

## EA072 – Inteligência Artificial em Aplicações Industriais (2s2020)

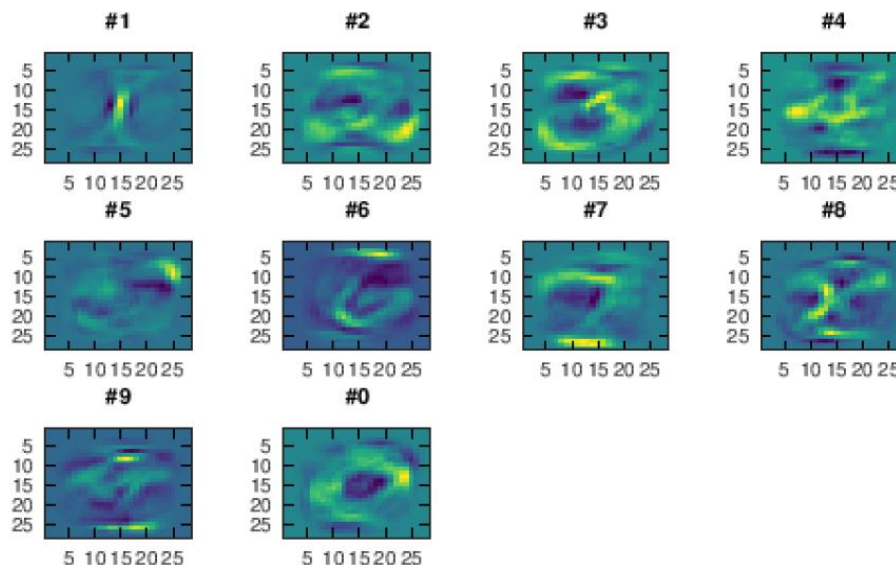
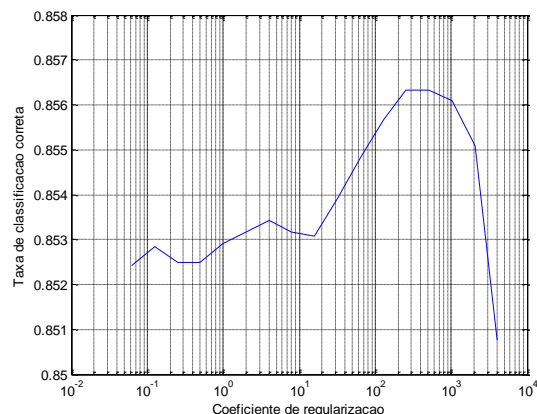
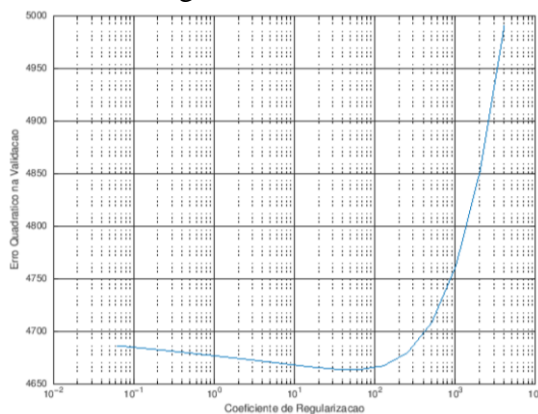
### Exercícios Computacionais 1 – ECp1

#### Atividade Individual – Peso 5 – Data de Entrega: 04/11/2020

#### Questão 1)

Objetivo: Síntese de modelos lineares para classificação de padrões.

