathbf{w} - \mathbf{y}\right) = \mathbf{w}^T X^T X\mathbf{w} - \mathbf{w}^T X^T \mathbf{y} - \mathbf{y}^T X\mathbf{w} + \mathbf{y}^T \mathbf{y} .$$

- Aplicando a condição necessária de otimalidade, que afirma que o gradiente se anula nos pontos extremos da função-objetivo, resulta:

$$\frac{dJ(\mathbf{w})}{d\mathbf{w}} = 2X^T X\mathbf{w} - 2X^T \mathbf{y} = 0 \Rightarrow X^T X\mathbf{w} = X^T \mathbf{y} .$$

- Se a matriz *X* for de posto completo, então $X^T X$ tem inversa, o que produz:

$$\mathbf{w} = \left(X^T X\right)^{-1} X^T \mathbf{y} .$$

- Esta equação representa a famosa solução de quadrados mínimos para um sistema linear de equações que não necessariamente admite solução exata.

3 Quadrados mínimos regularizados

- Tomando o mesmo cenário das Seções 1 e 2, é possível adicionar um termo de regularização, que penaliza o crescimento da norma do vetor **w**, produzindo o problema regularizado de regressão de quadrados mínimos, também denominado de *ridge regression*, na forma:

$$\min_{\mathbf{w}} \left\|X\mathbf{w} - \mathbf{y}\right\|_2^2 + \lambda \left\|\mathbf{w}\right\|_2^2, \ \lambda \geq 0 .$$

- Aplicando a condição necessária de otimalidade, como feito na Seção 2, obtém-se como solução ótima:

$$\mathbf{w} = \left(X^T X + \lambda I\right)^{-1} X^T \mathbf{y} .$$

- A definição de um valor para o parâmetro de regularização $\lambda \geq 0$ pode ser feita por técnicas de validação cruzada. Lembre-se que, quando a matriz *X* não tem posto completo, necessariamente deve-se tomar $\lambda > 0$.
- O modelo regularizado de regressão linear assume então a forma:

$$\mathbf{w}^T \mathbf{x} = \mathbf{y}^T X \left(X^T X + \lambda I\right)^{-1} \mathbf{x} .$$

- Note que é possível, e deve ser feito na atividade apresentada, expandir o vetor **y** com a saída desejada de modo a considerar tantas colunas quanto classes. Com isso, o vetor **w** também vai corresponder a uma matriz, com o mesmo número de colunas de **y**.

4 Emprego de múltiplos coeficientes de regularização

- A formulação de quadrados mínimos regularizados vale para um único classificador, conforme segue:

$$\min_{\mathbf{w}} \left\|X\mathbf{w} - \mathbf{y}\right\|_2^2 + \lambda \left\|\mathbf{w}\right\|_2^2, \ \lambda \geq 0 .$$

  onde $X \in \Re^{N \times 785}$, $\mathbf{w} \in \Re^{785 \times 1}$, $\mathbf{y} \in \Re^{N \times 1}$ e $\lambda \in \Re^{1 \times 1}$, sendo *N* o número de dados de entrada-saída para treinamento supervisionado.
- Como o roteiro da Questão 1 solicita um mesmo índice de regularização para todos os classificadores e, coincidentemente, como **w** e **y** aparecem sem nenhuma multiplicação à direita na solução ótima:

$$\mathbf{w} = \left(X^T X + \lambda I \right)^{-1} X^T \mathbf{y} ,$$

então os 10 classificadores podem ser obtidos simultaneamente, organizando as saídas desejadas de cada classificador como colunas da agora matriz $\mathbf{y} \in \Re^{N \times 10}$, produzindo uma solução ótima $\mathbf{w} \in \Re^{785 \times 10}$.

