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 4996 images belonging to 2 classes.
{'documentos': 0, 'nao_documentos': 1}
Found 2396 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_05.h5')
```

```
Epoch 1/50
156/156 [==============================] - 690s 4s/step - loss: 0.3835 - a
cc: 0.8399 - val_loss: 0.4611 - val_acc: 0.7724
Epoch 2/50
156/156 [==============================] - 575s 4s/step - loss: 0.2735 - a
cc: 0.8936 - val_loss: 0.2469 - val_acc: 0.9141
Epoch 3/50
156/156 [==============================] - 647s 4s/step - loss: 0.2303 - a
cc: 0.9105 - val_loss: 0.3047 - val_acc: 0.8676
Epoch 4/50
156/156 [==============================] - 658s 4s/step - loss: 0.2040 - a
cc: 0.9247 - val_loss: 0.1577 - val_acc: 0.9480
Epoch 5/50
156/156 [==============================] - 641s 4s/step - loss: 0.1847 - a
cc: 0.9375 - val_loss: 0.1549 - val_acc: 0.9450
Epoch 6/50
156/156 [==============================] - 541s 3s/step - loss: 0.1582 - a
cc: 0.9427 - val_loss: 0.1426 - val_acc: 0.9543
Epoch 7/50
156/156 [==============================] - 536s 3s/step - loss: 0.1438 - a
cc: 0.9489 - val_loss: 0.1235 - val_acc: 0.9607
Epoch 8/50
156/156 [==============================] - 562s 4s/step - loss: 0.1532 - a
cc: 0.9483 - val_loss: 0.1826 - val_acc: 0.9471
Epoch 9/50
156/156 [==============================] - 659s 4s/step - loss: 0.1261 - a
cc: 0.9571 - val_loss: 0.1185 - val_acc: 0.9640
Epoch 10/50
156/156 [==============================] - 584s 4s/step - loss: 0.1317 - a
cc: 0.9537 - val_loss: 0.1195 - val_acc: 0.9734
Epoch 11/50
156/156 [==============================] - 652s 4s/step - loss: 0.1189 - a
cc: 0.9595 - val_loss: 0.1098 - val_acc: 0.9662
Epoch 12/50
156/156 [==============================] - 685s 4s/step - loss: 0.1004 - a
cc: 0.9687 - val_loss: 0.1454 - val_acc: 0.9577
Epoch 13/50
156/156 [==============================] - 548s 4s/step - loss: 0.1018 - a
