**Jan/18/2025**

Today I started looking at different topics to research. I know I want to do something with AI, but I’m not sure what yet.

**Feb/2/2025**

I think I want to do something with hunger insecurities. I think I might focus on hunger insecurities in first world countries just for the fact that there is already so much research and data for hunger in other countries. I think I want to call it Fighting Famine for alliteration.

**Feb/8/2025**

I started researching neural networks today. I think I got a lot done, I will focus more on CNNs tomorrow.

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

NEURAL NETWORKS

Neural Networks are a complex system of different interconnecting layers made up of nodes. Neural Networks were made in part to emulate the human brain, with the end goal of producing a machine that can learn and almost think just as well as a human being. These networks can take in information and have it pass through multiple different layers, to try and recognize different patterns and features to meet the end goal it was programmed to meet.

HOW DO THEY WORK?

LAYERS

A neural network has three main layers, the input layer, the hidden layers, and the output layer.

The input layer is where the neural network receives raw data, this could be an image or words.

The hidden layers are responsible for processing the information by using weighted connections. Each ‘neuron’ or node in the hidden layer performs some type of mathematical operation and passes the result of that equation to the next layer. The more layers the more complex the neural network, but you must consider how many layers will help us achieve the target accuracy? How many neurons will help us achieve the target accuracy? How many connections should be retained from the previous layer’s neuron?

The output layer is responsible for generating the final answer, this could be a prediction or a classification.

The “neurons” or nodes are the basic unit of a neural network. Each node is a simple mathematical function that takes multiple inputs and then factors in weights and biases with the end goal of producing an output. Each node has its own linear regression model composed of input data, weights, a bias and an output. The formula would look similar to this.

**∑wixi + bias = w1x1 + w2x2 + w3x3 + bias**

**output = f(x) = 1 if ∑w1x1 + b>= 0; 0 if ∑w1x1 + b < 0**

WEIGHTS, BIAS AND ACTIVATION FUNCTION

Weights are used to determine the strength of connections between each node. These weights decide how much influence a neuron’s input has on the next node. This is done by multiplying the input by the weight as it passes through the connection.

Biases are an additional constant value added to the weighted sum of inputs. It helps the model make better predictions and allows shifting the activation function left or right, meaning the bias controls when a neuron activates. Without bias the neuron activates only when the input and the weight are large enough, but with a bias the neuron can still activate even if the input and weight are quite small or even when the input and weight are zero. Bias can control when the neuron activates by either requiring it to be a bigger value so it shifts the function to the right, or it can allow the value to be smaller by shifting the function to the left.

The activation function is similar to a gatekeeper of sorts as it will decide whether or not a node’s output will pass on to the next layer. These activation functions allow neural networks to learn and identify more complex problems and allow them to behave in a non-linear fashion. This is quite useful because most data or images that come from the real world are full of nuance and variability, and are therefore quite non-linear. The activation function decides which neurons should be active or inactive. Some common activation functions are the Sigmoid which is an s-shaped curve and is used in probabilities. ReLU is used in most deep learning models like CNN and RNN. Softmax is used for multi-class classification.

These three critical elements work together to ensure that the output is as precise as possible.

OPTIMIZATION ALGORITHM

An optimization algorithm is used to update the weights and the biases of a neural network based on the loss function. The gradient descent is a widely used optimization algorithm that can adjust the weights and biases in the direction of steepest descent.

PARAMETERS

Trainable parameters: These parameters include the biases of neurons in every layer except for the input layer, and the weights of the connections between the nodes. These parameters are considered trainable because they can be changed and updated.

Hyperparameters – Thes are fixed values, which are then fine-tuned through experimentation, to achieve the lowest possible cost value. Design-related hyperparameters include many hidden layers, the number of nodes, the types of activation function, the optimization algorithm used, and the loss function used. Some hyperparameters are adjusted after the design finalization, they are the learning rate which controls the magnitude of updates to the weights. High learning weights can lead to missing optimal weight values, and a lower learning weight could make the training process a lot longer. The regularization prevents overfitting by reducing the amount of learned information. The batch size controls the number of samples used to update the weights during training. Batch learning which uses a group of samples, which can aid in improving the performance and decrease outliers.

