UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES
INGENIERÍA ELECTRÓNICA

Universidad Tecnológica Nacional

Curso de grado

Sistema de navegación inteligente acelerado por hardware

Director: Ing. Silvio Abel Tapino

Tutor: Ing. Alejandro Furfaro

Co-tutor: Ing. Luciano Ferreyro

Tutor por la cátedra: Dr. Ing. Matias Hampel

Tutor por la cátedra: Ing. Maria Alejandra Gutierrez

Autor: Martín Fuschetto (mfuschetto@frba.utn.edu.ar)

Año académico 2021/2022

Contents

1 AGRADECIMIENTOS

... (TERMINAR)

2 INTRODUCCIÓN

Tradicionalmente la robótica estaba centrada en sectores industriales manufactureros orientados a la producción masiva. A mediados de los 60's se introducen robots manipuladores en distintos tipos de industrias. Típicamente los robots desarrollaban tareas repetitivas, el cual exigía tomar algunas piezas y reubicarlas en otra área a la cual el robot manipulador sea capaz de llegar con la máxima extensión de su articulación lo cual resultaba en un problema. Una solución a este problema fué desarrollar un vehículo móvil sobre rieles y así es como a mediado de los 80's aparecieron los primeros vehículos guiados automáticamente (AGV's).

Fuera del entorno industrial, en donde se imposibilitaba estructurar el entorno, se les doto a los robots un mayor grado de inteligencia y capacidad para poder desenvolverse.

Uno de los desafíos mas grandes en la aplicación de robots es la navegación en entornos desconocidos abarrotados de obstáculos. La navegación se vuelve aun mas compleja cuando se desconoce la ubicación de estos a priori. Se introdujo así entonces el concepto de conjunto difuso (Fuzzy Set) bajo el que reside la idea de que los elementos sobre los que se construye el pensamiento humano no son números sino etiquetas lingüísticas.

La lógica difusa se ha utilizado en el diseño de múltiples posibles soluciones en donde se han creado distintos sistemas de control de navegación orientados a robots para que estos puedan llegar a destino evitando obstáculos en su camino.

A lo largo de los años se han desarrollado algunos enfoques distintos, uno muy particular en el cual la navegación se divide en dos partes. La primera compuesta en comportamientos básicos tales como: lograr metas, evitación de obstáculos y seguimiento de muros. La segunda, una capa de supervisión responsable de la selección de las acciones (elección de comportamientos según el contexto).

El principal aporte de mi proyecto de investigación es realizar un sistema de control neuro-difuso nuevo que solo va a necesitar un controlador difuso pero que además, va a estar dividido en dos partes:

La primera (difuso) con comportamientos básicos: lograr metas, evitación de obstáculos, etc. A partir de ahora llamado **"Modulo de sistema de control difuso"**.

La segunda de supervisión (neuronal) en donde se va a seleccionar, arbitrar o fusionar comportamientos de la lista de estos en función de la salida del sistema neural convolucional el cual tendrá la capacidad de, a través de una cámara, poder reconocer el obstáculo a evadir y en función de la ubicación y dimensión de este decidir un comportamiento difuso. A partir de ahora llamado **"Modulo de procesamiento de imágenes"**.

Siempre buscando con esta idea que la toma de decisión se parezca un poco mas a la del ser humano, elegir un rumbo en función de lo que el robot ve y reconoce, ya que desde mi punto de vista es antinatural la elección de trayectoria solo en función de la ubicación del obstáculo y el target (destino), hay que tener en cuenta que nosotros, al encontrar un obstáculo que nos evita el paso también evaluamos el trayecto a seguir en función de las dimensiones del obstáculo y del largo del trayecto a recorrer. Por ejemplo al eludir un obstáculo muchas veces evaluamos si eludir por la izquierda o por la derecha es mas conveniente en función del largo de ambos caminos.

La detección de objetos es una tarea de suma importancia para la conducción autónoma, junto con esta tarea viene aparejada la responsabilidades de garantizar la precisión a la hora de detectar estos objetos y a su vez, de suma importancia, inferirlo en tiempo real para garantizar el adecuado control del móvil. Para satisfacer todas estas necesidades se propuso implementar una SqueezeDet, una red neuronal totalmente convolucional para la detección de objetos caracterizada por su precisión, tamaño, velocidad y consumo de energía.

