About Displacement

Works' name

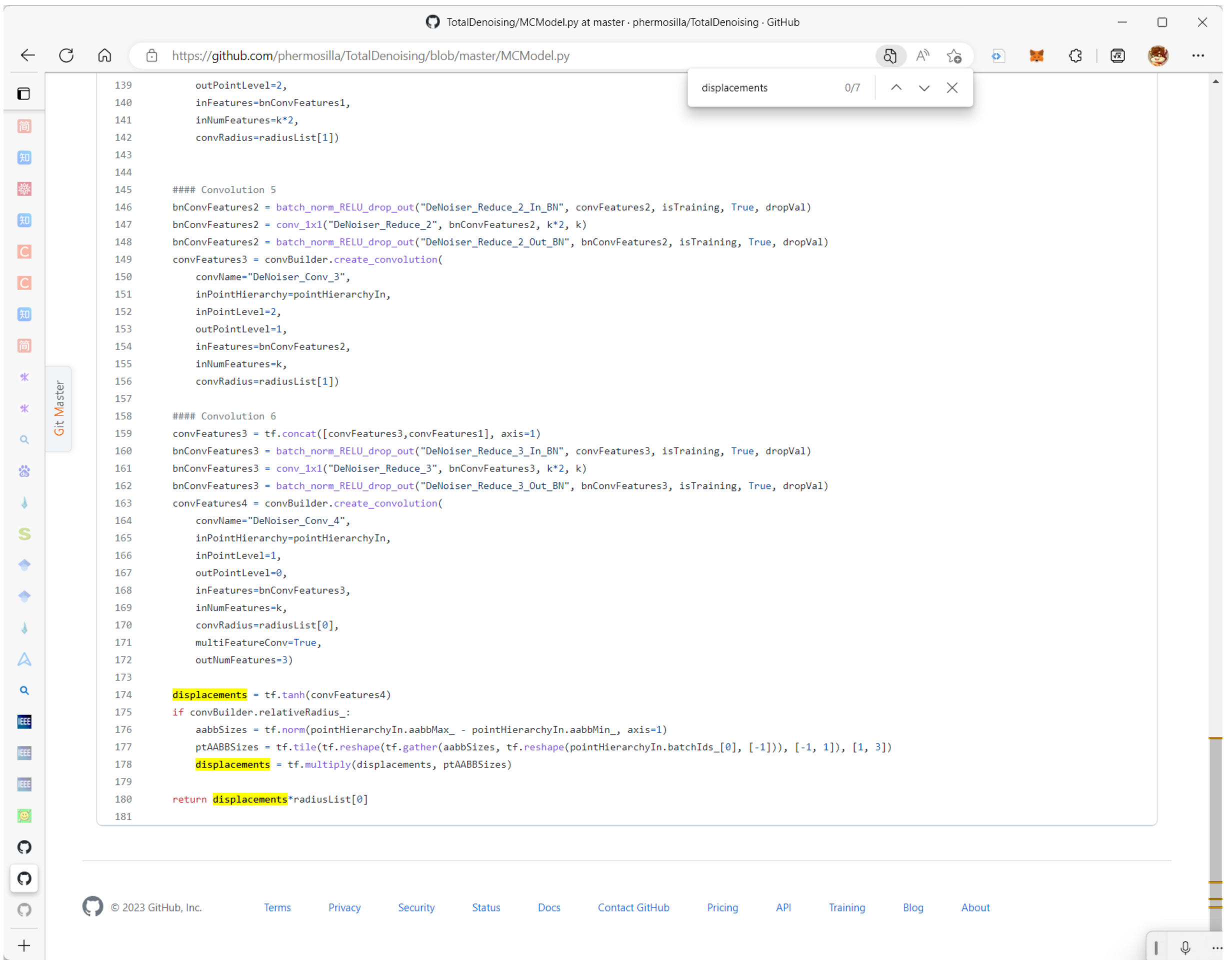
Net Name	Paper Name
PCN	POINTCLEANNET: Learning to Denoise and Remove Outliers from Dense Point Clouds
Total	Total Denoising: Unsupervised Learning of 3D Point Cloud Cleaning
3DPCD	3D POINT CLOUD DENOISING VIA DEEP NEURAL NETWORK BASED LOCAL SURFACE ESTIMATION
GPDNet	Learning Graph-Convolutional Representations for Point Cloud Denoising