TRAINING

The first step in training is Initialization where the neural network will randomly assign weights and biases.

Forward propagation is when you pass input data through the network to get a desired output. You input the data and each node receives the input and multiplies it by its weights, it adds bias, and applies an activation function. The output of one layer will then go on to become the input for the next layer and so on and so forth. Then finally the output layer reaches its final prediction.

After we have the network’s prediction, we compare it to the real answer, to measure how wrong or right the neural network is using a loss function. A loss function comes up with a quantitative value between the neural networks predicted output and the true value. Some common loss functions are cross-entropy loss, this is used when dealing with probabilities. Intuition is another loss function, so if the neural networks prediction is far from the correct label, the loss is therefore high and if the prediction is closer the loss is low. These are loss functions used in classification. Another loss function used for regression, is mean squared error, penalizes large errors more than small errors. The formula for calculating the loss-function, is also known as mean squared error (MSE).

* ***i* represents the index of the sample,**
* **y-hat is the predicted outcome,**
* **y is the actual value, and**
* ***m* is the number of samples.**

**𝐶𝑜𝑠𝑡 𝐹𝑢𝑛𝑐𝑡𝑖𝑜𝑛= 𝑀𝑆𝐸=1/2𝑚 ∑129\_(𝑖=1)^𝑚▒(𝑦 ̂^((𝑖) )−𝑦^((𝑖) ) )^2**

The next step is backpropagation. Once we know the error, we can then update the weights and the biases to reduce the error. This is done with backpropagation, which will usually involve gradient descent and partial derivatives. The gradient is how much each weight on each node has contributed to the mistake. We then take the derivative of the loss function with respect to each weight and bias; this will then tell us how to change each weight to decrease the error. The neural network will then calculate how much each node contributed to the total mistake. This error is propagated from the output layer to the input layer. The weights and the biases are then updated so when we run the program we will have a lower level of error.

We will then redo these steps over and over multiple epochs, each time the neural network will keep on getting better and better at what it is trying to learn.

Validation is then used to help confirm that the neural network is adept in learning patterns that generalize well to new data. This is a test to see if the model has actually learned instead of just memorized. Validation is done by introducing a completely new dataset that was not seen in training. This will tune hyperparameters and check performance. During training you would track the network's validation loss where we should see a decrease in if the neural network is improving. You also track validation accuracy which should increase if the model is doing well. To improve the result of validation you can use some techniques like early stopping, so you would stop when the model stops improving. You would use regularization to prevent overfitting. You can hyperparameter tuning, to adjust the network parameters for the best possible performance. A final test will be performed to gauge how well the network can perform on new unseen data.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Feb/9/2025**

I did part of CNN’s and I forgot to do history yesterday so I did that today.

HISTORY OF NEURAL NETWORKS

The idea of a thinking machine can be traced back to the Ancient Greeks, what we see as true thinking machines weren’t really seen till the mid 1900s. In 1943 Warren S. McCulloch and Walter Pitts published together “A logical calculus of the ideas immanent in nervous activity”. This research aimed to understand the mysteries of the human brain or more specifically how the human brain could produce complex and incredible patterns through connected neurons. One of the main ideas that would come out of this work was the comparison of human neurons with binary threshold to Boolean logic (0/1 or true/false statements). In 1958, Frank Rosenblatt developed the perceptron “The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain”. He advanced Warren S. McCulloch and Walter Pitts research a step further by introducing weights to the equation. Leveraging IBM 704(), Rosenblatt got a computer to learn how to distinguish cards marked left v.s cards marked on the right. In 1974, Paul Werbos was the first person in the US to note the use of backpropagation within neural networks with his PhD thesis. In 1989, Yann LeCun published a paper illustrating how the use of constraints in backpropagation, and how its integration into neural networks can be very useful to train algorithms. This research helped recognize neural networks as hand-written zip code digits provided by the U.S. Postal Service.