- Em termos conceituais, se está resolvendo o seguinte problema de otimização:

$$\min_{\mathbf{w}_1,...,\mathbf{w}_{10}} \sum_{c=1}^{10} \left\| X\mathbf{w}_c - \mathbf{y}_c \right\|_2^2 + \lambda \sum_{c=1}^{10} \left\| \mathbf{w}_c \right\|_2^2 , \ \lambda \geq 0 ,$$

que tem como soluções ótimas:

$$\mathbf{w}_c = \left(X^T X + \lambda I \right)^{-1} X^T \mathbf{y}_c , c = 1,...,10 .$$

- O que o professor pede no item 2 da Questão 1 é que o(a) aluno(a) avalie se o problema:

$$\min_{\mathbf{w}_1,...,\mathbf{w}_{10}} \sum_{c=1}^{10} \left\| X\mathbf{w}_c - \mathbf{y}_c \right\|_2^2 + \sum_{c=1}^{10} \lambda_c \left\| \mathbf{w}_c \right\|_2^2 , \ \lambda_c \geq 0, c = 1,...,10 ,$$

cujas soluções ótimas independentes são como segue:

$$\mathbf{w}_c = \left(X^T X + \lambda_c I \right)^{-1} X^T \mathbf{y}_c , c = 1,...,10 ,$$

pode levar a classificadores de melhor desempenho, sempre lembrando que, para cada dado de entrada a ser classificado, a classe escolhida é aquela de maior saída entre os 10 classificadores.

Questão 2) Síntese de modelos não-lineares, mas lineares nos parâmetros ajustáveis

Siga o mesmo fluxograma da Questão 1, tanto para a base MNIST como para a base CIFAR10, mas agora para uma máquina de aprendizado extremo (ELM). No caso da base MNIST, use 1000 neurônios na camada intermediária e pesos definidos aleatoriamente, com distribuição normal e desvio padrão 0,2. Não variar esses pesos, ou seja, usar a mesma camada intermediária para todos os experimentos solicitados e também adotar a mesma partição para os conjuntos de treinamento e validação, durante a busca pelo coeficiente de regularização. Para a base CIFAR10, use 1.500 neurônios na camada intermediária e desvio padrão de 0,01.

1. Encontre intervalos adequados para a busca grossa e apresente os valores dos 2 coeficientes de regularização encontrados, além dos 4 gráficos de evolução do erro.
2. Considerando os dados de treinamento, apresente a matriz de confusão e alguns exemplos de dígitos classificados de forma equivocada, ao menos de 3 classes distintas.
3. Mantendo a mesma partição entre conjuntos de treinamento e validação, o que você espera que ocorra com o coeficiente de regularização caso os neurônios da camada intermediária sejam inicializados com pesos sinápticos distintos a cada execução?
4. Promova algum tipo de alteração nas especificações da ELM e/ou de seu treinamento de modo a produzir resultados superiores àqueles conquistados ao se seguir o roteiro desta questão. Descreva adequadamente as alterações realizadas.

Especificações para Q3, Q4 e Q5: Trabalhe no ambiente Anaconda (https://www.anaconda.com/products/individual) ou JupyterLab (https://jupyter.readthedocs.io/en/latest/content-quickstart.html) ou Google Colab (https://colab.research.google.com/notebooks/intro.ipynb). Cada atividade deve compor um notebook.

Questão 3)

Tomando os mesmos problemas de classificação de dados das bases MNIST e CIFAR10, use o framework Keras, tendo o TensorFlow como *backend* e realize o treinamento de uma rede neural MLP. Busque inspiração em resultados já publicados na literatura e/ou adote o procedimento de tentativa-e-erro para definir, da melhor forma que você puder, o número de camadas intermediárias, o número de neurônios por camada, o algoritmo de ajuste de pesos, a taxa de *dropout* (onde for pertinente) e o número de épocas de treinamento. Procure trabalhar com a média de várias execuções (junto a cada configuração candidata) para se chegar a um índice de desempenho mais estável. Um código que pode servir de ponto de partida é fornecido a seguir, considerando 1 camada intermediária, 512 neurônios nesta camada intermediária, ADAM, ocorrência de dropout com taxa de 50%, 5 épocas de treinamento e função perda sendo uma forma de entropia cruzada. A sua proposta deve ser capaz de superar o desempenho desta sugestão abaixo e você deve descrever de forma objetiva o caminho trilhado até a sua configuração final de código para a rede neural MLP, assim como uma comparação de desempenho com a sugestão abaixo (adaptada para a base MNIST).

```
import tensorflow as tf
import os
mnist = tf.keras.datasets.mnist

(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(512, activation=tf.nn.relu),
  tf.keras.layers.Dropout(0.5),
  tf.keras.layers.Dense(10, activation=tf.nn.softmax)
])
model.compile(optimizer='adam',
          loss='sparse_categorical_crossentropy',
          metrics=['accuracy'])
model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
model_json = model.to_json()
json_file = open("model_MLP.json", "w")
json_file.write(model_json)
json_file.close()
model.save_weights("model_MLP.h5")
print("Model saved to disk")
os.getcwd()
```