O que deve ser entregue por e-mail: Um arquivo 'w[RA\_do(a)\_aluno(a)].txt' com a matriz  $W$ , conforme especificado no enunciado da questão, e um arquivo [Q1\_RA\_do(a)\_aluno(a)].pdf com os valores dos 4 coeficientes de regularização encontrados na busca (antes e após o refinamento, para o erro quadrático médio e para o erro de classificação). Apresentar esses resultados numa tabela. Incluir também no relatório dois gráficos semilog do desempenho dos classificadores junto aos dados de validação para os 11 valores de coeficiente de regularização sugeridos (um para o erro quadrático médio e outro para o erro de classificação), dois gráficos semilog com uma busca refinada no entorno do mínimo dos gráficos anteriores, e um gráfico de calor para os parâmetros de cada um dos 10 classificadores lineares. No arquivo TXT e no gráfico de calor, opte pelo melhor classificador produzido pelo critério de desempenho associado ao erro de classificação junto aos dados de validação. Apresente a matriz de confusão ([https://en.wikipedia.org/wiki/Confusion\\_matrix](https://en.wikipedia.org/wiki/Confusion_matrix)) e também exemplos de dígitos classificados equivocadamente, ao menos de três classes diferentes. Os gráficos devem ter aproximadamente os formatos e padrões de cores ilustrados a seguir.



Caso de estudo: 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 Octave / Matlab, o professor está fornecendo arquivos \*.mat para os dados de treinamento e teste, contendo 2 matrizes cada. 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  $X_t$  e  $S_t$ . O professor também disponibiliza um programa \*.m para visualização da imagem dos dígitos.

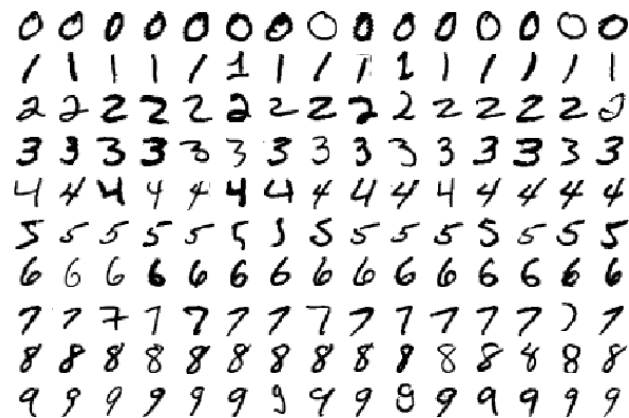


Figura 1 – Exemplos de imagens do conjunto de dados MNIST. Nos dados, a classe '0' é a última, e não a primeira, como na figura acima.

Obtenha um modelo de classificação linear, 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 como primeira coluna da matriz  $X$ :**

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

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) para poder implementar esta busca pelo melhor coeficiente de regularização, considerando dois critérios de desempenho para o classificador: erro quadrático médio e taxa de erro de classificação. O coeficiente de regularização deve ser buscado iniciando por uma busca no conjunto de valores candidatos:

$$\{2^{-10}, 2^{-8}, \dots, 2^0, 2^{+2}, \dots, 2^{+10}\}$$

Uma vez encontrado um valor adequado para o coeficiente de regularização, usar todas as 60.000 amostras de treinamento para sintetizar o classificador linear. Use sempre o conjunto de dados de treinamento e não o conjunto de dados de teste.

Sugestão: 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. É apresentada a seguir parte do código de um programa em Octave / Matlab que resolve parte das demandas desta questão.

```
load data;
load test;
[N,m] = size(X);
X = [ones(N,1) X];
[Nt,mt] = size(Xt);
Xt = [ones(Nt,1) Xt];
new_order = randperm(N);
X = X(new_order,:);
S = S(new_order,:);
part = 0.8;
Xtr = X(1:(part*N),:);
Xv = X((part*N+1):N,:);
Str = S(1:(part*N),:);
Sv = S((part*N+1):N,:);
v_c = [];
v_tx_acerto = [];
tx_acerto_max = 0;
for p=-10:2:10,
    c = 2^p;
    w = (Xtr'*Xtr+c*eye(m+1))\Xtr'*Str;
    S_est = Xv*w;
    acerto = 0;
    for i=1:(0.2*N),
        [vmax,indmax] = max(S_est(i,:));
        if Sv(i,indmax) > 0.5,
            acerto = acerto+1;
        end
    end
    tx_acerto = acerto/(0.2*N);
    v_c = [v_c;c];
    v_tx_acerto = [v_tx_acerto;tx_acerto];
    if tx_acerto >= tx_acerto_max,
        tx_acerto_max = tx_acerto;
        c_min = c;
        w_min = w;
    end
end
```

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.