CONVOLUTIONAL NEURAL NETWORK (CNN)

A convolutional neural network or CNN’s is a type of deep learning model that processes images, videos, and other spatial data. They are able to preserve spatial relationships using very special layers, instead of treating each pixel separately. CNN’s use convolution to detect patterns like, different textures, edges and shapes. There are four main layers: the convolutional layers, activation layers, pooling layers, and the fully connected layers.

The convolution layer handpicks important patterns from an image. It moves across the image, looking out for specific shapes like edges, textures and patterns. It works similarly to how you as a human will pick out certain features from an image to identify what it is. It makes processing images much more efficient as it doesn’t have to spend time looking over every single individual pixel. After scanning the entire image for certain patterns, it creates a feature map which is a new version of the image where only the specific details such as edges, shapes, and textures are highlighted. The number of filters used affects the depth of the networks output. Stride is the distance, or number of pixels. Zero padding is when the filters do not fit the image. This will then set everything outside the input to zero, which produces a larger or equally sized output. There are three types of padding, the valid padding is when there simply is no padding. Same padding ensures the input image is the same size as the output layer. Full padding increases the size of the output image.

The activation layer makes decisions in a CNN. It decides which patterns are the most relevant, and which ones are unimportant. It helps filter out any detail that is irrelevant and makes everything easier to identify the image because you can see the details much more clearly. The most common activation function in CNN’s is ReLU, it quickly filters out the weaker information and will only keep the most important information. This makes CNN’s much faster and much more efficient.

**Feb/16/2025**

I finished the rest of CNN’s today.

The pooling layer resizes the images making them much smaller yet they still keep the most important details. It pools all the most important information together, it is a summary tool that makes CNN’s faster and much, much more efficient.

The fully connected layer is used for decision-making, combining information, and learning complex patterns. The fully connected layer is the layer where all the nodes are connected to every node in the previous layer. It is often the very last step before a decision is made by the network. The fully connected layer, first flattens the image into a 1D vector. In the fully connected layer, each node is tasked with looking at all the features, and determining just how important each piece is to the final decision. The layer takes all the information from the previous layer and combines it with a weighted sum. This is where all the learning happens during training, the weights of connections get adjusted to make the right decision. The node then applies an activation function to its output so that it can decide what value to pass forward. This layer then combines everything together to make the final decision.

The output layer outputs class probabilities using Softmax (for multi-class classification) or Sigmoid (for binary classification)

CNNs work well for images as they can capture spatial hierarchies, parameter sharing which makes them computationally efficient, they are translation-invariant, meaning they can detect any features regardless of their position in the image.

**Feb/22/2025**

I did RNN’s today. I found out that RNNs aren’t actually the best version out there and that they aren’t used anymore.

RECURRENT NEURAL NETWORK (RNN)

A recurrent neural network or an RNN are a type of neural network that has sequential data. They are designed to process sequential data. RNNs have a memory mechanism that allows them to keep information from previous time steps, which makes them much better equipped to solve tasks involving ordered data or time-dependent data. This can be natural language processing (Machine translation and text generation) or speech recognition.

STRUCTURE OF RNNs

The input layer is where all data enters the network. The data is entered piece by piece.

The hidden layers are the memory of the neural network. In a regular neural network which treats each input separately, an RNN remembers what it has seen before and uses that information to help and influence the current step. Each time the RNN processes a new word it updates its memory and combines what it has just received as input and what it has previously learned.

After processing everything the RNN will produce an output. There are different outputs for different tasks, it could be a translated sentence, or it could maybe be the weather forecast.

DIFFERENT VARIATIONS OF RNNs

One-to-One: Each one input has one output which is identical to a normal neural network.

One-to-Many: One input generates a sequence of outputs.

Many-to-One: Many inputs that lead to one output.

Many-to-Many: Many inputs that lead to many outputs.

