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 4993 images belonging to 2 classes.
{'documentos': 0, 'nao_documentos': 1}
Found 2399 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_02.h5')
```

```
Epoch 1/50
156/156 [==============================] - 553s 4s/step - loss: 0.3699 - a
cc: 0.8407 - val_loss: 0.3011 - val_acc: 0.8801
Epoch 2/50
156/156 [==============================] - 548s 4s/step - loss: 0.2742 - a
cc: 0.8932 - val_loss: 0.2493 - val_acc: 0.9092
Epoch 3/50
156/156 [==============================] - 536s 3s/step - loss: 0.2258 - a
cc: 0.9187 - val_loss: 0.2033 - val_acc: 0.9227
Epoch 4/50
156/156 [==============================] - 531s 3s/step - loss: 0.2147 - a
cc: 0.9157 - val_loss: 0.2508 - val_acc: 0.8995
Epoch 5/50
156/156 [==============================] - 534s 3s/step - loss: 0.1895 - a
cc: 0.9309 - val_loss: 0.1875 - val_acc: 0.9278
Epoch 6/50
156/156 [==============================] - 538s 3s/step - loss: 0.1846 - a
cc: 0.9369 - val_loss: 0.1830 - val_acc: 0.9400
Epoch 7/50
156/156 [==============================] - 557s 4s/step - loss: 0.1822 - a
cc: 0.9359 - val_loss: 0.1598 - val_acc: 0.9434
Epoch 8/50
156/156 [==============================] - 622s 4s/step - loss: 0.1350 - a
cc: 0.9527 - val_loss: 0.1138 - val_acc: 0.9599
Epoch 9/50
156/156 [==============================] - 539s 3s/step - loss: 0.1335 - a
cc: 0.9521 - val_loss: 0.1306 - val_acc: 0.9590
Epoch 10/50
156/156 [==============================] - 532s 3s/step - loss: 0.1166 - a
cc: 0.9621 - val_loss: 0.1274 - val_acc: 0.9523
Epoch 11/50
156/156 [==============================] - 531s 3s/step - loss: 0.1141 - a
cc: 0.9599 - val_loss: 0.1180 - val_acc: 0.9594
Epoch 12/50
156/156 [==============================] - 534s 3s/step - loss: 0.1019 - a
cc: 0.9639 - val_loss: 0.1074 - val_acc: 0.9632
Epoch 13/50
156/156 [==============================] - 774s 5s/step - loss: 0.0977 - a