About

Net Name	Output	Target
PCN	Point	Point Min length
Total	Displacement	compare (Input Point + Displacement) & GT
3DPCD	404	404
GPDNet	noise (same as displacement)	compare(MSE) (Input Point - ϵ noise) & GT

Code

Total

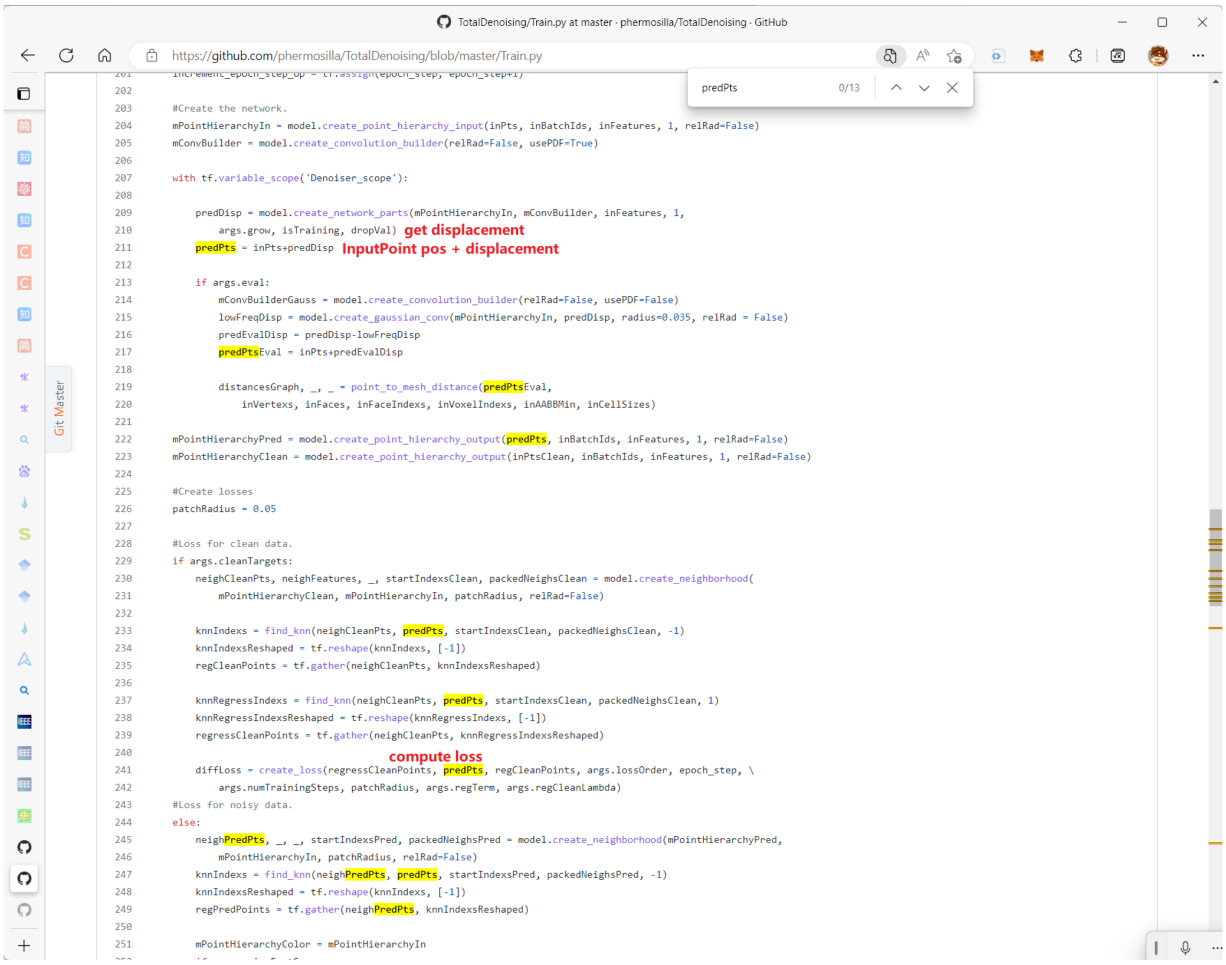

TotalDenoising/MCModel.py at master · phermosilla/TotalDenoising · GitHub

https://github.com/phermosilla/TotalDenoising/blob/master/MCModel.py

displacements 0/7

```python
            outPointLevel=2,
            inFeatures=bnConvFeatures1,
            inNumFeatures=k*2,
            convRadius=radiusList[1])


        #### Convolution 5
        bnConvFeatures2 = batch_norm_RELU_drop_out("DeNoiser_Reduce_2_In_BN", convFeatures2, isTraining, True, dropVal)
        bnConvFeatures2 = conv_1x1("DeNoiser_Reduce_2", bnConvFeatures2, k*2, k)
        bnConvFeatures2 = batch_norm_RELU_drop_out("DeNoiser_Reduce_2_Out_BN", bnConvFeatures2, isTraining, True, dropVal)
        convFeatures3 = convBuilder.create_convolution(
            convName="DeNoiser_Conv_3",
            inPointHierarchy=pointHierarchyIn,
            inPointLevel=2,
            outPointLevel=1,
            inFeatures=bnConvFeatures2,
            inNumFeatures=k,
            convRadius=radiusList[1])

        #### Convolution 6
        convFeatures3 = tf.concat([convFeatures3,convFeatures1], axis=1)
        bnConvFeatures3 = batch_norm_RELU_drop_out("DeNoiser_Reduce_3_In_BN", convFeatures3, isTraining, True, dropVal)
        bnConvFeatures3 = conv_1x1("DeNoiser_Reduce_3", bnConvFeatures3, k*2, k)
        bnConvFeatures3 = batch_norm_RELU_drop_out("DeNoiser_Reduce_3_Out_BN", bnConvFeatures3, isTraining, True, dropVal)
        convFeatures4 = convBuilder.create_convolution(
            convName="DeNoiser_Conv_4",
            inPointHierarchy=pointHierarchyIn,
            inPointLevel=1,
            outPointLevel=0,
            inFeatures=bnConvFeatures3,
            inNumFeatures=k,
            convRadius=radiusList[0],
            multiFeatureConv=True,
            outNumFeatures=3)

        displacements = tf.tanh(convFeatures4)
        if convBuilder.relativeRadius_:
            aabbSizes = tf.norm(pointHierarchyIn.aabbMax_ - pointHierarchyIn.aabbMin_, axis=1)
            ptAABBSizes = tf.tile(tf.reshape(tf.gather(aabbSizes, tf.reshape(pointHierarchyIn.batchIds_[0], [-1])), [-1, 1]), [1, 3])
            displacements = tf.multiply(displacements, ptAABBSizes)

        return displacements*radiusList[0]
```