PROBLEMS

They struggle to remember important information from further back in the sequence. When gradients become too small during the process of backpropagation, earlier layers fail to learn properly. Gradients can also become too large, leading to unstable training.

**Feb/23/2025**

I did LSTMs today because they are apparently the better version of RNNs.

LONG SHORT-TERM MEMORY NETWORKS (LSTMs)

Long short-term memory networks or LSTMs are an advanced type of recurrent neural network (RNN), designed to fix the long-term dependencies that appear in RNNs.

We need LSTMs because they solve the problems that come with RNNs, like the vanishing gradient problems. LSTMs solve this by having more advanced memory mechanisms. Unlike regular RNNs which have a single hidden state, LSTMs introduce a cell state that enables long-term memory, along with three gates that control information flow.

STRUCTURE OF LSTMs

The cell state is the memory of the LSTM, this state carries important information, and allows the network to retain or forget past data as needed. LSTMs also have the usual hidden states that come with the standard RNNs, that are used for short-term memory.

The first gate is the forget gate, which decides what past information should be “forgotten" by the cell state. It looks at the previous hidden state and the current input and outputs a value between zero (completely) and 1 (fully retained).

The input gate decides what new information should be stored in the cell state. It has a filter that will decide which values will be updated. I had a candidate update which creates new memory values to be stored. The forget gate and the input gate work hand in hand to update cell state, useless information is deleted and important information is added.

The output gate determines what the next hidden state should be, the hidden state is the passed to the next step and is used in the process of making predictions

TYPES OF LSTMs

There are certain variations of LSTMs that may aid in improving efficiency. The bidirectional LSTM – Processes data in both forward and backward directions to be able to look at things from two perspectives; the past and the future. The peephole LSTM – Allows gates to directly look at the cell state for better and more informed decisions. The gated recurrent unit is a simplified version of LSTM with fewer gates, making it faster while still effective for long-term dependencies.

**Mar/8/2025**

I started using AI in agriculture and in grocery stores today. I also started hunger in first world countries.

HUNGER IN FIRST WORLD COUNTRIES
 In 2023 the U.S. 47 million people in the United States experienced food insecurity, 14 million of the people are children. In 2023 in Canada 22.9% of our population in the ten provinces lived in a food-insecure household. That amounts to 9.16 million Canadians which includes 2.1 million children. Yet in 2023 46.5% of all food in Canada is wasted. 41% of this is avoidable. 23% of avoidable food waste is from best before dates. In the U.S about 30%-40% of food is wasted every year. Food waste doesn’t just affect people but also contributes to 11% of the world’s emissions of gases like methane, carbon dioxide and chlorofluorocarbons.

NEURAL NETWORKS IN AGRICULTURE

Canada is said to have lost the equivalent of seven small farms a day for 20 years. Each year we have less and less farmland, due to factors such as urbanization and soil quality. This forces us to convert forest land into farmland, which has so many environmental repercussions. Which means that we need farms to be more efficient and more effective, to not just help our farmers but to also help feed our people. If there is a surplus then prices will be lower. Using neural networks we can help farmers save billions of dollars each year, by using image recognition with convolutional neural networks, check for pests, soil composition, fertilizer management, livestock health and wellbeing. AI can also help make data based decisions, by factoring and monitoring the weather conditions, the application of fertilizer and pesticides. AI can also detect leaks and irrigation systems. Using convolutional neural networks you could detect mold, rot, insects, or other threats to crop health.

















AI IN GROCERY STORES

In Canada, we rely heavily on best before dates, yet these dates aren’t always a correct indicator of food safety. Best before dates simply indicate when they are at their best quality. After their best before dates they may lose their flavour or may become stale. Yet in Canada we often associate the best before date with the safety of the food, this leads to food waste. So if in grocery stores we were able to instead sell foods that have passed their expiry date at a discount price we can help 9.16 million Canadians with their food insecurity. Where does AI come into all of this? Well having to manage all of this can be hard for many different grocery stores and make them less willing to implement this system. So if we were able to use AI to manage all of the different products, and there best before date. This will not only help many people. It will also help decrease food waste and in turn help decrease our emissions. Which will not just benefit our country but the world in general. We don’t have to just implement this in Canada we can also implement this around the world which will be helping millions even billions of people.

