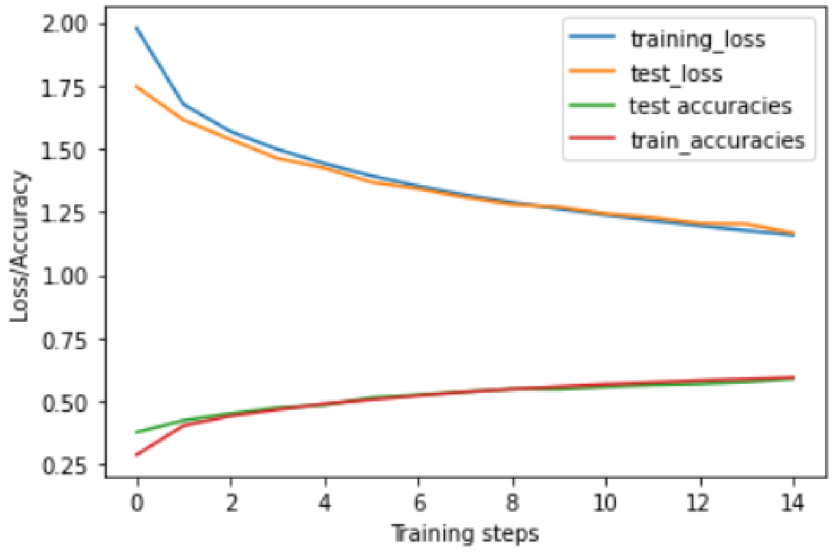
5 different optimization techniques - Group24

Our original model

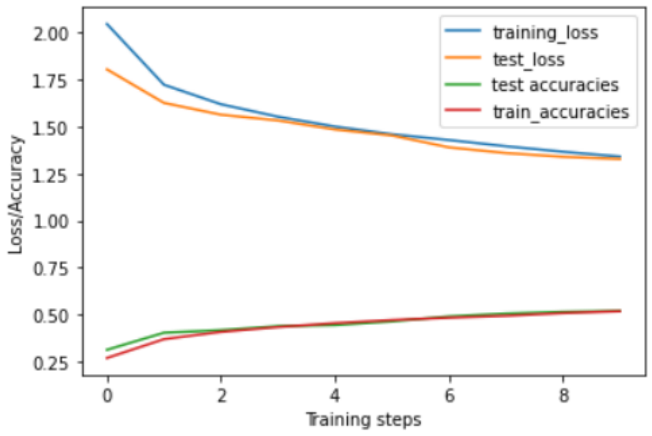
- Basic CNN layers - 4 layers
- Epoch 15
- Learning rate 0.00001
- Categorical accuracy
- Categorical cross-entropy
- Optimizer Adam



Hello, everybody and welcome to our adventure today. We are going to try Different stuff from our tool Box to prevent overfitting. This is our original model and where we are going to start. Ready? Let's start!

Residual CNN model

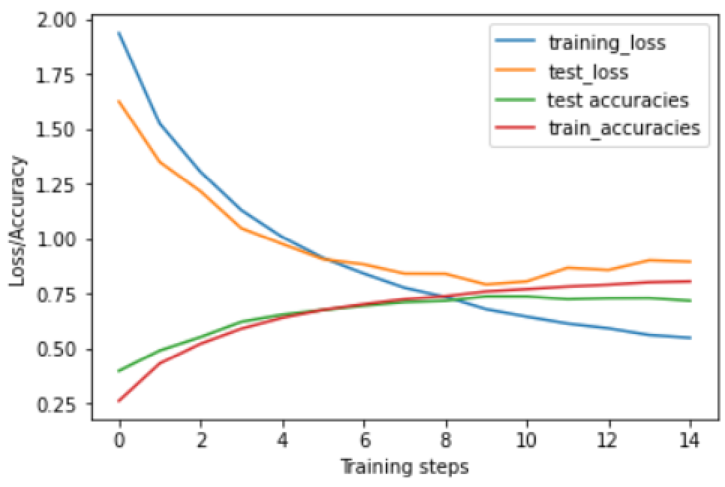
- Residual CNN layers - 3 blocks
- Epoch 10
- Learning rate 0.00001
- Categorical accuracy
- Categorical cross-entropy
- Optimizer Adam



Mhm… I don't think we changed a lot. It was a good idea to try more advance CNN network like ResNet, however We didn't get the best results. Maybe we can try using different Optimizer. Because I heard that for Res Net Momentum is better. Let's check it.

