In [1]:

```
# Não exibir warnings
import os
import sys
sys.stderr = open(os.devnull, "w")  # silence stderr
sys.stderr = sys.__stderr__  # unsilence stderr
```

In [2]:

```
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, confusion_matrix
import tensorflow as tf
import numpy as np
import pandas as pd
import seaborn as sns

from PIL import Image, ImageFile
ImageFile.LOAD_TRUNCATED_IMAGES = True
```

In [3]:

```
#Função de geração da matriz de confusão
def print_confusion_matrix(confusion_matrix, class_names, figsize = (10,7), fontsize=11
):
    df_cm = pd.DataFrame(
        confusion_matrix, index=class_names, columns=class_names,
    )
    fig = plt.figure(figsize=figsize)
    try:
        heatmap = sns.heatmap(df_cm, cmap="YlGnBu", annot=True, fmt="d")
    except ValueError:
        raise ValueError("Confusion matrix values must be integers.")
    heatmap.yaxis.set_ticklabels(heatmap.yaxis.get_ticklabels(), rotation=0, ha='right'
, fontsize=fontsize)
    heatmap.xaxis.set_ticklabels(heatmap.xaxis.get_ticklabels(), rotation=30, ha='righ
t', fontsize=fontsize)
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    #return fig
```

In [4]:

```
batch = 32
#num_train = 5600
#num_validation = 2400
```

In [5]:

```
# Part 1 - Configuring the CNN

# Initialising the CNN
classifier = Sequential()

# Step 1 - Convolution
classifier.add(Conv2D(64, (3, 3), input_shape = (128, 128, 3), activation = 'relu'))

# Step 2 - Pooling
classifier.add(MaxPooling2D(pool_size = (2, 2)))

# Adding a second convolutional layer
classifier.add(Conv2D(64, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))

# Adding a second convolutional layer
classifier.add(Conv2D(128, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))

# Adding a second convolutional layer
classifier.add(Conv2D(128, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))

classifier.add(Conv2D(256, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))

# Step 3 - Flattening
classifier.add(Flatten())

# Step 4 - Full connection
classifier.add(Dense(units = 256, activation = 'relu'))
classifier.add(Dense(units = 2, activation = 'sigmoid')) #mudar unidades para numero de
classes

# Compiling the CNN
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['a
ccuracy'])#adam
```

In [6]:

```
# Part 2 - Fitting the CNN to the images

from keras.preprocessing.image import ImageDataGenerator

#Conjunto de treinamento
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

training_set = train_datagen.flow_from_directory('classificador_A/train',
                                                 target_size = (128, 128),
                                                 color_mode="rgb",
                                                 batch_size = batch,
                                                 class_mode = 'categorical',
                                                 shuffle = True)

print (training_set.class_indices)
#Conjunto de validação
validation_datagen = ImageDataGenerator(rescale = 1./255)

validation_set = validation_datagen.flow_from_directory('classificador_A/validation/',
                                                target_size = (128, 128),
                                                color_mode="rgb",
                                                batch_size = batch, #alterado para 1
                                                class_mode = 'categorical',
                                                shuffle=True)
num_train = training_set.samples
num_validation = validation_set.samples
```

```
Found 4991 images belonging to 2 classes.
{'documentos': 0, 'nao_documentos': 1}
Found 2400 images belonging to 2 classes.
```