La investigación fue llevada a cabo en el marco de la materia Proyecto Final de la carrera de Ingeniería Electrónica de la Universidad Tecnológica Nacional (Regional Buenos Aires), y fue auspiciada por el Laboratorio de Procesamiento Digital (DPLAB) del Departamento de Electrónica de dicha facultad regional. La investigación realizada y el diseño final quedaron a disposición del DPLAB como base para futuros trabajos de investigación.

Se inicia este trabajo en un capitulo en donde se presenta el marco teorico del proyecto. ... (TERMINAR)

3 MARCO TEÓRICO

3.1 Redes Neuronales Convolucionales

3.1.1 Introducción

Sin duda alguna en la ultima década las redes neuronales convolucionales se popularizaron en gran medida. Todo comenzó con una nueva forma de computación inspirada en modelos biológicos, los cuales consisten en sistemas con un gran numero de procesos interconectados funcionando en paralelo los cuales procesan información y tienen la capacidad de aprender a través de la experiencia.

La detección de objetos en imágenes es una tarea de suma importancia a la hora de realizar un sistema de navegación autónomo, las redes neuronales vienen a resolver el problema.

Dentro de las las redes neuronales hay distintos tipos y una de ellas son las redes neuronales convolucionales (CNN) las cuales están inspiradas en las neuronas de la corteza visual primaria del cerebro. Estos tipos de redes neuronales pueden dotar a los robots de visión y es por eso que vienen a resolver el problema.

De esta forma esta sección de la tesis se enfocara en aportar un marco teórico de las redes neuronales convolucionales.

3.1.2 El perceptrón

En 1957 Frank Rosenblatt comenzó el desarrollo de una red neuronal a la que denomino "El perceptrón" fue tal su aporte que hoy en día se lo utiliza para reconocer patrones. Este modelo fue capaz de generalizar, es decir que luego un aprendizaje de ciertos patrones, y a pesar de sus limitaciones (era incapaz de resolver la OR exclusiva y en general incapaz de clasificar clases no separables linealmente) era capaz de reconocer patrones que jamas se le hayan presentado. La entrada de la función de activación se calcula como la suma ponderada de todas las entradas del perceptrón más un valor de bias.

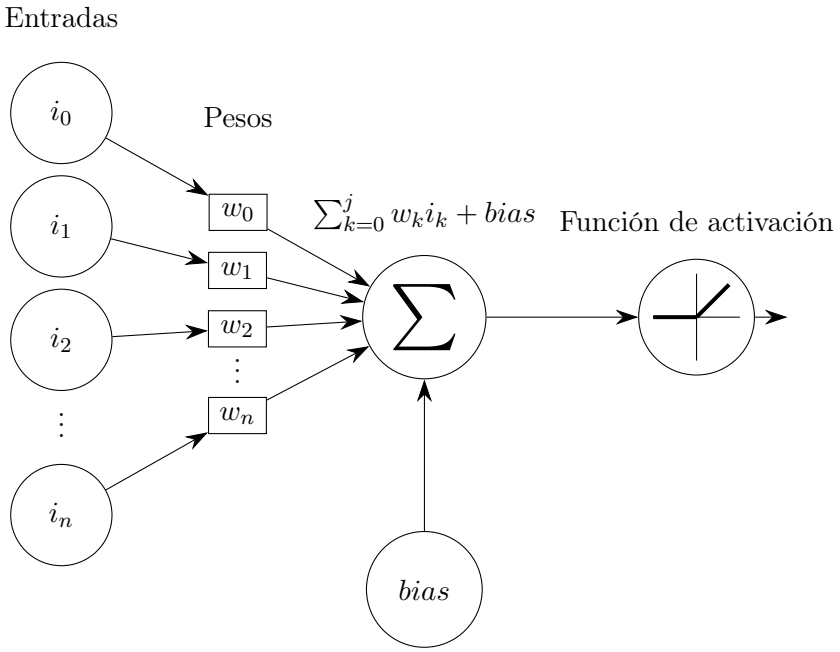


Figure 1: El perceptrón

3.1.3 ¿Que es una red neuronal?

Una red neuronal es un grupo interconecto de distintos elementos procesadores simples que operan en paralelo, cuya función se determina a partir de distintas características de la red, su

estructura, la fuerza o pesos en las conexiones y el procesamiento realizado en cada nodo.

Muchas veces se la define como un aproximador universal. El modelo conocido como perceptrón multicapa esta compuesto por "k" capas de neuronas que están interconectadas y cada salida de cada neurona es la suma ponderada de todas las salidas de las neuronas de la capa anterior mas un valor de bías (Figure 1).