 **Mar/8/2025**

I put everything in the website today, and did the problem, method, acknowledgement, and looked over my citations. I’m not putting the citations in this logbook but I will link the doc where I put the citations. <https://docs.google.com/document/d/1a0tU2QkOq585x5vY6bqe3VmugAHQDKUay7GR2qD6E-w/edit?tab=t.0>

This logbook will be at the science fair where it will be fully updated, because I still have to rewrite this for the trifold and have to make my video. So this is going to be the last entry for the online version but I will update it.

NEURAL NETWORKS

Neural Networks are a complex system of different interconnecting layers made up of nodes. Neural Networks were made in part to emulate the human brain, with the end goal of producing a machine that can learn and almost think just as well as a human being. These networks can take in information and have it pass through multiple different layers, to try and recognize different patterns and features to meet the end goal it was programmed to meet.

HOW DO THEY WORK?

LAYERS

A neural network has three main layers, the input layer, the hidden layers, and the output layer.

The input layer is where the neural network receives raw data, this could be an image or words.

The hidden layers are responsible for processing the information by using weighted connections. Each ‘neuron’ or node in the hidden layer performs some type of mathematical operation and passes the result of that equation to the next layer. The more layers the more complex the neural network, but you must consider how many layers will help us achieve the target accuracy? How many neurons will help us achieve the target accuracy? How many connections should be retained from the previous layer’s neuron?

The output layer is responsible for generating the final answer, this could be a prediction or a classification.

The “neurons” or nodes are the basic unit of a neural network. Each node is a simple mathematical function that takes multiple inputs and then factors in weights and biases with the end goal of producing an output. Each node has its own linear regression model composed of input data, weights, a bias and an output. The formula would look similar to this.

**∑wixi + bias = w1x1 + w2x2 + w3x3 + bias**

**output = f(x) = 1 if ∑w1x1 + b>= 0; 0 if ∑w1x1 + b < 0**

WEIGHTS, BIAS AND ACTIVATION FUNCTION

Weights are used to determine the strength of connections between each node. These weights decide how much influence a neuron’s input has on the next node. This is done by multiplying the input by the weight as it passes through the connection.

Biases are an additional constant value added to the weighted sum of inputs. It helps the model make better predictions and allows shifting the activation function left or right, meaning the bias controls when a neuron activates. Without bias the neuron activates only when the input and the weight are large enough, but with a bias the neuron can still activate even if the input and weight are quite small or even when the input and weight are zero. Bias can control when the neuron activates by either requiring it to be a bigger value so it shifts the function to the right, or it can allow the value to be smaller by shifting the function to the left.

The activation function is similar to a gatekeeper of sorts as it will decide whether or not a node’s output will pass on to the next layer. These activation functions allow neural networks to learn and identify more complex problems and allow them to behave in a non-linear fashion. This is quite useful because most data or images that come from the real world are full of nuance and variability, and are therefore quite non-linear. The activation function decides which neurons should be active or inactive. Some common activation functions are the Sigmoid which is an s-shaped curve and is used in probabilities. ReLU is used in most deep learning models like CNN and RNN. Softmax is used for multi-class classification.

These three critical elements work together to ensure that the output is as precise as possible.

OPTIMIZATION ALGORITHM

An optimization algorithm is used to update the weights and the biases of a neural network based on the loss function. The gradient descent is a widely used optimization algorithm that can adjust the weights and biases in the direction of steepest descent.

PARAMETERS

Trainable parameters: These parameters include the biases of neurons in every layer except for the input layer, and the weights of the connections between the nodes. These parameters are considered trainable because they can be changed and updated.