In [7]:

```
with tf.device('/gpu:0'): #rodar na GPU
    history = classifier.fit_generator(training_set,
                          steps_per_epoch = (num_train//batch),
                          epochs = 50,
                          validation_data = validation_set,
                          validation_steps = (num_validation//batch), verbose=1)


classifier.save('modelo_classificador_A_dist_03.h5')
```

```
Epoch 1/50
155/155 [==============================] - 581s 4s/step - loss: 0.5410 - a
cc: 0.6899 - val_loss: 0.3828 - val_acc: 0.8500
Epoch 2/50
155/155 [==============================] - 539s 3s/step - loss: 0.2856 - a
cc: 0.8903 - val_loss: 0.2511 - val_acc: 0.9125
Epoch 3/50
155/155 [==============================] - 538s 3s/step - loss: 0.2485 - a
cc: 0.9081 - val_loss: 0.2338 - val_acc: 0.9242
Epoch 4/50
155/155 [==============================] - 539s 3s/step - loss: 0.2073 - a
cc: 0.9270 - val_loss: 0.1999 - val_acc: 0.9446
Epoch 5/50
155/155 [==============================] - 563s 4s/step - loss: 0.1946 - a
cc: 0.9308 - val_loss: 0.1658 - val_acc: 0.9442
Epoch 6/50
155/155 [==============================] - 613s 4s/step - loss: 0.1768 - a
cc: 0.9347 - val_loss: 0.1865 - val_acc: 0.9371
Epoch 7/50
155/155 [==============================] - 555s 4s/step - loss: 0.1663 - a
cc: 0.9411 - val_loss: 0.1419 - val_acc: 0.9546
Epoch 8/50
155/155 [==============================] - 530s 3s/step - loss: 0.1351 - a
cc: 0.9514 - val_loss: 0.1462 - val_acc: 0.9550
Epoch 9/50
155/155 [==============================] - 530s 3s/step - loss: 0.1281 - a
cc: 0.9536 - val_loss: 0.1619 - val_acc: 0.9521
Epoch 10/50
155/155 [==============================] - 527s 3s/step - loss: 0.1283 - a
cc: 0.9554 - val_loss: 0.1746 - val_acc: 0.9537
Epoch 11/50
155/155 [==============================] - 531s 3s/step - loss: 0.1156 - a
cc: 0.9599 - val_loss: 0.1386 - val_acc: 0.9587
Epoch 12/50
155/155 [==============================] - 529s 3s/step - loss: 0.1176 - a
cc: 0.9617 - val_loss: 0.1832 - val_acc: 0.9458
Epoch 13/50
155/155 [==============================] - 534s 3s/step - loss: 0.0939 - a
cc: 0.9675 - val_loss: 0.1214 - val_acc: 0.9633
Epoch 14/50
155/155 [==============================] - 534s 3s/step - loss: 0.1217 - a
cc: 0.9611 - val_loss: 0.1564 - val_acc: 0.9471
Epoch 15/50
155/155 [==============================] - 598s 4s/step - loss: 0.0850 - a
cc: 0.9728 - val_loss: 0.0977 - val_acc: 0.9717
Epoch 16/50
155/155 [==============================] - 716s 5s/step - loss: 0.1031 - a
cc: 0.9679 - val_loss: 0.1007 - val_acc: 0.9708
Epoch 17/50
155/155 [==============================] - 580s 4s/step - loss: 0.0866 - a
cc: 0.9714 - val_loss: 0.0998 - val_acc: 0.9696
Epoch 18/50
155/155 [==============================] - 466s 3s/step - loss: 0.0760 - a
cc: 0.9732 - val_loss: 0.1350 - val_acc: 0.9637
Epoch 19/50
155/155 [==============================] - 464s 3s/step - loss: 0.0702 - a
cc: 0.9754 - val_loss: 0.1108 - val_acc: 0.9704
Epoch 20/50
155/155 [==============================] - 515s 3s/step - loss: 0.0760 - a
cc: 0.9768 - val_loss: 0.0978 - val_acc: 0.9708
Epoch 21/50
```