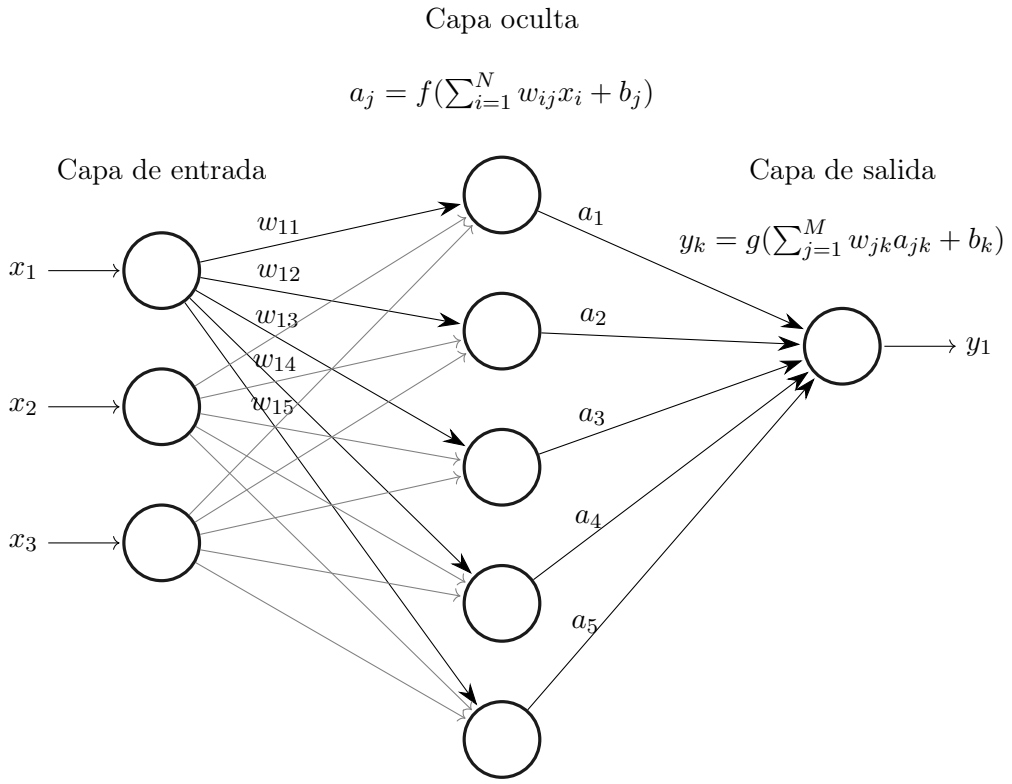


Figure 2: Perceptrón multicapa

Debido a que la salida de una determinada neurona de la capa "k" se calcula utilizando todas las salidas de la neurona anterior se denomina capa de neuronas "fully conected".

- Vector x: Entrada de la capa (conformado por las salidas de cada una de las neuronas de la capa anterior).
- n: Cantidad de neuronas de la capa anterior.
- Vector y: Vector de salidas.
- m: Cantidad de neuronas de la capa a calcular.
- Matriz w: Coeficientes de la capa "fully connected".
- Vector b: Bias de la capa "fully connected".

$$\begin{bmatrix} w_{11} & \dots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{m1} & \dots & w_{mn} \end{bmatrix} * \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} a_1 \\ \vdots \\ a_m \end{bmatrix}$$

3.1.4 Capas convolucionales

La principal diferencia de una red neuronal convolucional de cualquier otro red neuronal es que utiliza una operación llama "convolución" en algunas de sus capas en vez de utilizar la multiplicación de matrices. Esta operación recibe como entrada una imagen y luego le aplica un

filtro (también llamado "kernel") y retorna otra imagen con las características de la imagen de entrada.