cc: 0.9658 - val_loss: 0.1173 - val_acc: 0.9657
Epoch 14/50
156/156 [==============================] - 495s 3s/step - loss: 0.0867 - a
cc: 0.9720 - val_loss: 0.0967 - val_acc: 0.9729
Epoch 15/50
156/156 [==============================] - 488s 3s/step - loss: 0.0865 - a
cc: 0.9702 - val_loss: 0.0988 - val_acc: 0.9712
Epoch 16/50
156/156 [==============================] - 473s 3s/step - loss: 0.0968 - a
cc: 0.9690 - val_loss: 0.0871 - val_acc: 0.9772
Epoch 17/50
156/156 [==============================] - 476s 3s/step - loss: 0.0801 - a
cc: 0.9734 - val_loss: 0.0988 - val_acc: 0.9679
Epoch 18/50
156/156 [==============================] - 475s 3s/step - loss: 0.0853 - a
cc: 0.9712 - val_loss: 0.1108 - val_acc: 0.9695
Epoch 19/50
156/156 [==============================] - 497s 3s/step - loss: 0.0693 - a
cc: 0.9758 - val_loss: 0.1037 - val_acc: 0.9742
Epoch 20/50
156/156 [==============================] - 524s 3s/step - loss: 0.0702 - a
cc: 0.9780 - val_loss: 0.1110 - val_acc: 0.9729
Epoch 21/50
```

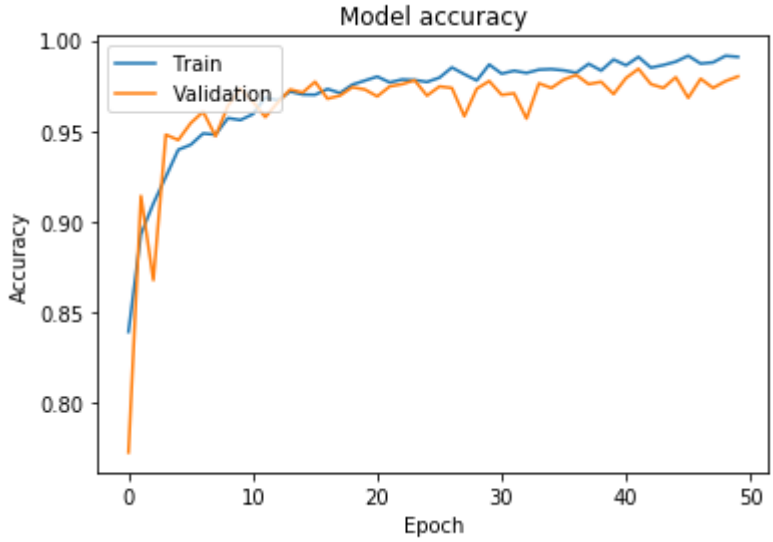
```
156/156 [==============================] - 479s 3s/step - loss: 0.0594 - a
cc: 0.9802 - val_loss: 0.1130 - val_acc: 0.9691
Epoch 22/50
156/156 [==============================] - 481s 3s/step - loss: 0.0722 - a
cc: 0.9770 - val_loss: 0.1015 - val_acc: 0.9746
Epoch 23/50
156/156 [==============================] - 472s 3s/step - loss: 0.0643 - a
cc: 0.9786 - val_loss: 0.0885 - val_acc: 0.9759
Epoch 24/50
156/156 [==============================] - 496s 3s/step - loss: 0.0556 - a
cc: 0.9782 - val_loss: 0.0940 - val_acc: 0.9780
Epoch 25/50
156/156 [==============================] - 483s 3s/step - loss: 0.0798 - a
cc: 0.9760 - val_loss: 0.1011 - val_acc: 0.9695
Epoch 26/50
156/156 [==============================] - 479s 3s/step - loss: 0.0541 - a
cc: 0.9796 - val_loss: 0.1134 - val_acc: 0.9746
Epoch 27/50
156/156 [==============================] - 477s 3s/step - loss: 0.0438 - a