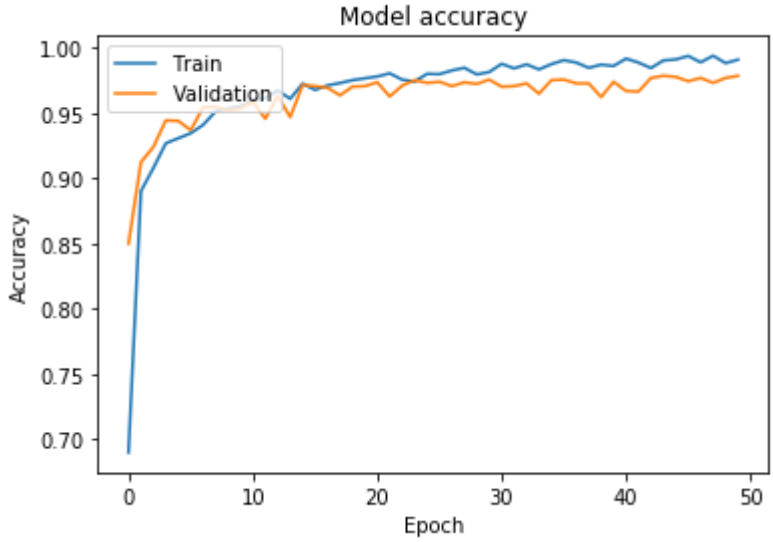
```
155/155 [==============================] - 493s 3s/step - loss: 0.0670 - a
cc: 0.9782 - val_loss: 0.0949 - val_acc: 0.9738
Epoch 22/50
155/155 [==============================] - 482s 3s/step - loss: 0.0609 - a
cc: 0.9806 - val_loss: 0.1654 - val_acc: 0.9629
Epoch 23/50
155/155 [==============================] - 483s 3s/step - loss: 0.0701 - a
cc: 0.9758 - val_loss: 0.1321 - val_acc: 0.9712
Epoch 24/50
155/155 [==============================] - 483s 3s/step - loss: 0.0763 - a
cc: 0.9744 - val_loss: 0.0894 - val_acc: 0.9754
Epoch 25/50
155/155 [==============================] - 483s 3s/step - loss: 0.0578 - a
cc: 0.9802 - val_loss: 0.1062 - val_acc: 0.9733
Epoch 26/50
155/155 [==============================] - 518s 3s/step - loss: 0.0576 - a
cc: 0.9800 - val_loss: 0.0862 - val_acc: 0.9742
Epoch 27/50
155/155 [==============================] - 502s 3s/step - loss: 0.0513 - a
cc: 0.9829 - val_loss: 0.1060 - val_acc: 0.9708
Epoch 28/50
155/155 [==============================] - 563s 4s/step - loss: 0.0454 - a
cc: 0.9849 - val_loss: 0.1079 - val_acc: 0.9738
Epoch 29/50
155/155 [==============================] - 480s 3s/step - loss: 0.0575 - a
cc: 0.9798 - val_loss: 0.1014 - val_acc: 0.9725
Epoch 30/50
155/155 [==============================] - 503s 3s/step - loss: 0.0566 - a
cc: 0.9817 - val_loss: 0.1032 - val_acc: 0.9758
Epoch 31/50
155/155 [==============================] - 539s 3s/step - loss: 0.0403 - a
cc: 0.9879 - val_loss: 0.1207 - val_acc: 0.9704
Epoch 32/50
155/155 [==============================] - 476s 3s/step - loss: 0.0426 - a
cc: 0.9845 - val_loss: 0.1127 - val_acc: 0.9708
Epoch 33/50
155/155 [==============================] - 482s 3s/step - loss: 0.0382 - a
cc: 0.9875 - val_loss: 0.1138 - val_acc: 0.9729
Epoch 34/50
155/155 [==============================] - 471s 3s/step - loss: 0.0488 - a
cc: 0.9837 - val_loss: 0.1143 - val_acc: 0.9650
Epoch 35/50
155/155 [==============================] - 522s 3s/step - loss: 0.0366 - a
cc: 0.9877 - val_loss: 0.1086 - val_acc: 0.9754
Epoch 36/50
155/155 [==============================] - 484s 3s/step - loss: 0.0297 - a
cc: 0.9907 - val_loss: 0.1081 - val_acc: 0.9758
Epoch 37/50
155/155 [==============================] - 556s 4s/step - loss: 0.0295 - a
cc: 0.9889 - val_loss: 0.1390 - val_acc: 0.9729
Epoch 38/50
155/155 [==============================] - 466s 3s/step - loss: 0.0419 - a
cc: 0.9849 - val_loss: 0.1209 - val_acc: 0.9729
Epoch 39/50
155/155 [==============================] - 531s 3s/step - loss: 0.0369 - a
cc: 0.9873 - val_loss: 0.1283 - val_acc: 0.9625
Epoch 40/50
155/155 [==============================] - 530s 3s/step - loss: 0.0362 - a
cc: 0.9863 - val_loss: 0.1229 - val_acc: 0.9742
Epoch 41/50
155/155 [==============================] - 500s 3s/step - loss: 0.0231 - a
```

