# Additional Neural Network Concepts

## Advanced Optimization

Adam optimization algorithm, which improves the training process of neural networks compared to traditional gradient descent.

### Adam Algorithm Intuition

- **Adam**: **Adaptive Moment Estimation** and adjusts the learning rate automatically based on the behavior of the parameters during training.
- It uses different learning rates for each parameter, allowing for more efficient convergence to the minimum of the cost function.
- If $w_j$ (or $b$) keeps moving in the same direction, **increase** $\alpha_j$.
- If $w_j$ (or $b$) keeps oscillating , **reduce** $\alpha_j$.

### MINIST Adam

```
model = Sequential([
  Dense(units=25, activation='relu'),
  Dense(units=15, activation='relu'),
  Dense(units=20, activation='linear'),
                ])
# it has some initial learning rate
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-3),
    loss=SparseCategoricalCrossEntropy(from_logits=True))

model.fit(X, Y,epochs=100)
```

- It is more robust to the choice of initial learning rate, making it a preferred option among practitioners for training neural networks.

## Additional Layer Types

# Recap: Dense Layer

Every neuron in the layer gets its inputs all the activations from the previous layer.

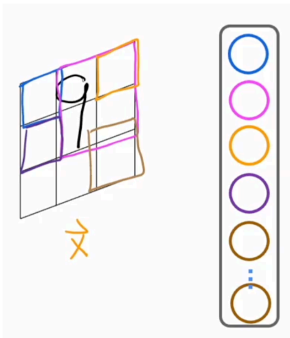
$a_j^{[l]} = g(\vec{w}_j^{[l]} \cdot \vec{a}^{[l-1]} + b_j^{[l]})$

# Convolutional Layer

If **each neuron (unit)** in a layer only looks at **a subset (part)** of the **outputs** from the previous layer, this approach can offer certain advantages.

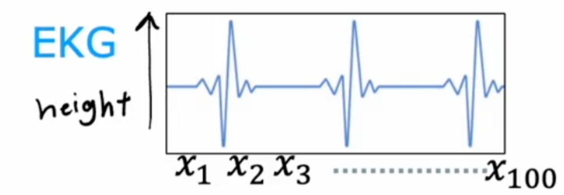
- Faster computations.
- Less training data (Less prone to overfitting).

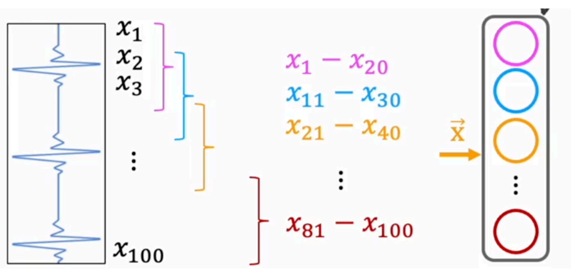
The example illustrates how a convolutional layer processes a handwritten digit image by having each neuron look at a limited rectangular area of the image.




# Convolutional Neural Network

In a different context, the convolutional layer can be applied to **EKG signals (electrocardiograms)**, where neurons analyze small windows of the signal to classify heart conditions. (it is 1D array not 2D array example)



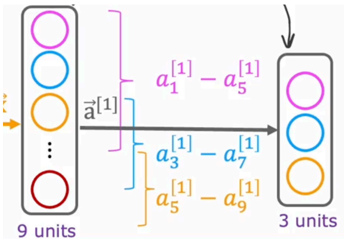




- Each neuron in the convolutional layer examines a small window of the EKG signal:
    - For example, the first neuron might analyze the first 20 data points, while the second

neuron looks at points 11 to 30.

## Second Convolutional Layer:

- The second layer takes the activations from the first layer as its input.
- Each neuron in this layer also looks at a limited number of activations from the previous layer, rather than all of them.
- For example:
  - The first neuron in the second layer might look at the first $5$ activations from the first layer.
  - The second neuron might analyze activations $3$ to $7$.
  - The third neuron might focus on activations $5$ to $9$.



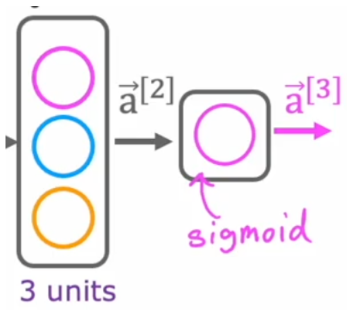

- This method allows the network to identify patterns in the EKG signal that can indicate heart conditions, improving classification accuracy.

💡 **Purpose of the Second Layer:**

- This structure allows the network to learn more complex features by combining the information from the first layer's activations.
- By focusing on different subsets of activations, the second layer can capture higher-level patterns in the EKG signal that may be indicative of specific heart conditions.

## Final Output Layer

- The final output layer (often a sigmoid layer) takes the activations from the last convolutional layer.
- This output layer typically makes a binary classification decision, such as determining whether the patient has a heart disease or not, based on the combined information from the previous layers.




💡 **QUES:** The Adam optimizer is the recommended optimizer for finding the optimal parameters of the model. How do you use the Adam optimizer in TensorFlow?

- The call to `model.compile()` will automatically pick the best optimizer, whether it is gradient descent, Adam or something else. So there's no need to pick an optimizer manually.
- When calling `model.compile`, set `optimizer=tf.keras.optimizers.Adam(learning_rate=1e-3)`. 👍🏻👍🏻
- The Adam optimizer works only with Softmax outputs. So if a neural network has a Softmax output layer, TensorFlow will automatically pick the Adam optimizer.
- The call to `model.compile()` uses the Adam optimizer by default

*Explain:* Correct. Set the optimizer to Adam.

```
model = Sequential([
    Dense(units=25, activation='relu'),
```

```
    Dense(units=15, activation='relu'),
    Dense(units=20, activation='linear'),
                                    ])
# it has some initial learning rate
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-3),
        loss=SparseCategoricalCrossEntropy(from_logits=True))

model.fit(X, Y,epochs=100)
```