cc: 0.9681 - val_loss: 0.1301 - val_acc: 0.9578
Epoch 14/50
156/156 [==============================] - 801s 5s/step - loss: 0.1089 - a
cc: 0.9609 - val_loss: 0.1220 - val_acc: 0.9603
Epoch 15/50
156/156 [==============================] - 647s 4s/step - loss: 0.0955 - a
cc: 0.9657 - val_loss: 0.1359 - val_acc: 0.9611
Epoch 16/50
156/156 [==============================] - 806s 5s/step - loss: 0.0882 - a
cc: 0.9671 - val_loss: 0.0920 - val_acc: 0.9717
Epoch 17/50
156/156 [==============================] - 590s 4s/step - loss: 0.0750 - a
cc: 0.9746 - val_loss: 0.0992 - val_acc: 0.9704
Epoch 18/50
156/156 [==============================] - 561s 4s/step - loss: 0.0688 - a
cc: 0.9776 - val_loss: 0.1120 - val_acc: 0.9620
Epoch 19/50
156/156 [==============================] - 566s 4s/step - loss: 0.0719 - a
cc: 0.9784 - val_loss: 0.1112 - val_acc: 0.9687
Epoch 20/50
156/156 [==============================] - 469s 3s/step - loss: 0.0767 - a
cc: 0.9754 - val_loss: 0.1793 - val_acc: 0.9472
Epoch 21/50
```

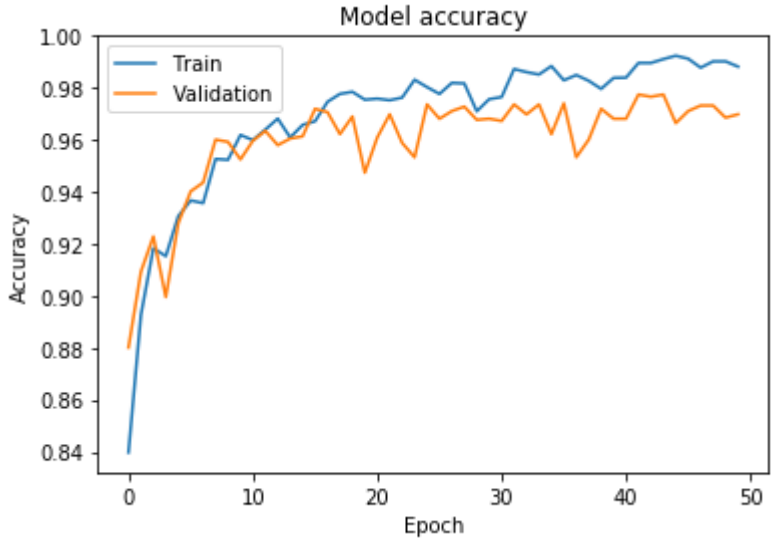
```
156/156 [==============================] - 518s 3s/step - loss: 0.0660 - a
cc: 0.9758 - val_loss: 0.1136 - val_acc: 0.9607
Epoch 22/50
156/156 [==============================] - 506s 3s/step - loss: 0.0769 - a
cc: 0.9752 - val_loss: 0.0953 - val_acc: 0.9696
Epoch 23/50
156/156 [==============================] - 525s 3s/step - loss: 0.0727 - a
cc: 0.9762 - val_loss: 0.1382 - val_acc: 0.9586
Epoch 24/50
156/156 [==============================] - 489s 3s/step - loss: 0.0494 - a
cc: 0.9830 - val_loss: 0.1460 - val_acc: 0.9531
Epoch 25/50
156/156 [==============================] - 482s 3s/step - loss: 0.0593 - a
cc: 0.9802 - val_loss: 0.1272 - val_acc: 0.9734
Epoch 26/50
156/156 [==============================] - 468s 3s/step - loss: 0.0644 - a
cc: 0.9776 - val_loss: 0.1858 - val_acc: 0.9679
Epoch 27/50
156/156 [==============================] - 464s 3s/step - loss: 0.0523 - a