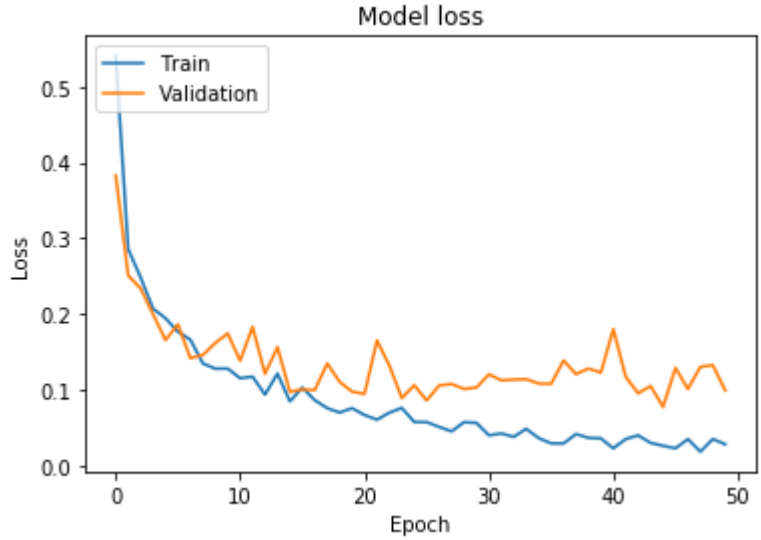
```
cc: 0.9919 - val_loss: 0.1803 - val_acc: 0.9671
Epoch 42/50
155/155 [==============================] - 598s 4s/step - loss: 0.0353 - a
cc: 0.9889 - val_loss: 0.1173 - val_acc: 0.9667
Epoch 43/50
155/155 [==============================] - 581s 4s/step - loss: 0.0402 - a
cc: 0.9847 - val_loss: 0.0957 - val_acc: 0.9771
Epoch 44/50
155/155 [==============================] - 551s 4s/step - loss: 0.0302 - a
cc: 0.9905 - val_loss: 0.1053 - val_acc: 0.9788
Epoch 45/50
155/155 [==============================] - 589s 4s/step - loss: 0.0264 - a
cc: 0.9913 - val_loss: 0.0777 - val_acc: 0.9779
Epoch 46/50
155/155 [==============================] - 574s 4s/step - loss: 0.0231 - a
cc: 0.9939 - val_loss: 0.1291 - val_acc: 0.9746
Epoch 47/50
155/155 [==============================] - 547s 4s/step - loss: 0.0351 - a
cc: 0.9891 - val_loss: 0.1012 - val_acc: 0.9771
Epoch 48/50
155/155 [==============================] - 559s 4s/step - loss: 0.0187 - a
cc: 0.9942 - val_loss: 0.1305 - val_acc: 0.9733
Epoch 49/50
155/155 [==============================] - 608s 4s/step - loss: 0.0353 - a
cc: 0.9885 - val_loss: 0.1328 - val_acc: 0.9771
Epoch 50/50
155/155 [==============================] - 612s 4s/step - loss: 0.0284 - a
cc: 0.9911 - val_loss: 0.0995 - val_acc: 0.9788
```

In [8]:

```
# Final accuracy and loss
print ("Train accuracy: %.3f" % (history.history['acc'][-1]))
print ("Train loss: %.3f" % (history.history['loss'][-1]),"\n")

print ("Validation accuracy: %.3f" % (history.history['val_acc'][-1]))
print ("Validation loss: %.3f" % (history.history['val_loss'][-1]))

# Plot training & validation accuracy values
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

# Plot training & validation loss values
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```

