'''  
chord\_nn.py  
version 0.1  
author: R2\_B09  
'''  
**from** keras.models **import** Sequential, load\_model  
**from** keras.layers **import** Dense  
**import** keras.callbacks  
**import** numpy **as** NP  
**import** h5py  
**import** soleil **as** sol  
  
FIRST\_RUN = False  
INPUT\_VALS = NP.load("input\_dataset/samples/database/NP\_INPUT\_NEURON\_VALUES.npy")  
OUTPUT\_VALS = NP.load("input\_dataset/samples/database/NP\_OUTPUT\_NEURON\_VALUES.npy")  
VALIDATION\_INPUT\_VALS = NP.load("input\_dataset/samples/database/NP\_INPUT\_NEURON\_VALUES\_NOT\_SAMPLE.npy")  
VALIDATION\_OUTPUT\_VALS = NP.load("input\_dataset/samples/database/NP\_OUTPUT\_NEURON\_VALUES\_NOT\_SAMPLE.npy")  
  
**def** save\_history(mh):  
   MODE = 'w' **if** FIRST\_RUN **else** 'a'  
  
   **with** open("models/history/mse.txt", MODE) **as** ModelHistoryMSE:  
        ModelHistoryMSE.write(str(mh.history['mean\_squared\_error']))  
  
   **with** open("models/history/val\_acc.txt", MODE) **as** ModelHistoryVA:  
       ModelHistoryVA.write(str(mh.history['val\_acc']))  
  
   **with** open("models/history/val\_loss.txt", MODE) **as** ModelHistoryVL:  
       ModelHistoryVL.write(str(mh.history['val\_loss']))  
  
**def** checkpoint(verb = 1):  
   **return** keras.callbacks.ModelCheckpoint("models/chord\_identifier.h5", verbose=verb, monitor='val\_acc', save\_best\_only=True, mode='max')

**if** \_\_name\_\_ == "\_\_main\_\_":  
   VALIDATION\_INPUT = NP.load("input\_dataset/samples/database/NP\_INPUT\_VALIDATION.npy")  
   VALIDATION\_OUTPUT = NP.load("input\_dataset/samples/database/NP\_OUTPUT\_VALIDATION.npy")  
  
   **print**('INPUT\_VALS shape: ' + str(NP.shape(INPUT\_VALS)))  
   **print**('INPUT\_VALS shape of each element: ' + str(NP.shape(INPUT\_VALS[0])))  
   **print**('OUTPUT\_VALS shape: ' + str(NP.shape(OUTPUT\_VALS)))  
   **print**('OUTPUT\_VALS shape of each element: ' + str(NP.shape(OUTPUT\_VALS[0])))  
  
   **if** FIRST\_RUN:  
       chord\_identifier = Sequential()  
       chord\_identifier.add(Dense(14, input\_shape = (24,), activation = 'sigmoid'))  
       chord\_identifier.add(Dense(2))  
       chord\_identifier.add(Dense(2))  
       chord\_identifier.add(Dense(14, activation = 'softmax'))  
  
       chord\_identifier.compile(optimizer = 'sgd', loss = 'mean\_squared\_error', metrics = ['mse', 'accuracy'])  
       checkpointer = checkpoint()  
       callbacks\_list = [checkpointer]  
  
       h = chord\_identifier.fit(INPUT\_VALS, OUTPUT\_VALS, epochs = 800, verbose = 1, validation\_data = (VALIDATION\_INPUT, VALIDATION\_OUTPUT), callbacks=callbacks\_list)  
       save\_history(h)  
  
       sol.graph\_from\_History(things\_to\_graph=['acc', 'val\_acc'], MHObject=h, title="Model accuracy", ylabel="Accuracy", xlabel="Epoch", legendlist=['train', 'test'], legendloc = 'upper left')  
       sol.graph\_from\_History(things\_to\_graph=['loss', 'val\_loss'], MHObject=h, title="Model losses", ylabel="Loss", xlabel="Epoch", legendlist=['train', 'test'], legendloc = 'upper left')  
  
   **else**:  
       # recall and train  
       chord\_identifier = keras.models.load\_model("models/chord\_identifier.h5")  
       chord\_identifier.compile(optimizer='sgd', loss='mean\_squared\_error', metrics=['mse', 'accuracy'])  
       checkpointer = checkpoint()  
  
       h = chord\_identifier.fit(INPUT\_VALS, OUTPUT\_VALS, epochs=1000, verbose=1, validation\_data=(VALIDATION\_INPUT, VALIDATION\_OUTPUT), callbacks=callbacks\_list)  
       save\_history(h)  
  
       sol.graph\_from\_History(things\_to\_graph=['acc', 'val\_acc'], MHObject=h, title="Model accuracy", ylabel="Accuracy", xlabel="Epoch", legendlist=['train', 'test'], legendloc = 'upper left'