cc: 0.9818 - val_loss: 0.1277 - val_acc: 0.9708
Epoch 28/50
156/156 [==============================] - 640s 4s/step - loss: 0.0500 - a
cc: 0.9816 - val_loss: 0.0992 - val_acc: 0.9725
Epoch 29/50
156/156 [==============================] - 585s 4s/step - loss: 0.0897 - a
cc: 0.9648 - val_loss: 0.1367 - val_acc: 0.9675
Epoch 30/50
156/156 [==============================] - 470s 3s/step - loss: 0.0700 - a
cc: 0.9756 - val_loss: 0.1090 - val_acc: 0.9679
Epoch 31/50
156/156 [==============================] - 469s 3s/step - loss: 0.0724 - a
cc: 0.9764 - val_loss: 0.1301 - val_acc: 0.9670
Epoch 32/50
156/156 [==============================] - 464s 3s/step - loss: 0.0387 - a
cc: 0.9870 - val_loss: 0.0986 - val_acc: 0.9734
Epoch 33/50
156/156 [==============================] - 449s 3s/step - loss: 0.0430 - a
cc: 0.9860 - val_loss: 0.1098 - val_acc: 0.9696
Epoch 34/50
156/156 [==============================] - 456s 3s/step - loss: 0.0417 - a
cc: 0.9850 - val_loss: 0.1156 - val_acc: 0.9734
Epoch 35/50
156/156 [==============================] - 459s 3s/step - loss: 0.0383 - a
cc: 0.9882 - val_loss: 0.1639 - val_acc: 0.9620
Epoch 36/50
156/156 [==============================] - 456s 3s/step - loss: 0.0530 - a
cc: 0.9828 - val_loss: 0.1335 - val_acc: 0.9738
Epoch 37/50
156/156 [==============================] - 455s 3s/step - loss: 0.0407 - a
cc: 0.9848 - val_loss: 0.1735 - val_acc: 0.9531
Epoch 38/50
156/156 [==============================] - 455s 3s/step - loss: 0.0593 - a
cc: 0.9826 - val_loss: 0.1284 - val_acc: 0.9599
Epoch 39/50
156/156 [==============================] - 454s 3s/step - loss: 0.0599 - a
cc: 0.9794 - val_loss: 0.0942 - val_acc: 0.9717
Epoch 40/50
156/156 [==============================] - 452s 3s/step - loss: 0.0510 - a
cc: 0.9838 - val_loss: 0.1314 - val_acc: 0.9679
Epoch 41/50
156/156 [==============================] - 456s 3s/step - loss: 0.0476 - a
```

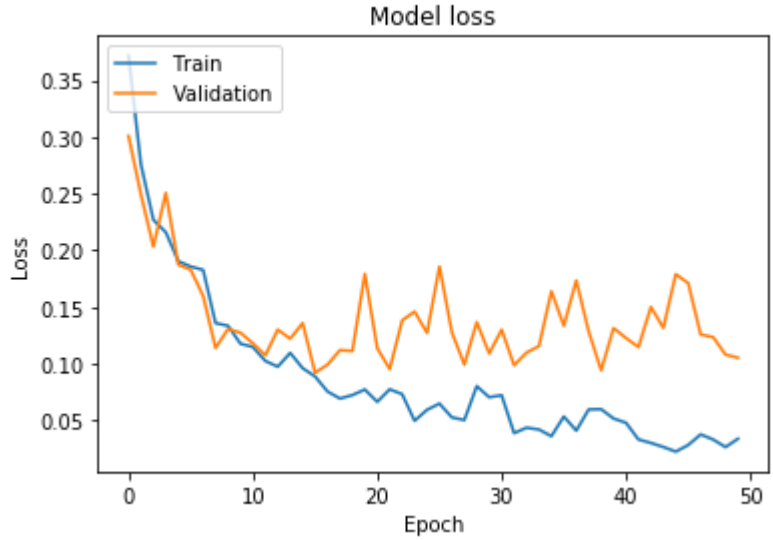