A continuación el proceso:

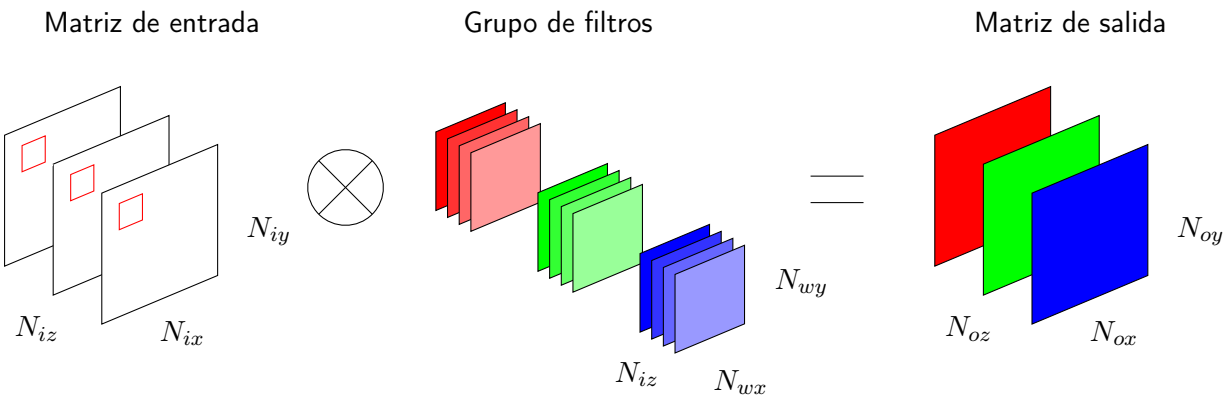


Figure 3: Operación de convolución

- N_{iz} : cantidad de canales de la matriz de entrada a la capa.
- N_{ix} : ancho de la matriz de entrada.
- N_{iy} : largo de la matriz de entrada.
- N_{wx} : ancho de cada filtro (del grupo de filtros) de la capa.
- N_{wy} : largo de cada filtro (del grupo de filtros) de la capa.
- N_{oz} : cantidad de filtros de la capa o bien los canales de la matriz de salida.
- N_{ox} : ancho de la matriz de salida.
- N_{oy} : largo de la matriz de salida.

Las salidas de una determinada capa convolucional se obtiene al desplazar N_{of} filtros de dimensiones $N_{wx}xN_{wy}xN_{iz}$ por las salidas de la capa anterior. En cada desplazamiento se realiza una suma ponderada de las salidas de la capa anterior por los pesos del filtro.

A continuación se presenta su algoritmo en pseudocódigo.

Algorithm 1 Algoritmo de convolución

```
for (c = 0; c < N_oz; ++c) do                          ▷ Selector de filtro
    for (i = 0; i < N_ix; ++i) do                      ▷ Eje X de la matriz de entrada
        for (j = 0; j < N_iy; ++j) do                  ▷ Eje Y de la matriz de entrada
            for (v = 0; v < N_iz; ++v) do              ▷ Canales de la matriz de entrada
                for (k = 0; k < N_wx; ++k) do          ▷ Eje X del filtro
                    for (l = 0; l < N_wy; ++l) do      ▷ Eje Y del filtro
                        o[j, i, c] += i[l, k, v, j, i] * w[l, k, v, c]
                    end for
                end for
            end for
            o[j, i, c] += b[c]
        end for
    end for
end for
```

3.1.4.1 Funciones de activación

Normalmente utilizadas luego de una capa convolucional (o "full connected", según corresponda) y su función es la de agregar alinealidades al comportamiento de la red. Aplican un valor a cada valor de entrada (descripto mediante una función matemática). A continuación una explicación de las mas utilizadas.

Sigmoide

Una capa de activación Sigmoide transforma todos los valores de la entrada a una escala del (0,1) en donde los valores muy altos tienden a 1 y los valores muy bajos a 0.

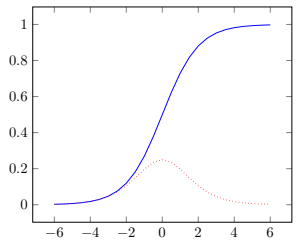


Figure 4: Función de activación: Sigmoide

ReLu

Una capa de activación ReLu reemplaza todos los valores negativos recibidos a su entrada por ceros, su propósito es hacer el modulo alineal.

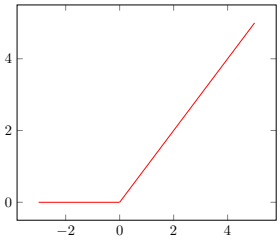


Figure 5: Función de activación: ReLu

3.1.4.2 Método de submuestreo: capa de Pooling

La capa de Pooling tiene como objetivo submuestrear la matriz de entrada reduciendo su tamaño. Una de sus principales ventajas es la de reducir el costo computacional de la red neuronal convolucional completa ya que reduce la cantidad de parámetros que esta tiene que aprender. A continuación una explicación de las mas utilizadas a la hora de construir una red neuronal convolucional.