cc: 0.9852 - val_loss: 0.1215 - val_acc: 0.9738
Epoch 28/50
156/156 [==============================] - 471s 3s/step - loss: 0.0540 - a
cc: 0.9814 - val_loss: 0.1752 - val_acc: 0.9581
Epoch 29/50
156/156 [==============================] - 471s 3s/step - loss: 0.0624 - a
cc: 0.9782 - val_loss: 0.0999 - val_acc: 0.9734
Epoch 30/50
156/156 [==============================] - 478s 3s/step - loss: 0.0437 - a
cc: 0.9868 - val_loss: 0.0804 - val_acc: 0.9776
Epoch 31/50
156/156 [==============================] - 475s 3s/step - loss: 0.0552 - a
cc: 0.9818 - val_loss: 0.1290 - val_acc: 0.9700
Epoch 32/50
156/156 [==============================] - 479s 3s/step - loss: 0.0509 - a
cc: 0.9834 - val_loss: 0.1328 - val_acc: 0.9708
Epoch 33/50
156/156 [==============================] - 543s 3s/step - loss: 0.0553 - a
cc: 0.9822 - val_loss: 0.1592 - val_acc: 0.9569
Epoch 34/50
156/156 [==============================] - 651s 4s/step - loss: 0.0503 - a
cc: 0.9840 - val_loss: 0.1013 - val_acc: 0.9763
Epoch 35/50
156/156 [==============================] - 543s 3s/step - loss: 0.0475 - a
cc: 0.9842 - val_loss: 0.0959 - val_acc: 0.9738
Epoch 36/50
156/156 [==============================] - 533s 3s/step - loss: 0.0506 - a
cc: 0.9836 - val_loss: 0.1176 - val_acc: 0.9784
Epoch 37/50
156/156 [==============================] - 519s 3s/step - loss: 0.0524 - a
cc: 0.9824 - val_loss: 0.0640 - val_acc: 0.9810
Epoch 38/50
156/156 [==============================] - 503s 3s/step - loss: 0.0343 - a
cc: 0.9872 - val_loss: 0.1421 - val_acc: 0.9759
Epoch 39/50
156/156 [==============================] - 463s 3s/step - loss: 0.0465 - a
cc: 0.9834 - val_loss: 0.0879 - val_acc: 0.9772
Epoch 40/50
156/156 [==============================] - 497s 3s/step - loss: 0.0306 - a
cc: 0.9896 - val_loss: 0.1506 - val_acc: 0.9704
Epoch 41/50
156/156 [==============================] - 559s 4s/step - loss: 0.0361 - a
```