Hyperparameters – Thes are fixed values, which are then fine-tuned through experimentation, to achieve the lowest possible cost value. Design-related hyperparameters include many hidden layers, the number of nodes, the types of activation function, the optimization algorithm used, and the loss function used. Some hyperparameters are adjusted after the design finalization, they are the learning rate which controls the magnitude of updates to the weights. High learning weights can lead to missing optimal weight values, and a lower learning weight could make the training process a lot longer. The regularization prevents overfitting by reducing the amount of learned information. The batch size controls the number of samples used to update the weights during training. Batch learning which uses a group of samples, which can aid in improving the performance and decrease outliers.

TRAINING

The first step in training is Initialization where the neural network will randomly assign weights and biases.

Forward propagation is when you pass input data through the network to get a desired output. You input the data and each node receives the input and multiplies it by its weights, it adds bias, and applies an activation function. The output of one layer will then go on to become the input for the next layer and so on and so forth. Then finally the output layer reaches its final prediction.

After we have the network’s prediction, we compare it to the real answer, to measure how wrong or right the neural network is using a loss function. A loss function comes up with a quantitative value between the neural networks predicted output and the true value. Some common loss functions are cross-entropy loss, this is used when dealing with probabilities. Intuition is another loss function, so if the neural networks prediction is far from the correct label, the loss is therefore high and if the prediction is closer the loss is low. These are loss functions used in classification. Another loss function used for regression, is mean squared error, penalizes large errors more than small errors. The formula for calculating the loss-function, is also known as mean squared error (MSE).

* ***i* represents the index of the sample,**
* **y-hat is the predicted outcome,**
* **y is the actual value, and**
* ***m* is the number of samples.**

**𝐶𝑜𝑠𝑡 𝐹𝑢𝑛𝑐𝑡𝑖𝑜𝑛= 𝑀𝑆𝐸=1/2𝑚 ∑129\_(𝑖=1)^𝑚▒(𝑦 ̂^((𝑖) )−𝑦^((𝑖) ) )^2**

The next step is backpropagation. Once we know the error, we can then update the weights and the biases to reduce the error. This is done with backpropagation, which will usually involve gradient descent and partial derivatives. The gradient is how much each weight on each node has contributed to the mistake. We then take the derivative of the loss function with respect to each weight and bias; this will then tell us how to change each weight to decrease the error. The neural network will then calculate how much each node contributed to the total mistake. This error is propagated from the output layer to the input layer. The weights and the biases are then updated so when we run the program we will have a lower level of error.

We will then redo these steps over and over multiple epochs, each time the neural network will keep on getting better and better at what it is trying to learn.

Validation is then used to help confirm that the neural network is adept in learning patterns that generalize well to new data. This is a test to see if the model has actually learned instead of just memorized. Validation is done by introducing a completely new dataset that was not seen in training. This will tune hyperparameters and check performance. During training you would track the network's validation loss where we should see a decrease in if the neural network is improving. You also track validation accuracy which should increase if the model is doing well. To improve the result of validation you can use some techniques like early stopping, so you would stop when the model stops improving. You would use regularization to prevent overfitting. You can hyperparameter tuning, to adjust the network parameters for the best possible performance. A final test will be performed to gauge how well the network can perform on new unseen data.

HISTORY OF NEURAL NETWORKS

The idea of a thinking machine can be traced back to the Ancient Greeks, what we see as true thinking machines weren’t really seen till the mid 1900s. In 1943 Warren S. McCulloch and Walter Pitts published together “A logical calculus of the ideas immanent in nervous activity”. This research aimed to understand the mysteries of the human brain or more specifically how the human brain could produce complex and incredible patterns through connected neurons. One of the main ideas that would come out of this work was the comparison of human neurons with binary threshold to Boolean logic (0/1 or true/false statements). In 1958, Frank Rosenblatt developed the perceptron “The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain”. He advanced Warren S. McCulloch and Walter Pitts research a step further by introducing weights to the equation. Leveraging IBM 704(), Rosenblatt got a computer to learn how to distinguish cards marked left v.s cards marked on the right. In 1974, Paul Werbos was the first person in the US to note the use of backpropagation within neural networks with his PhD thesis. In 1989, Yann LeCun published a paper illustrating how the use of constraints in backpropagation, and how its integration into neural networks can be very useful to train algorithms. This research helped recognize neural networks as hand-written zip code digits provided by the U.S. Postal Service.