Max Pooling

Max Pooling consiste en tomar una "ventana" de la matriz de entrada y efectuar una operación fija que retorne un escalar. En este caso retorna el valor máximo de la ventana y lo asigna como salida. A modo de ejemplo:

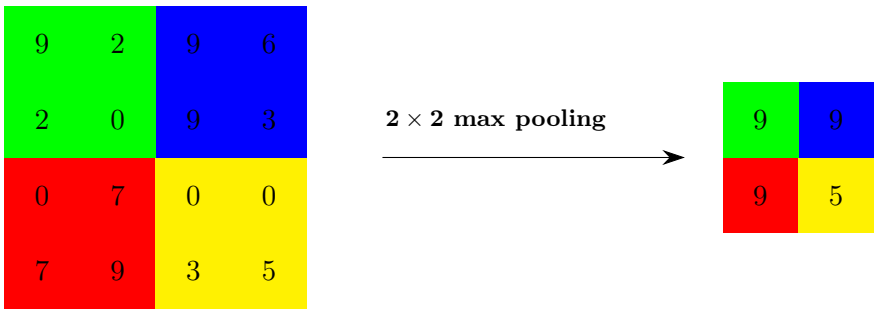


Figure 6: Max Pooling

Average Pooling

Average Pooling consiste en tomar una "ventana" de la matriz de entrada y efectuar una operación fija que retorne un escalar. En este caso retorna el valor medio de la ventana y lo asigna como salida. A modo de ejemplo:

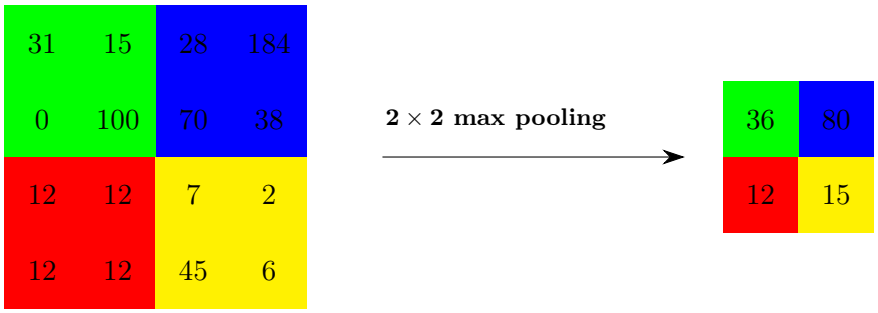


Figure 7: Average Pooling

3.1.4.3 Capa de concatenación

Esta capa es la encargada de tomar las entradas y concatenarlas a lo largo de una dimensión especifica, la única condición es que las entradas deben tener el mismo tamaño en todas las dimensiones excepto en la dimensión de concatenación.

3.1.4.4 Otras capas que suelen aparecer en una red neuronal convolucional

Existen otras capas que suelen aparecer en algunas redes neuronales convolucionales como por ejemplo la capa de Dropout la cual consiste en desconectar un porcentaje de las neuronas en cada iteración del entrenamiento lo cual mejor la generalización de la red y ayuda a converger el entrenamiento.

La Normalización por lotes (mas conocida en ingles por "Batch normalization") que consiste en añadir un capa entre las neuronas y la función de activación para normalizar las activaciones de salida.

Existen algunas capas como la capa de convolución transpuesta (también conocida como capa de deconvolucion o capa de convolución fraccional) la cual se la utiliza para recuperar las dimensiones reducidas. Por ejemplo, luego de realizar una convolución con un step mayor que uno lo cual reduce el tamaño de la matriz de salida.

3.1.5 Arquitectura de una red neuronal convolucional

Una arquitectura de una red neuronal convolucional normalmente esta formada por una pila de capas distintas que transforman la matriz de entrada en una matriz de salida a través de una función diferenciable.

Algunas características importantes en las arquitecturas son: Los formatos de entrada/salida, la cantidad de capas de la red, parámetros de cada capa, la secuencia y forma de conexión de las capas entre si, y el algoritmo de entrenamiento.
Algunas redes y sus arquitecturas:

- LeNet-5: 2 capas convolucionales y 3 "fully connected". Al rededor de 60.000 parámetros. Año 1998.

