What is BatchNorm? And why?

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

Why:

**Improved Training Speed**

Normalization helps the gradients to propagate well, which means the weights and biases can be updated more effectively. This leads to faster training of neural networks.

**Stabilizes Training**

Normalization helps to stabilize the training process by reducing the impact of initialization and helps in training deep networks.

**Reduces Internal Covariate Shift**

Internal covariate shift refers to the change in the distribution of the inputs of layers during training, which slows down training and makes it harder to train deep networks. BatchNorm reduces this shift, making training more stable.

**Regularization Effect**

BatchNorm has a slight regularization effect similar to dropout. It adds some noise to each layer's activations, so the model becomes less sensitive to the specific weights, reducing overfitting to some extent.

**Easier Hyperparameter Search**

BatchNorm makes it easier to search for hyperparameters as the model is less sensitive to bad choices of learning rate or weight initialization.

**Less Dependence on Initialization**

The normalization operation reduces the sensitivity to the initial starting weights, making it possible to start with a wider range of initial weights, making the training process more robust.

**Allows Higher Learning Rates**

By reducing the internal covariate shift, BatchNorm allows for the use of higher learning rates, speeding up the training process.

**Enables Training of Activations That Don't Saturation Easily**

Before BatchNorm, activations like sigmoid were not used in deep networks due to the vanishing gradient problem. However, normalization allows for a wider range of activation functions to be used.