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 2397 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_04.h5')
```

```
Epoch 1/50
156/156 [==============================] - 761s 5s/step - loss: 0.4045 - a
cc: 0.8319 - val_loss: 0.3113 - val_acc: 0.8839
Epoch 2/50
156/156 [==============================] - 701s 4s/step - loss: 0.2720 - a
cc: 0.8982 - val_loss: 0.2416 - val_acc: 0.9163
Epoch 3/50
156/156 [==============================] - 642s 4s/step - loss: 0.2338 - a
cc: 0.9127 - val_loss: 0.1961 - val_acc: 0.9366
Epoch 4/50
156/156 [==============================] - 563s 4s/step - loss: 0.1919 - a
cc: 0.9343 - val_loss: 0.1557 - val_acc: 0.9488
Epoch 5/50
156/156 [==============================] - 546s 3s/step - loss: 0.1824 - a
cc: 0.9371 - val_loss: 0.1607 - val_acc: 0.9459
Epoch 6/50
156/156 [==============================] - 545s 3s/step - loss: 0.1674 - a
cc: 0.9437 - val_loss: 0.1596 - val_acc: 0.9425
Epoch 7/50
156/156 [==============================] - 542s 3s/step - loss: 0.1480 - a
cc: 0.9493 - val_loss: 0.1765 - val_acc: 0.9510
Epoch 8/50
156/156 [==============================] - 528s 3s/step - loss: 0.1416 - a
cc: 0.9535 - val_loss: 0.1158 - val_acc: 0.9658
Epoch 9/50
156/156 [==============================] - 530s 3s/step - loss: 0.1214 - a
cc: 0.9607 - val_loss: 0.1460 - val_acc: 0.9497
Epoch 10/50
156/156 [==============================] - 530s 3s/step - loss: 0.1182 - a
cc: 0.9633 - val_loss: 0.1111 - val_acc: 0.9598
Epoch 11/50
156/156 [==============================] - 529s 3s/step - loss: 0.1119 - a
cc: 0.9609 - val_loss: 0.0929 - val_acc: 0.9653
Epoch 12/50
156/156 [==============================] - 529s 3s/step - loss: 0.1041 - a
cc: 0.9663 - val_loss: 0.1143 - val_acc: 0.9670
Epoch 13/50
156/156 [==============================] - 531s 3s/step - loss: 0.0967 - a
cc: 0.9687 - val_loss: 0.0959 - val_acc: 0.9662
Epoch 14/50