A seguir, serão apresentados conceitos fundamentais que devem ser utilizados para resolver esta questão.

## 1 Regressão de quadrados mínimos

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

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

- 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  $\mathbf{w}$ , constata-se que a regressão de quadrados mínimos requer a solução de um sistema linear sobredeterminado, na forma:

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

- 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  $X \in \mathfrak{R}^{N \times (n+1)}$ ,  $\mathbf{w} \in \mathfrak{R}^{(n+1) \times 1}$ ,  $\mathbf{y} \in \mathfrak{R}^{N \times 1}$  e  $N \geq (n+1)$ , sendo  $X$  uma matriz de posto completo, tem garantia de solução exata apenas quando  $N = (n+1)$ . Na situação em que  $N > (n+1)$ , 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  $\mathbf{w}$  tal que  $X\mathbf{w} = \mathbf{y}$ , mas não impede que se busque encontrar  $\mathbf{w}$  que resolva o seguinte problema de programação quadrática:

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

- A função-objetivo fica:

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

- 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  $\mathbf{w}$ , produzindo o

problema regularizado de regressão de quadrados mínimos, também denominado de *ridge regression*, na forma:

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

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

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

- 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:

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

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

## Questão 2)

Objetivo: Síntese de modelos não-lineares para classificação de padrões, mas lineares nos parâmetros ajustáveis. Para tanto, repetir o fluxograma da Q1, mas agora para uma máquina de aprendizado extremo (ELM) com 500 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.

O que deve ser entregue: 1 arquivo [Q2\_RA\_do(a)\_aluno(a)].pdf com os valores dos 2 coeficientes de regularização encontrados na busca (um para o erro quadrático médio e outro para o erro de classificação) e 2 gráficos semilog do desempenho dos classificadores junto aos dados de validação para os 11 valores de coeficiente de regularização sugeridos (um para o erro quadrático médio e outro para o erro de classificação), seguidos pelos 2 gráficos da fase de refinamento. Se o intervalo sugerido não for adequado, propor um intervalo de busca mais adequado. Análises a serem feitas: (2.1) 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. (2.2) Seguindo a sugestão de empregar 500 unidades na camada intermediária da rede neural, apresente argumentos para sustentar o ganho de desempenho verificado e uma execução em menor tempo computacional, quando comparado com o classificador linear da Q1. (2.3) Compare os coeficientes de regularização obtidos nessas duas primeiras atividades (classificador linear da Q1 e ELM da Q2), os apresentando numa única tabela para fácil visualização, e procure justificar a diferença. (2.4) 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? (2.5) 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>) e elabore o seu relatório como um Jupyter notebook. Todas as atividades deverão, assim, compor um único relatório no Jupyter notebook, ou um por questão. É possível também trabalhar no ambiente Google Colab (<https://colab.research.google.com/notebooks/intro.ipynb>).

### Questão 3)

Tomando o mesmo problema de classificação de dados da base MNIST, 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.

```
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 o mesmo problema de classificação de dados da base MNIST 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. 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. 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:

- Para as Questões 1 e 2, devem ser entregues um arquivo TXT e dois arquivos PDF, contendo as respostas para Q1 e Q2.
- Para as Questões 3, 4 e 5, deve ser entregue um único notebook (arquivo com extensão ipynb) ou então 3 notebooks, um para cada questão.
- Em todas as atividades, procure fixar a semente do gerador pseudo-aleatório, de modo a sempre produzir o mesmo resultado.
- 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.