```
cc: 0.9838 - val_loss: 0.1226 - val_acc: 0.9679
Epoch 42/50
156/156 [==============================] - 458s 3s/step - loss: 0.0329 - a
cc: 0.9894 - val_loss: 0.1146 - val_acc: 0.9772
Epoch 43/50
156/156 [==============================] - 456s 3s/step - loss: 0.0299 - a
cc: 0.9894 - val_loss: 0.1501 - val_acc: 0.9763
Epoch 44/50
156/156 [==============================] - 462s 3s/step - loss: 0.0265 - a
cc: 0.9908 - val_loss: 0.1316 - val_acc: 0.9772
Epoch 45/50
156/156 [==============================] - 449s 3s/step - loss: 0.0221 - a
cc: 0.9922 - val_loss: 0.1790 - val_acc: 0.9662
Epoch 46/50
156/156 [==============================] - 457s 3s/step - loss: 0.0281 - a
cc: 0.9910 - val_loss: 0.1711 - val_acc: 0.9708
Epoch 47/50
156/156 [==============================] - 454s 3s/step - loss: 0.0374 - a
cc: 0.9876 - val_loss: 0.1257 - val_acc: 0.9730
Epoch 48/50
156/156 [==============================] - 455s 3s/step - loss: 0.0329 - a
cc: 0.9900 - val_loss: 0.1235 - val_acc: 0.9730
Epoch 49/50
156/156 [==============================] - 454s 3s/step - loss: 0.0262 - a
cc: 0.9900 - val_loss: 0.1080 - val_acc: 0.9683
Epoch 50/50
156/156 [==============================] - 455s 3s/step - loss: 0.0339 - a
cc: 0.9880 - val_loss: 0.1052 - val_acc: 0.9696
```

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: 0.988
Train loss: 0.034

Validation accuracy: 0.970
Validation loss: 0.105
```

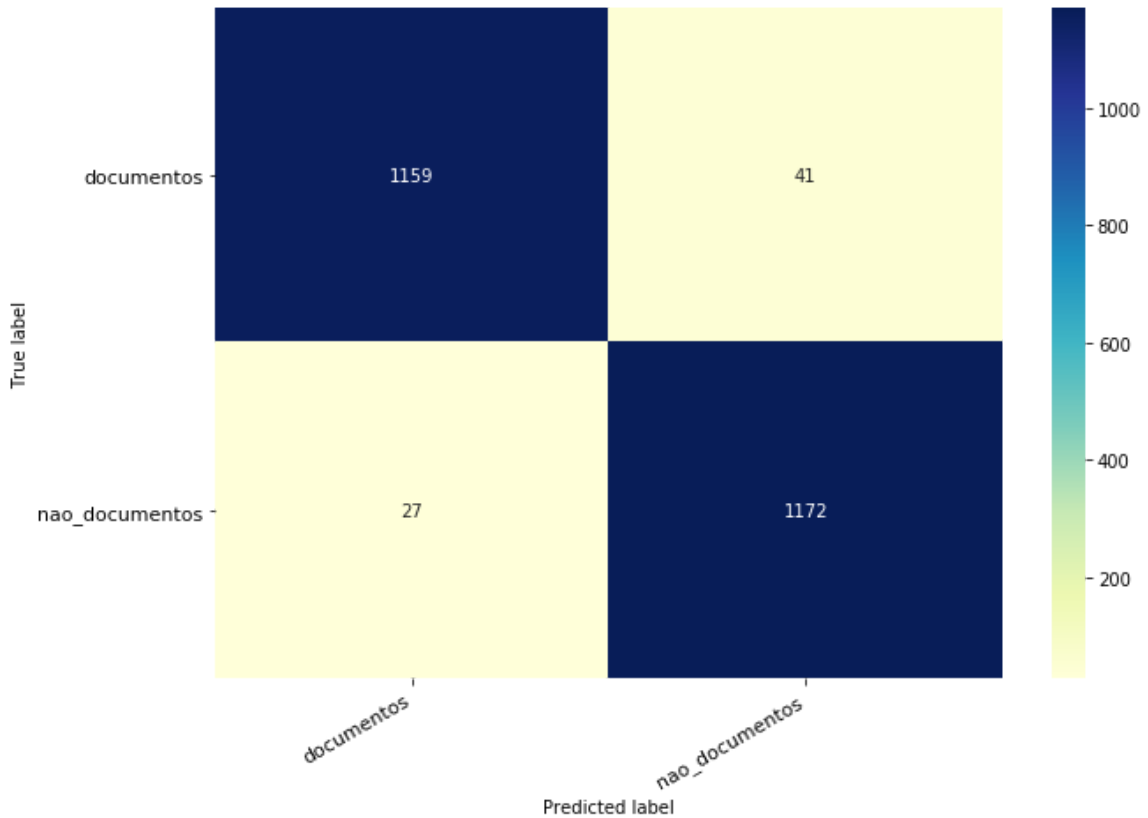




In [9]:

```
### 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+1, verbos
e=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 2399 images belonging to 2 classes.
75/75 [==============================] - 89s 1s/step
```



In [10]:

```
### 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 [==============================] - 67s 56ms/step
1199/1199 [==============================] - 80s 67ms/step
Test accuracy: 0.972
Test loss: 0.101
```