156/156 [==============================] - 528s 3s/step - loss: 0.0885 - a
cc: 0.9720 - val_loss: 0.0884 - val_acc: 0.9679
Epoch 15/50
156/156 [==============================] - 528s 3s/step - loss: 0.0854 - a
cc: 0.9728 - val_loss: 0.0754 - val_acc: 0.9708
Epoch 16/50
156/156 [==============================] - 528s 3s/step - loss: 0.0771 - a
cc: 0.9738 - val_loss: 0.0975 - val_acc: 0.9700
Epoch 17/50
156/156 [==============================] - 528s 3s/step - loss: 0.0793 - a
cc: 0.9734 - val_loss: 0.0977 - val_acc: 0.9687
Epoch 18/50
156/156 [==============================] - 528s 3s/step - loss: 0.0843 - a
cc: 0.9736 - val_loss: 0.0837 - val_acc: 0.9734
Epoch 19/50
156/156 [==============================] - 532s 3s/step - loss: 0.0614 - a
cc: 0.9798 - val_loss: 0.1648 - val_acc: 0.9611
Epoch 20/50
156/156 [==============================] - 526s 3s/step - loss: 0.0728 - a
cc: 0.9750 - val_loss: 0.1023 - val_acc: 0.9742
Epoch 21/50
```

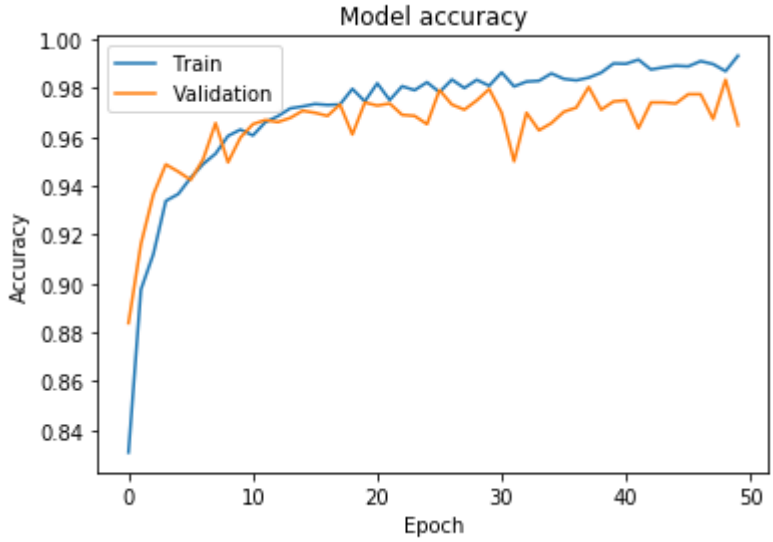
```
156/156 [==============================] - 528s 3s/step - loss: 0.0581 - a
cc: 0.9822 - val_loss: 0.1116 - val_acc: 0.9729
Epoch 22/50
156/156 [==============================] - 528s 3s/step - loss: 0.0710 - a
cc: 0.9754 - val_loss: 0.0816 - val_acc: 0.9738
Epoch 23/50
156/156 [==============================] - 533s 3s/step - loss: 0.0588 - a
cc: 0.9810 - val_loss: 0.0967 - val_acc: 0.9691
Epoch 24/50
156/156 [==============================] - 530s 3s/step - loss: 0.0636 - a
cc: 0.9794 - val_loss: 0.0883 - val_acc: 0.9687
Epoch 25/50
156/156 [==============================] - 528s 3s/step - loss: 0.0472 - a
cc: 0.9826 - val_loss: 0.1145 - val_acc: 0.9653
Epoch 26/50
156/156 [==============================] - 527s 3s/step - loss: 0.0561 - a
cc: 0.9786 - val_loss: 0.0853 - val_acc: 0.9793
Epoch 27/50
156/156 [==============================] - 527s 3s/step - loss: 0.0496 - a