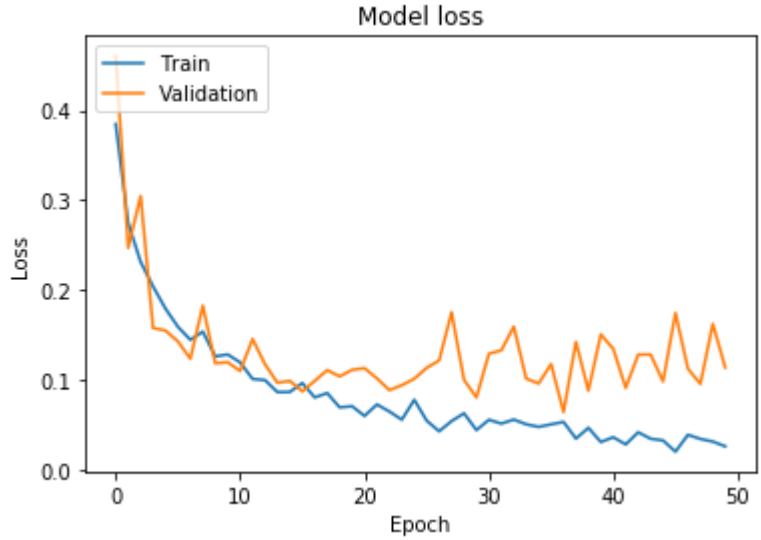
```
cc: 0.9864 - val_loss: 0.1345 - val_acc: 0.9793
Epoch 42/50
156/156 [==============================] - 632s 4s/step - loss: 0.0279 - a
cc: 0.9910 - val_loss: 0.0910 - val_acc: 0.9843
Epoch 43/50
156/156 [==============================] - 663s 4s/step - loss: 0.0416 - a
cc: 0.9850 - val_loss: 0.1280 - val_acc: 0.9759
Epoch 44/50
156/156 [==============================] - 564s 4s/step - loss: 0.0340 - a
cc: 0.9866 - val_loss: 0.1282 - val_acc: 0.9738
Epoch 45/50
156/156 [==============================] - 571s 4s/step - loss: 0.0323 - a
cc: 0.9884 - val_loss: 0.0982 - val_acc: 0.9797
Epoch 46/50
156/156 [==============================] - 578s 4s/step - loss: 0.0200 - a
cc: 0.9916 - val_loss: 0.1744 - val_acc: 0.9683
Epoch 47/50
156/156 [==============================] - 515s 3s/step - loss: 0.0383 - a
cc: 0.9874 - val_loss: 0.1124 - val_acc: 0.9788
Epoch 48/50
156/156 [==============================] - 568s 4s/step - loss: 0.0339 - a
cc: 0.9880 - val_loss: 0.0954 - val_acc: 0.9738
Epoch 49/50
156/156 [==============================] - 536s 3s/step - loss: 0.0313 - a
cc: 0.9916 - val_loss: 0.1623 - val_acc: 0.9776
Epoch 50/50
156/156 [==============================] - 613s 4s/step - loss: 0.0257 - a
cc: 0.9910 - val_loss: 0.1138 - val_acc: 0.9801
```

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: 0.026

Validation accuracy: 0.980
Validation loss: 0.114
```

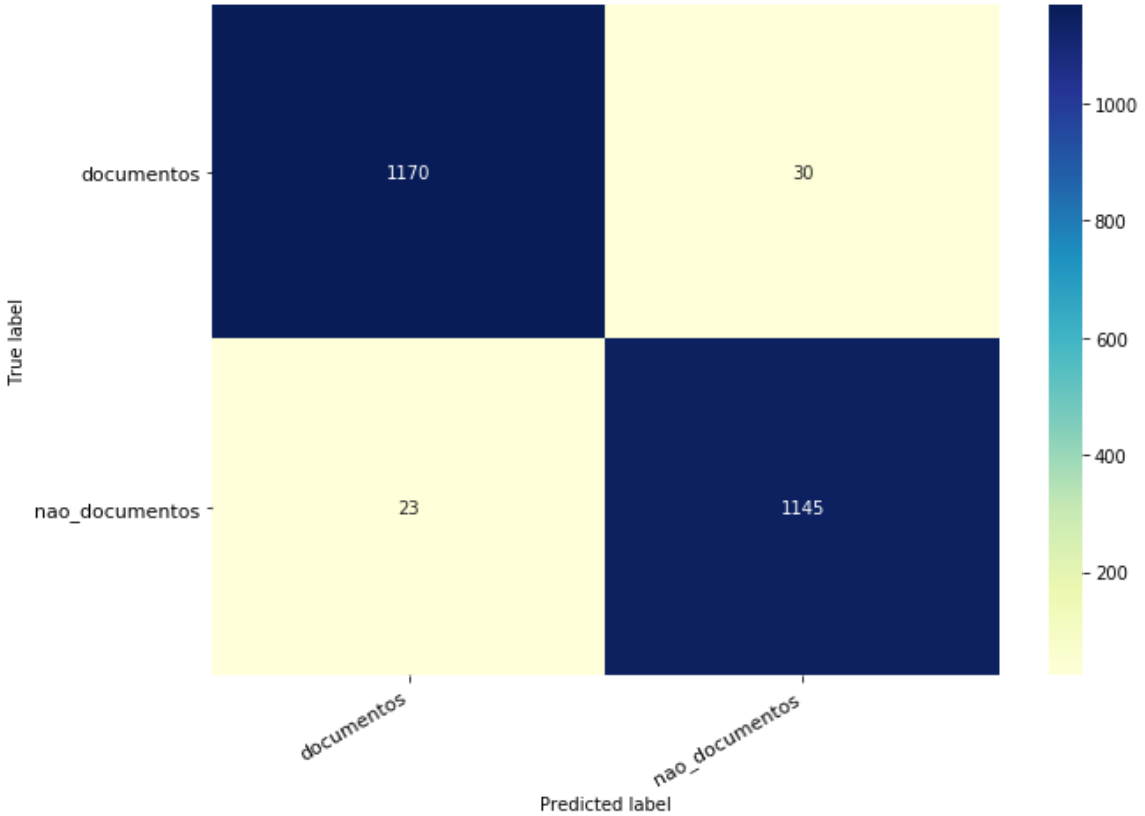




In [9]:

```
### 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, 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 2396 images belonging to 2 classes.
74/74 [==============================] - 134s 2s/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 1198 images belonging to 2 classes.
1198/1198 [==============================] - 79s 66ms/step
1198/1198 [==============================] - 72s 60ms/step
Test accuracy: 0.975
Test loss: 0.153
```