© 2023 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

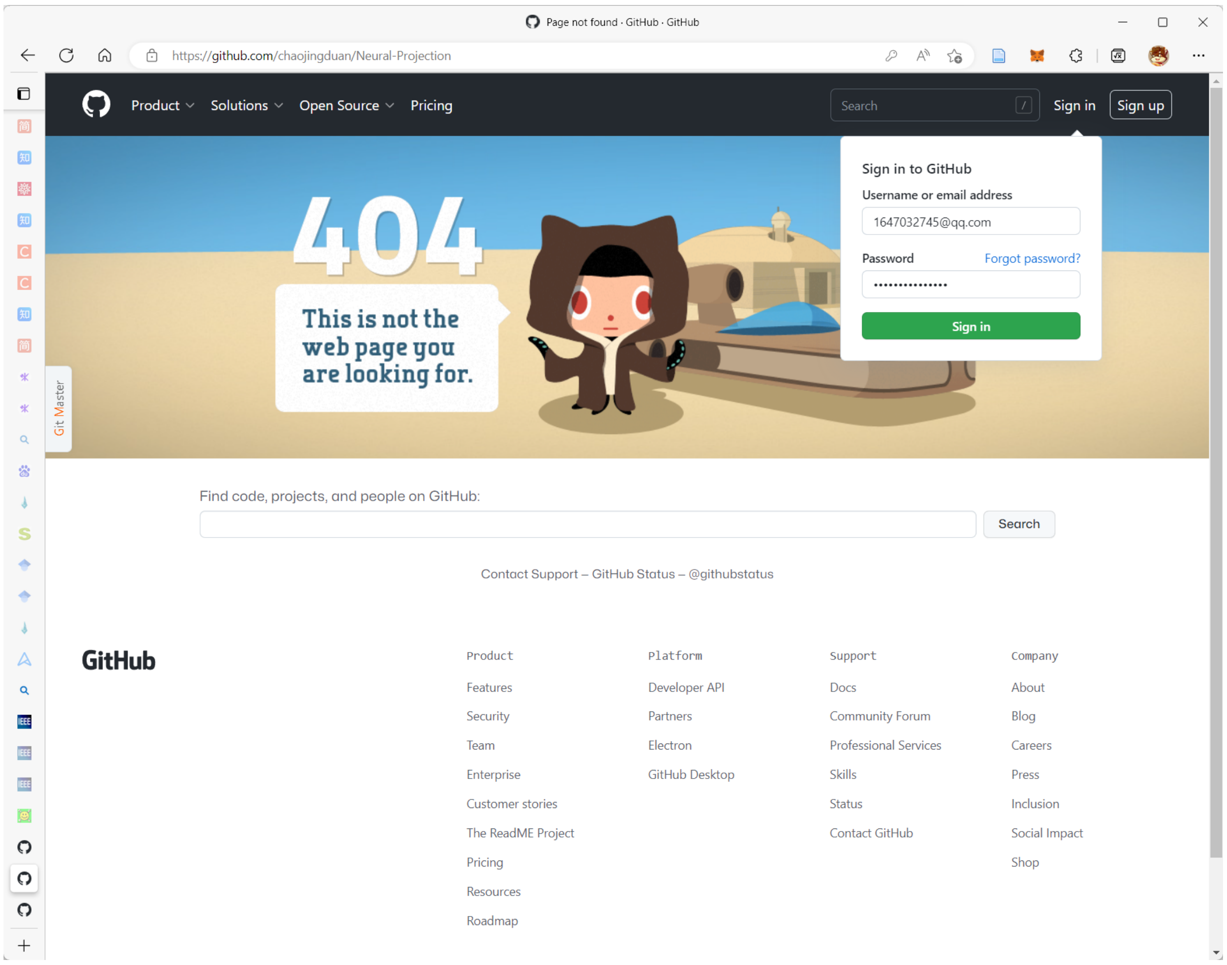

TotalDenoising/Train.py at master · phermosilla/TotalDenoising · GitHub

https://github.com/phermosilla/TotalDenoising/blob/master/Train.py

predPts 0/13

```python
    increment_epoch_step_op = tf.assign(epoch_step, epoch_step+1)

    #Create the network.
    mPointHierarchyIn = model.create_point_hierarchy_input(inPts, inBatchIds, inFeatures, 1, relRad=False)
    mConvBuilder = model.create_convolution_builder(relRad=False, usePDF=True)

    with tf.variable_scope('Denoiser_scope'):

        predDisp = model.create_network_parts(mPointHierarchyIn, mConvBuilder, inFeatures, 1,
            args.grow, isTraining, dropVal)  get displacement
        predPts = inPts+predDisp  InputPoint pos + displacement

        if args.eval:
            mConvBuilderGauss = model.create_convolution_builder(relRad=False, usePDF=False)
            lowFreqDisp = model.create_gaussian_conv(mPointHierarchyIn, predDisp, radius=0.035, relRad = False)
            predEvalDisp = predDisp-lowFreqDisp
            predPtsEval = inPts+predEvalDisp

            distancesGraph, _, _ = point_to_mesh_distance(predPtsEval,
                inVertexs, inFaces, inFaceIndexs, inVoxelIndexs, inAABBMin, inCellSizes)

    mPointHierarchyPred = model.create_point_hierarchy_output(predPts, inBatchIds, inFeatures, 1, relRad=False)
    mPointHierarchyClean = model.create_point_hierarchy_output(inPtsClean, inBatchIds, inFeatures, 1, relRad=False)