cc: 0.9836 - val_loss: 0.0800 - val_acc: 0.9734
Epoch 28/50
156/156 [==============================] - 527s 3s/step - loss: 0.0576 - a
cc: 0.9804 - val_loss: 0.1047 - val_acc: 0.9712
Epoch 29/50
156/156 [==============================] - 528s 3s/step - loss: 0.0483 - a
cc: 0.9836 - val_loss: 0.0975 - val_acc: 0.9751
Epoch 30/50
156/156 [==============================] - 529s 3s/step - loss: 0.0457 - a
cc: 0.9812 - val_loss: 0.0916 - val_acc: 0.9797
Epoch 31/50
156/156 [==============================] - 527s 3s/step - loss: 0.0391 - a
cc: 0.9866 - val_loss: 0.1330 - val_acc: 0.9700
Epoch 32/50
156/156 [==============================] - 530s 3s/step - loss: 0.0615 - a
cc: 0.9808 - val_loss: 0.2835 - val_acc: 0.9501
Epoch 33/50
156/156 [==============================] - 526s 3s/step - loss: 0.0509 - a
cc: 0.9830 - val_loss: 0.1461 - val_acc: 0.9700
Epoch 34/50
156/156 [==============================] - 528s 3s/step - loss: 0.0458 - a
cc: 0.9832 - val_loss: 0.1290 - val_acc: 0.9628
Epoch 35/50
156/156 [==============================] - 528s 3s/step - loss: 0.0410 - a
cc: 0.9862 - val_loss: 0.1522 - val_acc: 0.9658
Epoch 36/50
156/156 [==============================] - 527s 3s/step - loss: 0.0402 - a
cc: 0.9840 - val_loss: 0.0930 - val_acc: 0.9704
Epoch 37/50
156/156 [==============================] - 525s 3s/step - loss: 0.0525 - a
cc: 0.9834 - val_loss: 0.1216 - val_acc: 0.9721
Epoch 38/50
156/156 [==============================] - 527s 3s/step - loss: 0.0505 - a
cc: 0.9844 - val_loss: 0.0792 - val_acc: 0.9805
Epoch 39/50
156/156 [==============================] - 526s 3s/step - loss: 0.0459 - a
cc: 0.9866 - val_loss: 0.1026 - val_acc: 0.9712
Epoch 40/50
156/156 [==============================] - 527s 3s/step - loss: 0.0245 - a
cc: 0.9902 - val_loss: 0.1127 - val_acc: 0.9746
Epoch 41/50
156/156 [==============================] - 528s 3s/step - loss: 0.0270 - a
```