<u>Residual CNN model</u>

- Residual CNN layers - 3 blocks
- Epoch 15
- Learning rate 0.01
- Categorical accuracy
- Categorical cross-entropy
- Optimizer Momentum (momentum = 0.99)



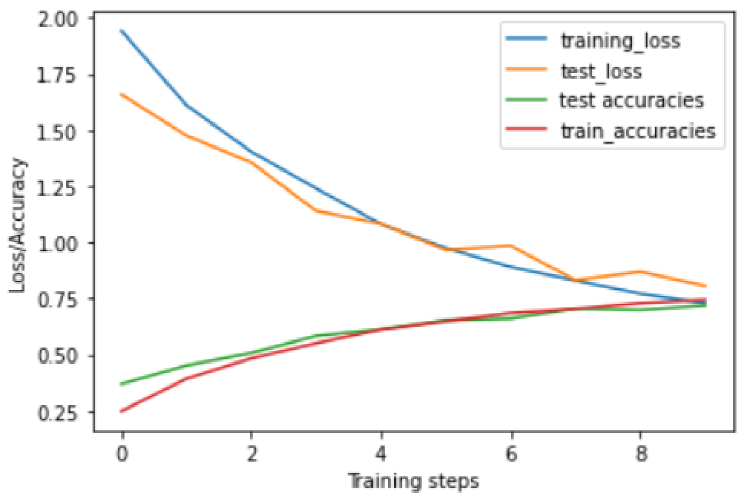
We can see that our model is overfitting now. **We observe overfitting if the model performs well on the training data but does not perform well on the test data**. So we will try to minimize our hyperparameters → Epoch=10 + learning rate = 0.0001 Let's see what we got

Why did we use Momentum with ResNet?

Momentum is very good for ResNet architecture for image classification problem. ResNet is very deep network and many researchers say that ADAM is the best, but my practical experience showed the Momentum is the best for training ResNet. As we can see we proved that.

Residual CNN model

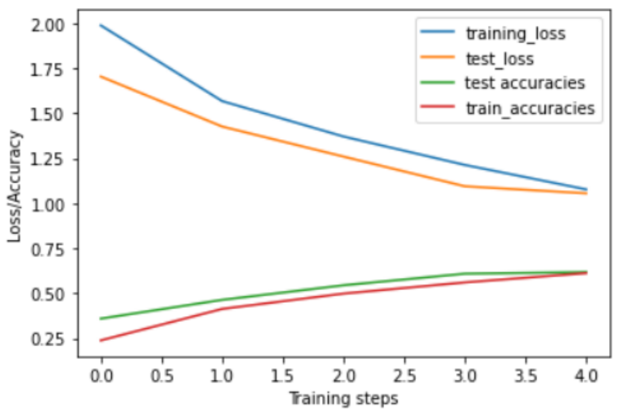
- Residual CNN layers - 3 blocks
- Epoch 10
- Learning rate 0.001
- Categorical accuracy
- Categorical cross-entropy
- Optimizer Momentum (momentum = 0.99)



Sadly we can see that our model still overfits the data. Not to worry we have other techniques in our tool box. For example let's try batch Normalization. Maybe that is just what we need.

Basic CNN model with Batch normalization layer

- Basic CNN layers - 4
- Epoch 5
- Learning rate 0.001
- Categorical accuracy
- Categorical cross-entropy
- Optimizer Adam



-

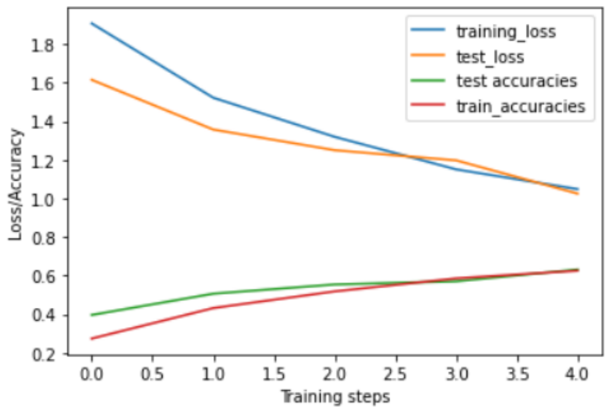
Yeah!
We decided to go back to our Basic CNN model and apply two batch normalization layers. As we can see our model looks little better than before.
But we still have work to do.
Now let's try to optimize our model with different technique. But what else do we have left in our tool box?
Did you guess it already? Yes, let's do a drop out layer.