Questão 4)
Tomando os mesmos problemas de classificação de dados das bases MNIST e CIFAR10 e novamente usando o framework Keras, tendo o TensorFlow como *backend*, realize o treinamento de uma rede neural com camadas convolucionais, usando maxpooling e dropout. Mais uma vez, é apresentada a seguir uma sugestão de código e de configuração de hiperparâmetros que pode ser tomada como ponto de partida. A sua proposta deve superar, em termos de desempenho médio, essa sugestão fornecida abaixo, voltada para a base MNIST. Compare os resultados (em termos de taxa de acerto na classificação) com aqueles obtidos pelos três tipos de máquinas de aprendizado adotadas nas atividades anteriores (classificador linear, ELM e MLP), consolidando os resultados numa única tabela, para cada base de dados. Descreva de forma objetiva o caminho trilhado até a sua configuração final de código.

```
import tensorflow as tf
import os
mnist = tf.keras.datasets.mnist

(x_train, y_train),(x_test, y_test) = mnist.load_data()

# reshape to be [samples][width][height][pixels]
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 = tf.keras.models.Sequential()
model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3),
                        activation='relu',
                        input_shape=(28, 28, 1)))
model.add(tf.keras.layers.Conv2D(64, (3, 3), activation='relu'))
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
model.add(tf.keras.layers.Dropout(0.25))
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(128, activation='relu'))
model.add(tf.keras.layers.Dropout(0.5))
model.add(tf.keras.layers.Dense(10, activation='softmax'))

model.compile(optimizer='adam',
          loss='sparse_categorical_crossentropy',
          metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)

model_json = model.to_json()
json_file = open("model_CNN.json", "w")
json_file.write(model_json)
json_file.close()
model.save_weights("model_CNN.h5")
print("Model saved to disk")
os.getcwd()
```

Questão 5)

Tomando a base de dados MNIST, treinar um *autoencoder* e constatar o sucesso da tarefa de codificação e de decodificação (mesmo que ainda com algumas imperfeições), indicando que duas variáveis latentes podem ser suficientes para codificar a essência de toda a variedade exibida pelas versões manuscritas de 10 dígitos. Por fim, explorar o *manifold* gerado no gargalo do *autoencoder*. Para tanto, como um primeiro passo, execute o notebook fornecido pelo professor (**Q5.ipynb**) e procure compreender o que está sendo feito em cada trecho de código. Atividades práticas:

1. Com base no desempenho obtido com a proposta fornecida, procure melhorar a distribuição das classes ao longo do *manifold*, atuando na arquitetura do *autoencoder* e nos parâmetros de treinamento. Forneça o novo notebook gerado, com a denominação **([seu_RA]Q5_1.ipynb**).
2. Apresente um notebook que aborde algum outro problema a ser resolvido com *autoencoder* (pode ser a versão variacional), executando e comentando as principais etapas envolvidas na solução. Devem ser apresentados resultados gráficos que comprovem o bom desempenho. Forneça o novo notebook gerado, com a denominação **([seu_RA]Q5_2.ipynb**).

Observações gerais:

- Os entregáveis devem ser enviados para o e-mail [vonzuben@dca.fee.unicamp.br] com Assunto [IA353 - EF1] até 23h59 do prazo final. Aguarde um retorno por e-mail com a confirmação de recebimento.