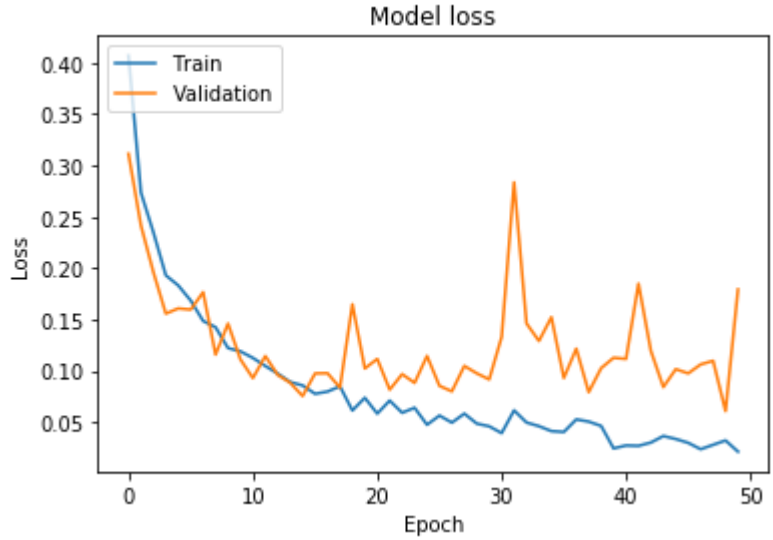
```
cc: 0.9902 - val_loss: 0.1116 - val_acc: 0.9751
Epoch 42/50
156/156 [==============================] - 525s 3s/step - loss: 0.0268 - a
cc: 0.9918 - val_loss: 0.1849 - val_acc: 0.9636
Epoch 43/50
156/156 [==============================] - 524s 3s/step - loss: 0.0300 - a
cc: 0.9878 - val_loss: 0.1193 - val_acc: 0.9742
Epoch 44/50
156/156 [==============================] - 526s 3s/step - loss: 0.0365 - a
cc: 0.9886 - val_loss: 0.0842 - val_acc: 0.9742
Epoch 45/50
156/156 [==============================] - 525s 3s/step - loss: 0.0332 - a
cc: 0.9894 - val_loss: 0.1016 - val_acc: 0.9738
Epoch 46/50
156/156 [==============================] - 531s 3s/step - loss: 0.0299 - a
cc: 0.9890 - val_loss: 0.0975 - val_acc: 0.9776
Epoch 47/50
156/156 [==============================] - 526s 3s/step - loss: 0.0235 - a
cc: 0.9912 - val_loss: 0.1062 - val_acc: 0.9776
Epoch 48/50
156/156 [==============================] - 527s 3s/step - loss: 0.0278 - a
cc: 0.9900 - val_loss: 0.1097 - val_acc: 0.9674
Epoch 49/50
156/156 [==============================] - 522s 3s/step - loss: 0.0318 - a
cc: 0.9872 - val_loss: 0.0608 - val_acc: 0.9835
Epoch 50/50
156/156 [==============================] - 524s 3s/step - loss: 0.0212 - a
cc: 0.9934 - val_loss: 0.1792 - val_acc: 0.9649
```

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: 0.993
Train loss: 0.021

Validation accuracy: 0.965
Validation loss: 0.179
```

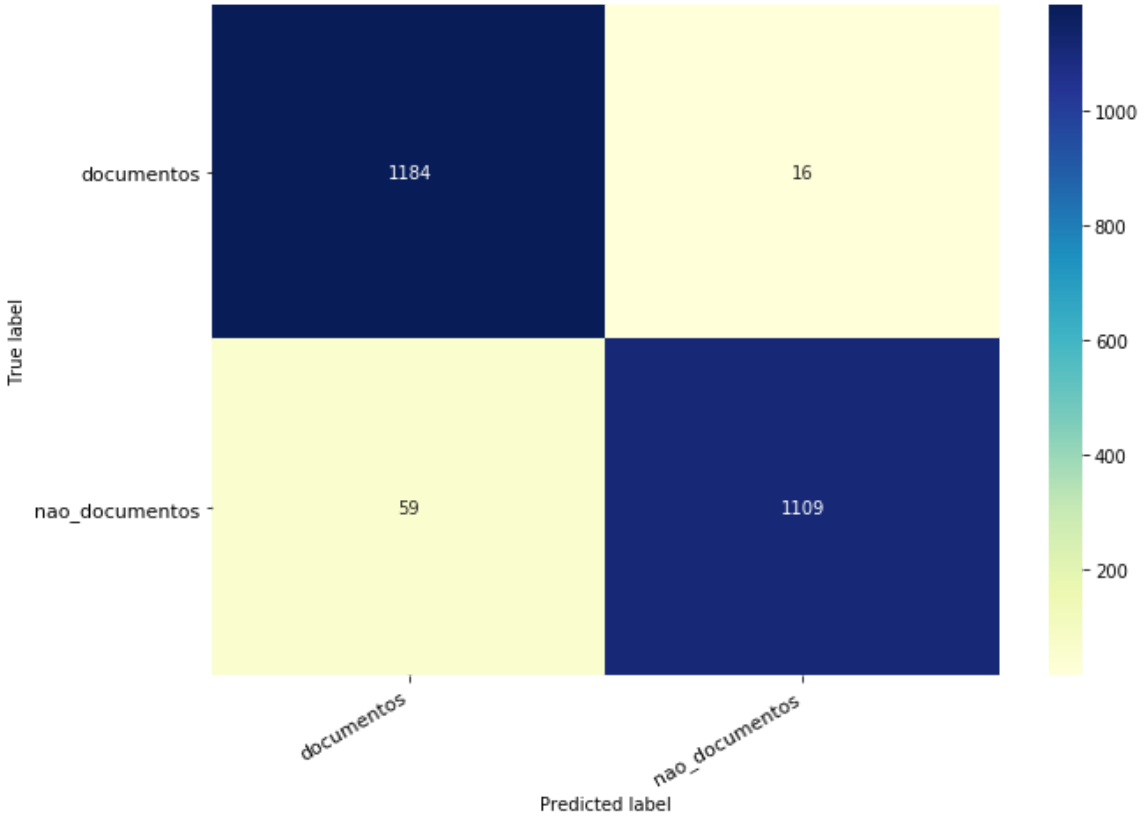




In [9]:

```
### 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 2397 images belonging to 2 classes.
74/74 [==============================] - 116s 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 1200 images belonging to 2 classes.
1200/1200 [==============================] - 73s 61ms/step
1200/1200 [==============================] - 81s 68ms/step
Test accuracy: 0.966
Test loss: 0.215
```