```
Train accuracy: 0.991
Train loss: 0.028

Validation accuracy: 0.979
Validation loss: 0.099
```

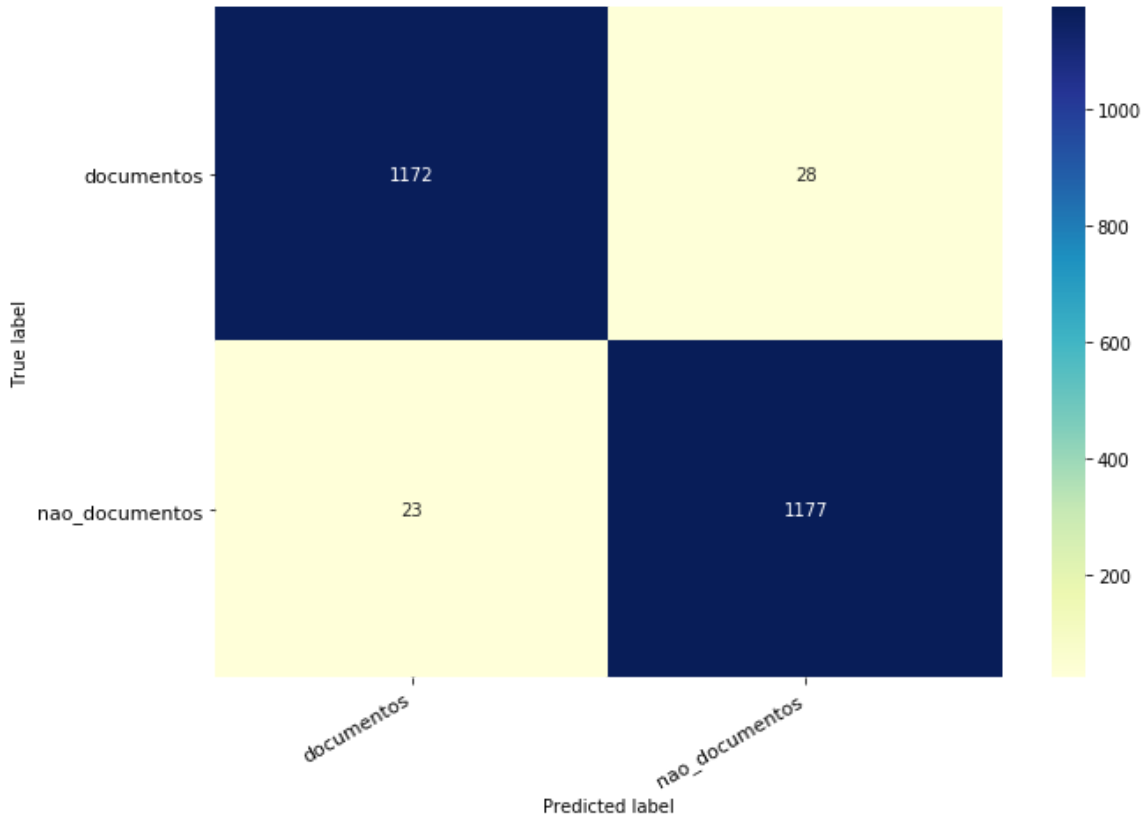




In [12]:

```
### Conjunto de Validação ###

print ("### Matriz de confusão para o conjunto de validação ###")

#Conjunto de validação
validation_datagen = ImageDataGenerator(rescale = 1./255)

validation_set = validation_datagen.flow_from_directory('classificador_A/validation/',
                                                        target_size = (128, 128),
                                                        color_mode="rgb",
                                                        batch_size = batch, #alterado para 1
                                                        class_mode = 'categorical',
                                                        shuffle= False)


#Confution Matrix
Y_pred = classifier.predict_generator(validation_set, num_validation//batch, verbose=1)

test_preds = np.argmax(Y_pred, axis=-1)
l=test_preds.shape[0]
test_trues = validation_set.classes
cm =confusion_matrix(test_trues[:l], test_preds)

print_confusion_matrix(cm, ["documentos", "nao_documentos"], figsize = (10,7), fontsize
=11)
```

```
### Matriz de confusão para o conjunto de validação ###
Found 2400 images belonging to 2 classes.
75/75 [==============================] - 112s 1s/step
```



In [13]:

```
### Conjunto de Teste ###

print ("### Matriz de confusão para o conjunto de teste ###")

test_datagen = ImageDataGenerator(rescale = 1./255)

test_set = test_datagen.flow_from_directory('classificador_A/test/',
                                            target_size = (128, 128),
                                            color_mode="rgb",
                                            batch_size = 1,
                                            class_mode = 'categorical',
                                            shuffle=False)
num_test = test_set.samples


#Confution Matrix
Y_pred = classifier.predict_generator(test_set, num_test, verbose=1)

test_preds = np.argmax(Y_pred, axis=-1)
l=test_preds.shape[0]
test_trues = test_set.classes
cm =confusion_matrix(test_trues[:l], test_preds)

print_confusion_matrix(cm, ["documentos", "nao_documentos"], figsize = (10,7), fontsize
=11)

# Accuracy and Loss for the Test set
loss, acc = classifier.evaluate_generator(test_set, num_test, verbose=1)

# Final accuracy and loss
print ("Test accuracy: %.3f" % acc)
print ("Test loss: %.3f" % loss)
```

```
### Matriz de confusão para o conjunto de teste ###
Found 1199 images belonging to 2 classes.
1199/1199 [==============================] - 74s 62ms/step
1199/1199 [==============================] - 96s 80ms/step
Test accuracy: 0.982
Test loss: 0.104
```