    #Create losses
    patchRadius = 0.05

    #Loss for clean data.
    if args.cleanTargets:
        neighCleanPts, neighFeatures, _, startIndexsClean, packedNeighsClean = model.create_neighborhood(
            mPointHierarchyClean, mPointHierarchyIn, patchRadius, relRad=False)

        knnIndexs = find_knn(neighCleanPts, predPts, startIndexsClean, packedNeighsClean, -1)
        knnIndexsReshaped = tf.reshape(knnIndexs, [-1])
        regCleanPoints = tf.gather(neighCleanPts, knnIndexsReshaped)

        knnRegressIndexs = find_knn(neighCleanPts, predPts, startIndexsClean, packedNeighsClean, 1)
        knnRegressIndexsReshaped = tf.reshape(knnRegressIndexs, [-1])
        regressCleanPoints = tf.gather(neighCleanPts, knnRegressIndexsReshaped)
                                        compute loss
        diffLoss = create_loss(regressCleanPoints, predPts, regCleanPoints, args.lossOrder, epoch_step, \
            args.numTrainingSteps, patchRadius, args.regTerm, args.regCleanLambda)
    #Loss for noisy data.
    else:
        neighPredPts, _, _, startIndexsPred, packedNeighsPred = model.create_neighborhood(mPointHierarchyPred,
            mPointHierarchyIn, patchRadius, relRad=False)
        knnIndexs = find_knn(neighPredPts, predPts, startIndexsPred, packedNeighsPred, -1)
        knnIndexsReshaped = tf.reshape(knnIndexs, [-1])
        regPredPoints = tf.gather(neighPredPts, knnIndexsReshaped)

        mPointHierarchyColor = mPointHierarchyIn
```