CONVOLUTIONAL NEURAL NETWORK (CNN)

A convolutional neural network or CNN’s is a type of deep learning model that processes images, videos, and other spatial data. They are able to preserve spatial relationships using very special layers, instead of treating each pixel separately. CNN’s use convolution to detect patterns like, different textures, edges and shapes. There are four main layers: the convolutional layers, activation layers, pooling layers, and the fully connected layers.

The convolution layer handpicks important patterns from an image. It moves across the image, looking out for specific shapes like edges, textures and patterns. It works similarly to how you as a human will pick out certain features from an image to identify what it is. It makes processing images much more efficient as it doesn’t have to spend time looking over every single individual pixel. After scanning the entire image for certain patterns, it creates a feature map which is a new version of the image where only the specific details such as edges, shapes, and textures are highlighted. The number of filters used affects the depth of the network's output. Stride is the distance, or number of pixels. Zero padding is when the filters do not fit the image. This will then set everything outside the input to zero, which produces a larger or equally sized output. There are three types of padding, the valid padding is when there simply is no padding. Same padding ensures the input image is the same size as the output layer. Full padding increases the size of the output image.

The activation layer makes decisions in a CNN. It decides which patterns are the most relevant, and which ones are unimportant. It helps filter out any detail that is irrelevant and makes everything easier to identify the image because you can see the details much more clearly. The most common activation function in CNN’s is ReLU, it quickly filters out the weaker information and will only keep the most important information. This makes CNN’s much faster and much more efficient.

The pooling layer resizes the images making them much smaller yet they still keep the most important details. It pools all the most important information together, it is a summary tool that makes CNN’s faster and much, much more efficient.

The fully connected layer is used for decision-making, combining information, and learning complex patterns. The fully connected layer is the layer where all the nodes are connected to every node in the previous layer. It is often the very last step before a decision is made by the network. The fully connected layer, first flattens the image into a 1D vector. In the fully connected layer, each node is tasked with looking at all the features, and determining just how important each piece is to the final decision. The layer takes all the information from the previous layer and combines it with a weighted sum. This is where all the learning happens during training, the weights of connections get adjusted to make the right decision. The node then applies an activation function to its output so that it can decide what value to pass forward. This layer then combines everything together to make the final decision.

The output layer outputs class probabilities using Softmax (for multi-class classification) or Sigmoid (for binary classification)

CNNs work well for images as they can capture spatial hierarchies, parameter sharing which makes them computationally efficient, they are translation-invariant, meaning they can detect any features regardless of their position in the image.

RECURRENT NEURAL NETWORK (RNN)

A recurrent neural network or an RNN are a type of neural network that has sequential data. They are designed to process sequential data. RNNs have a memory mechanism that allows them to keep information from previous time steps, which makes them much better equipped to solve tasks involving ordered data or time-dependent data. This can be natural language processing (Machine translation and text generation) or speech recognition.

STRUCTURE OF RNNs

The input layer is where all data enters the network. The data is entered piece by piece.

The hidden layers are the memory of the neural network. In a regular neural network which treats each input separately, an RNN remembers what it has seen before and uses that information to help and influence the current step. Each time the RNN processes a new word it updates its memory and combines what it has just received as input and what it has previously learned.

After processing everything the RNN will produce an output. There are different outputs for different tasks, it could be a translated sentence, or it could maybe be the weather forecast.

DIFFERENT VARIATIONS OF RNNs

One-to-One: Each one input has one output which is identical to a normal neural network.

One-to-Many: One input generates a sequence of outputs.

Many-to-One: Many inputs that lead to one output.

Many-to-Many: Many inputs that lead to many outputs.