Why did we use batch normalization?

Reduces overfitting. Batch normalization has a regularizing effect since it adds noise to the inputs of every layer. This discourages overfitting since the model no longer produces deterministic values for a given training example alone

Basic CNN model with Drop out layer

- Basic CNN layers - 4
- Drop out layer - 1
- Epoch 5
- Learning rate 0.001
- Categorical accuracy
- Categorical cross-entropy
- Optimizer Adam



Noooo! Okay, maybe that is not what we were looking for. Let's stay with our previous model for now. How does that sound?

Why did we try dropout layer?

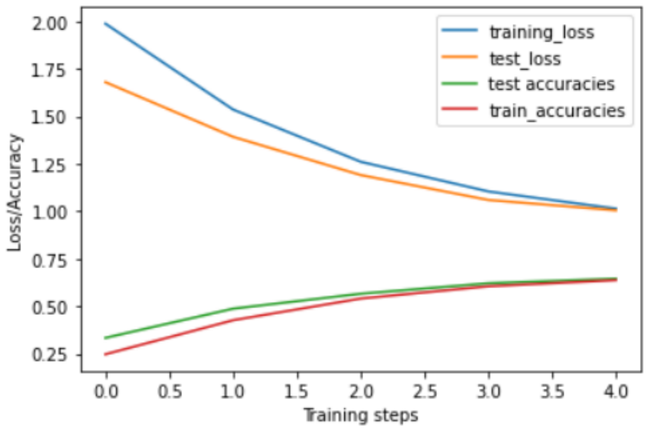
Dropout layer should prevent all neurons in a layer from synchronously optimizing their weights. This adaptation, made in random groups, prevents all the neurons from converging to the same goal, thus decorrelating the weights.

Why may our dropout layer not work?

First, dropout is generally less effective at regularizing convolutional layers. The reason? **Since convolutional layers have few parameters, they need less regularization to begin with**. Furthermore, because of the spatial relationships encoded in feature maps, activations can become highly correlated.

Basic CNN model with penalty L2

- Basic CNN layers - 4
- Epoch 5
- Learning rate 0.001
- Categorical accuracy
- Categorical cross-entropy
- Optimizer Adam



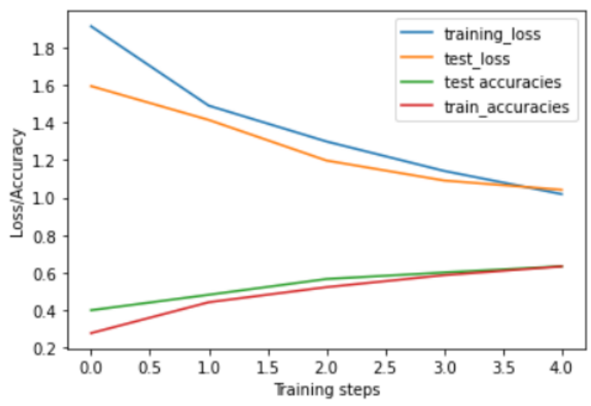
Phew! We got on the right track again. However, our model doesn't look that stable. Let's try something interesting and do 2 in 1. We are going to Implement penalties and batch normalization together! Exciting, right?

What did we do and why with the L2 penalty?

Regularization optimizes a model by penalizing complex models, therefore minimizing loss and complexity. Thus, this forces our neural network to be simpler. Here we will use an L2 regularizer, as it is the most common and is more stable than an L1 regularizer. Here we'll add a regularizer to the second and third layers of our network with a learning rate (Lr) of 0.001.

Basic CNN model with penalty L2 and batch normalization

- Basic CNN layers - 4
- Batch normalization layer - 1
- Epoch 5
- Learning rate 0.001
- Categorical accuracy
- Categorical cross-entropy
- Optimizer Adam



Okay, we see that maybe that wasn't the best idea ever, but we at least try. Even though we couldn't find the best solution. We optimized our model and we saw our techniques in action and that was pretty cool, wasn't it? So we say bye for now and see you next time. 🙂