3DPCD


Page not found · GitHub · GitHub

https://github.com/chaojingduan/Neural-Projection

Product Solutions Open Source Pricing Search Sign in Sign up

404

This is not the web page you are looking for.

Sign in to GitHub

Username or email address: 1647032745@qq.com

Password: ••••••••••••••• Forgot password?

Sign in

Find code, projects, and people on GitHub:

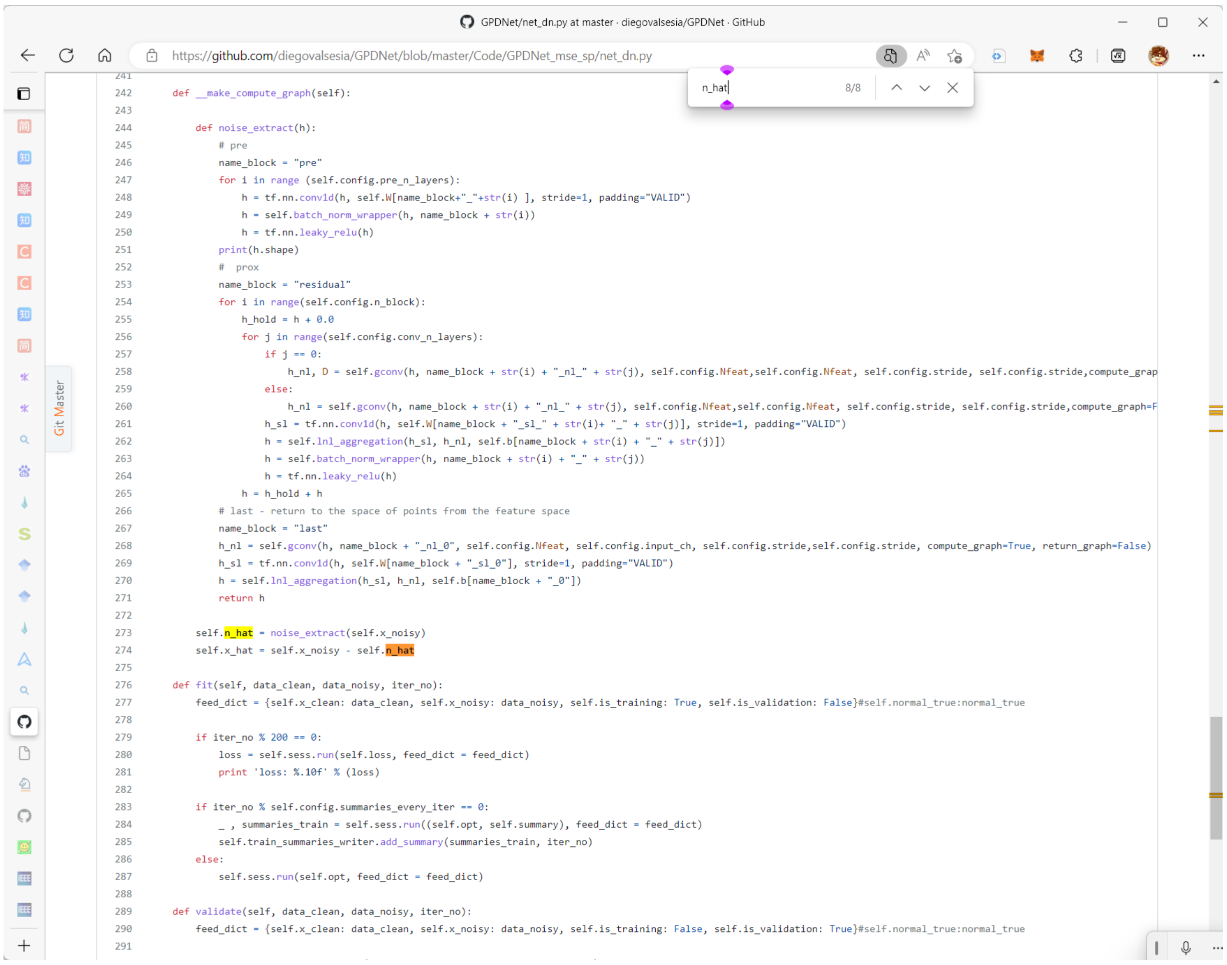
Search

Contact Support – GitHub Status – @githubstatus

GitHub

Product	Platform	Support	Company
Features	Developer API	Docs	About
Security	Partners	Community Forum	Blog
Team	Electron	Professional Services	Careers
Enterprise	GitHub Desktop	Skills	Press
Customer stories		Status	Inclusion
The ReadME Project		Contact GitHub	Social Impact
Pricing			Shop
Resources			
Roadmap			

GPDNet



GPDNet/net_dn.py at master · diegovalsesia/GPDNet · GitHub

https://github.com/diegovalsesia/GPDNet/blob/master/Code/GPDNet_mse_sp/net_dn.py

```python
    def __make_compute_graph(self):

        def noise_extract(h):
            # pre
            name_block = "pre"
            for i in range (self.config.pre_n_layers):
                h = tf.nn.conv1d(h, self.W[name_block+"_"+str(i) ], stride=1, padding="VALID")
                h = self.batch_norm_wrapper(h, name_block + str(i))
                h = tf.nn.leaky_relu(h)
            print(h.shape)
            #  prox
            name_block = "residual"
            for i in range(self.config.n_block):
                h_hold = h + 0.0
                for j in range(self.config.conv_n_layers):
                    if j == 0:
                        h_nl, D = self.gconv(h, name_block + str(i) + "_nl_" + str(j), self.config.Nfeat,self.config.Nfeat, self.config.stride, self.config.stride,compute_grap
                    else:
                        h_nl = self.gconv(h, name_block + str(i) + "_nl_" + str(j), self.config.Nfeat,self.config.Nfeat, self.config.stride, self.config.stride,compute_graph=F
