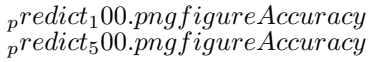
$_{m}odel() - >$
$Sequential:$
$Definemodel.model =$
$Sequential()model.add(ZeroPadding2D(padding =$
$pad, input_{s}hape =$
$(input_{s}ize[0], input_{s}ize[1], 1), name =$
$"padding_{l}ayer"))model.add(Conv2D(conv_{f}ilter_{n}um, conv_{k}ernel_{s}ize, activation =$
$"relu", padding =$
$"valid", kernel_{i}nitializer =$
$"he_{u}niform", input_{s}hape =$
$(30, 30, 1), name =$
$"convolution_{l}ayer"))model.add(MaxPooling2D(pool_{s}ize, name =$
$"max_{p}ooling_{l}ayer"))model.add(Flatten(name =$
$"flatten_{l}ayer"))model.add(Dense(10, activation =$
$"softmax", name =$
$"dense_{l}ayer"))Compilemodel.model.Compile(optimizer =$
$Adam(), loss =$
$"categorical_{c}rossentropy", metrics =$
$["accuracy"])Returnmodel.returnmodel$
`conv_weights.h`
`dense_weights.h`
`definitions.h`
`conv_weights.h`
`dense_weights.h`
`definitions.h`
`in.dat`
`out.dat`
`gen_data.ipynb`
$_{t}o_{d}ense_{s}tream, intfilter, hls ::$
$stream <$
$float >$
$dense_{t}o_{s}oftmax_{s}tream)floatflat_{v}alue; floatdense_{a}rray[DENSE_{S}IZE] = 0;$
$_{f}or_{f}lat :$
$for(inti =$
$0; i <$
$FLAT_{S}IZE/FILTERS; ++$
$i)flat_{v}alue = flat_{t}o_{d}ense_{s}tream.read();$
$_{S}IZE; ++$
$d)intindex = filter * (FLAT_{S}IZE/FILTERS) + i; dense_{a}rray[d] += dense_{w}eights[index][d] * flat_{v}alue;$
$_{S}IZE; ++$
$j)dense_{t}o_{s}oftmax_{s}tream.write(dense_{a}rray[j]);$
$_{l}ayer(floatpad_{i}mg0[PAD_{I}MG_{R}OWS][PAD_{I}MG_{C}OLS], floatpad_{i}mg1[PAD_{I}MG_{R}OWS][PAD_{I}MG_{C}OLS], floatpad$
$stream <$
$float >$
$conv_{t}o_{p}ool_{s}treams[FILTERS])convolution(pad_{i}mg0, 0, conv_{t}o_{p}ool_{s}treams[0]); convolution(pad_{i}mg1, 1, conv_{t}o_{p}ool_{s}trea$
`flattening`
$_{t}o_{f}lat_{s}tream, hls ::$
$stream <$
$float >$
$flat_{t}o_{d}ense_{s}tream)flat_{f}or_{r}ows : for(intr = 0; r < POOL_{I}MG_{R}OWS; + + r)flat_{f}or_{c}ols : for(intc = 0; c < POOL_{I}M$
`CNN`
$_{i}n[IMG_{R}OWS][IMG_{C}OLS], floatprediction[DIGITS])/ * * * * * * * *Pre - processingdata. * * * * * * * */$
$_{i}mg0[PAD_{I}MG_{R}OWS][PAD_{I}MG_{C}OLS] =$
$0; normalization_{a}nd_{p}adding(img_{i}n, pad_{i}mg0);$
$_{S}YNTHESIS_{p}rintf("Paddedimage."); print_{p}ad_{i}mg(pad_{i}mg); endifendif$
$_{i}mg1[PAD_{I}MG_{R}OWS][PAD_{I}MG_{C}OLS]; floatpad_{i}mg2[PAD_{I}MG_{R}OWS][PAD_{I}MG_{C}OLS]; floatpad_{i}mg3[PAD_{I}M$
$_{f}or_{r}ows :$
$for(inti =$
$0; i <$
$PAD_{I}MG_{R}OWS; ++$
$i)clone_{f}or_{c}ols :$
$for(intj =$
$0; j <$
$PAD_{I}MG_{C}OLS; ++$
$j)pad_{i}mg1[i][j] = pad_{i}mg0[i][j]; pad_{i}mg2[i][j] = pad_{i}mg0[i][j]; pad_{i}mg3[i][j] = pad_{i}mg0[i][j];$
$_{s}ection(pad_{i}mg0, pad_{i}mg1, pad_{i}mg2, pad_{i}mg3, prediction);$
`CNN`
$_{R}esult_{1}.pngfigure()$
$_{R}esult_{2}.pngfigure()$
$_{R}esult_{3}.pngfigure()$


$_{p}redict_{1}00.pngfigureAccuracy$
$_{p}redict_{5}00.pngfigureAccuracy$