binary cross entropy with L2 regularization

def binary\_crossentropy\_with\_l2(y\_true, y\_pred, l2\_reg=0.01):

cross\_entropy = tf.keras.losses.binary\_crossentropy(y\_true, y\_pred)

l2\_loss = l2\_reg \* tf.reduce\_sum([tf.nn.l2\_loss(var) for var in tf.trainable\_variables()])

loss = cross\_entropy + l2\_loss

return tf.keras.backend.mean(loss)

In the code above, we first compute the binary cross entropy using the **tf.keras.losses.binary\_crossentropy** function. Then, we compute the L2 regularization loss by iterating over all the trainable variables and computing their L2 norm using the **tf.nn.l2\_loss** function. Finally, we add the cross entropy loss and the L2 regularization loss together, and return their mean using **tf.keras.backend.mean**.

You can adjust the strength of the regularization by changing the **l2\_reg** parameter.

formula for binary cross entropy with l2 regularization and lovasz loss

def lovasz\_hinge(y\_true, y\_pred, alpha=1):

y\_true = tf.cast(y\_true, tf.float32)

y\_pred = tf.cast(y\_pred, tf.float32)

signs = 2 \* y\_true - 1

errors = tf.maximum(0., 1. - signs \* y\_pred)

errors\_sorted, permutation = tf.nn.top\_k(errors, k=tf.shape(errors)[0], name="descending\_sort")

gt\_sorted = tf.gather(y\_true, permutation)

grad = \_lovasz\_grad(gt\_sorted)

loss = tf.tensordot(tf.nn.relu(errors\_sorted), tf.stop\_gradient(grad), 1)

return loss \* alpha

def \_lovasz\_grad(gt\_sorted):

gts = tf.reduce\_sum(gt\_sorted)

intersection = gts - tf.cumsum(gt\_sorted)

union = gts + tf.cumsum(1 - gt\_sorted)

jaccard = 1. - intersection / union

jaccard = tf.concat((jaccard[0:1], jaccard[1:] - jaccard[:-1]), 0)

return jaccard

The **lovasz\_hinge** function takes the true labels **y\_true** and predicted logits **y\_pred** as input, along with the optional **alpha** parameter that controls the weighting of the Lovasz loss.

The function first computes the signs of the true labels (-1 or 1), and then computes the hinge loss between the predicted logits and the signs of the true labels. It then sorts the hinge losses in descending order and computes the gradient of the sorted true labels with respect to the loss, and uses this gradient to compute the Lovasz loss.

The **\_lovasz\_grad** function computes the gradient of the sorted true labels with respect to the Lovasz loss, which is used in the computation of the loss itself.

Note that there are other implementations of the Lovasz loss, such as **lovasz\_softmax**, which is used when the predicted logits are passed through a softmax activation.

def binary\_crossentropy\_with\_l2\_and\_lovasz(y\_true, y\_pred, l2\_reg=0.01, alpha=1):

binary\_loss = tf.keras.backend.mean(tf.keras.losses.binary\_crossentropy(y\_true, y\_pred))

l2\_loss = l2\_reg \* tf.reduce\_sum([tf.nn.l2\_loss(var) for var in tf.trainable\_variables()])

lovasz\_loss = lovasz\_hinge(y\_true, y\_pred, alpha=alpha)

loss = binary\_loss + l2\_loss + lovasz\_loss

return loss

In the above formula, we first compute the binary cross entropy loss using the **tf.keras.losses.binary\_crossentropy** function, followed by the L2 regularization loss, which is the sum of the L2 norm of all trainable variables multiplied by the regularization coefficient **l2\_reg**.

Then we add the Lovasz loss, which is computed by calling the **lovasz\_hinge** function with the true labels **y\_true** and predicted labels **y\_pred** as input, along with the **alpha** parameter that controls the slope of the Lovasz function. The Lovasz loss is a measure of the distance between the predicted labels and the true labels, and it is designed to optimize for the IoU metric in segmentation tasks.

Finally, we add up the three losses to get the total loss and return it. You can adjust the regularization strength and the alpha parameter of the Lovasz loss to balance the contributions of the different loss components to the overall loss. Note that the **lovasz\_hinge** function is used instead of **lovasz\_binary\_crossentropy** as the latter assumes the probabilities are already passed through the sigmoid activation, while **lovasz\_hinge** works with raw logits.