PROBLEMS

They struggle to remember important information from further back in the sequence. When gradients become too small during the process of backpropagation, earlier layers fail to learn properly. Gradients can also become too large, leading to unstable training.

LONG SHORT-TERM MEMORY NETWORKS (LSTMs)

Long short-term memory networks or LSTMs are an advanced type of recurrent neural network (RNN), designed to fix the long-term dependencies that appear in RNNs.

We need LSTMs because they solve the problems that come with RNNs, like the vanishing gradient problems. LSTMs solve this by having more advanced memory mechanisms. Unlike regular RNNs which have a single hidden state, LSTMs introduce a cell state that enables long-term memory, along with three gates that control information flow.

STRUCTURE OF LSTMs

The cell state is the memory of the LSTM, this state carries important information, and allows the network to retain or forget past data as needed. LSTMs also have the usual hidden states that come with the standard RNNs, that are used for short-term memory.

The first gate is the forget gate, which decides what past information should be “forgotten" by the cell state. It looks at the previous hidden state and the current input and outputs a value between zero (completely) and 1 (fully retained).

The input gate decides what new information should be stored in the cell state. It has a filter that will decide which values will be updated. I had a candidate update which creates new memory values to be stored. The forget gate and the input gate work hand in hand to update cell state, useless information is deleted and important information is added.

The output gate determines what the next hidden state should be, the hidden state is the passed to the next step and is used in the process of making predictions

TYPES OF LSTMs

There are certain variations of LSTMs that may aid in improving efficiency. The bidirectional LSTM – Processes data in both forward and backward directions to be able to look at things from two perspectives; the past and the future. The peephole LSTM – Allows gates to directly look at the cell state for better and more informed decisions. The gated recurrent unit is a simplified version of LSTM with fewer gates, making it faster while still effective for long-term dependencies.

HUNGER IN FIRST WORLD COUNTRIES
 In 2023 the U.S. 47 million people in the United States experienced food insecurity, 14 million of the people are children. In 2023 in Canada 22.9% of our population in the ten provinces lived in a food-insecure household. That amounts to 9.16 million Canadians which includes 2.1 million children. Yet in 2023 46.5% of all food in Canada is wasted. 41% of this is avoidable. 23% of avoidable food waste is from best before dates. In the U.S about 30%-40% of food is wasted every year. Food waste doesn’t just affect people but also contributes to 11% of the world’s emissions of gases like methane, carbon dioxide and chlorofluorocarbons.

NEURAL NETWORKS IN AGRICULTURE

Canada is said to have lost the equivalent of seven small farms a day for 20 years. Each year we have less and less farmland, due to factors such as urbanization and soil quality. This forces us to convert forest land into farmland, which has so many environmental repercussions. Which means that we need farms to be more efficient and more effective, to not just help our farmers but to also help feed our people. If there is a surplus then prices will be lower. Using neural networks we can help farmers save billions of dollars each year, by using image recognition with convolutional neural networks, check for pests, soil composition, fertilizer management, livestock health and wellbeing. AI can also help make data based decisions, by factoring and monitoring the weather conditions, the application of fertilizer and pesticides. AI can also detect leaks and irrigation systems. Using convolutional neural networks you could detect mold, rot, insects, or other threats to crop health.

















AI IN GROCERY STORES

In Canada, we rely heavily on best before dates, yet these dates aren’t always a correct indicator of food safety. Best before dates simply indicate when they are at their best quality. After their best before dates they may lose their flavour or may become stale. Yet in Canada we often associate the best before date with the safety of the food, this leads to food waste. So if in grocery stores we were able to instead sell foods that have passed their expiry date at a discount price we can help 9.16 million Canadians with their food insecurity. Where does AI come into all of this? Well having to manage all of this can be hard for many different grocery stores and make them less willing to implement this system. So if we were able to use AI to manage all of the different products, and there best before date. This will not only help many people. It will also help decrease food waste and in turn help decrease our emissions. Which will not just benefit our country but the world in general. We don’t have to just implement this in Canada we can also implement this around the world which will be helping millions even billions of people.