                    h_sl = tf.nn.conv1d(h, self.W[name_block + "_sl_" + str(i)+ "_" + str(j)], stride=1, padding="VALID")
                    h = self.lnl_aggregation(h_sl, h_nl, self.b[name_block + str(i) + "_" + str(j)])
                    h = self.batch_norm_wrapper(h, name_block + str(i) + "_" + str(j))
                    h = tf.nn.leaky_relu(h)
                h = h_hold + h
            # last - return to the space of points from the feature space
            name_block = "last"
            h_nl = self.gconv(h, name_block + "_nl_0", self.config.Nfeat, self.config.input_ch, self.config.stride,self.config.stride, compute_graph=True, return_graph=False)
            h_sl = tf.nn.conv1d(h, self.W[name_block + "_sl_0"], stride=1, padding="VALID")
            h = self.lnl_aggregation(h_sl, h_nl, self.b[name_block + "_0"])
            return h

        self.n_hat = noise_extract(self.x_noisy)
        self.x_hat = self.x_noisy - self.n_hat

    def fit(self, data_clean, data_noisy, iter_no):
        feed_dict = {self.x_clean: data_clean, self.x_noisy: data_noisy, self.is_training: True, self.is_validation: False}#self.normal_true:normal_true

        if iter_no % 200 == 0:
            loss = self.sess.run(self.loss, feed_dict = feed_dict)
            print 'loss: %.10f' % (loss)

        if iter_no % self.config.summaries_every_iter == 0:
            _ , summaries_train = self.sess.run((self.opt, self.summary), feed_dict = feed_dict)
            self.train_summaries_writer.add_summary(summaries_train, iter_no)
        else:
            self.sess.run(self.opt, feed_dict = feed_dict)

    def validate(self, data_clean, data_noisy, iter_no):
        feed_dict = {self.x_clean: data_clean, self.x_noisy: data_noisy, self.is_training: False, self.is_validation: True}#self.normal_true:normal_true

```