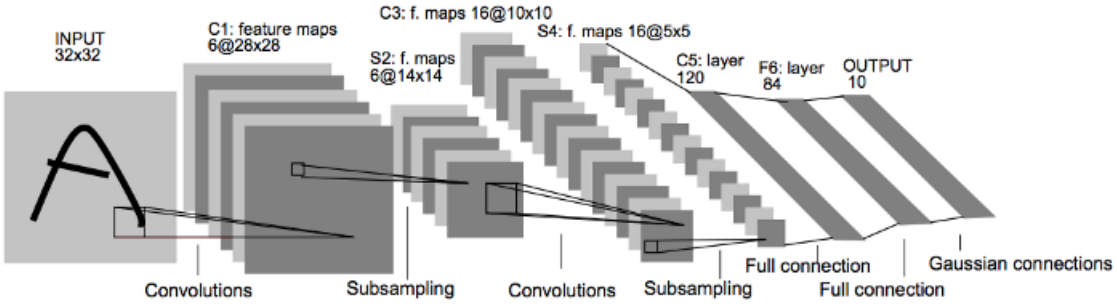


Figure 8: LeNet-5

- AlexNet: 8 capas, 3 "fully connected" y 5 convolucionales. Al rededor de 60 millones de parametros. Año 2012.
- VGG-16: 13 capas convolucionales y 3 "fully connected". Al rededor de 138 millones de parametros. Año 2014.
- ResNet-50: 50 capas, con modulos llamados "ResNet" (cada uno de 2 o 3 capas convolucionales). Al rededor de 26 millones de parametros. Año 2015.

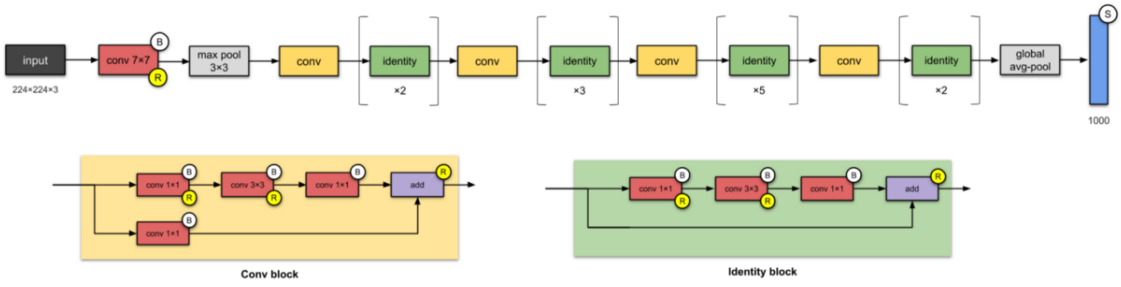
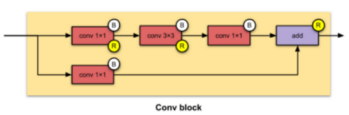


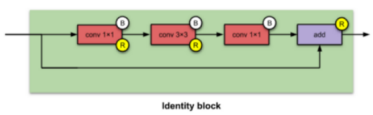
Figure 9: ResNet-50, de Raimi Karim, towardsdatascience

3.1.6 Micro-arquitectura de una red neuronal convolucional

La micro-arquitectura hace referencia a las capas individuales y/o módulos. A modo de ejemplo observemos el modulo "ResNet" de la red neuronal convolucionar ResNet-50.



(a) "Conv block" (Bloque de convolución)



(b) Identity block (Bloque de identidad)

Figure 10: ResNet-50 modulo

3.1.7 Entrenamiento de la red

3.1.7.1 Introducción

La fase de entrenamiento de la red neuronal convolucional consiste en, a partir de un set de imágenes de entrenamiento y sus respectivas etiquetas de los objetos a reconocer realizar iteraciones (también llamadas épocas) para ir determinando los pesos de cada uno de las neuronas en la red. Esta determinación se realiza a través del método de calculo conocido como "Backpropagation".

Entonces en cada época, la etapa de entrenamiento se divide en dos etapas: Una fase directa (o hacia adelante) en donde una matriz de entrada pasa a través de toda la red neuronal. Se comparan las salidas obtenidas con las deseadas y se calcula el error para una de las salidas.
Y una fase inversa (o hacia atrás) donde los gradientes se pasan hacia atrás y los pesos se actualizan. Cada capa recibe un gradiente de perdida respecto a la salida y devuelve un gradiente de perdida respecto a la entrada.
Algo a tener en cuenta es que la fase de retroceso necesita datos que son almacenados durante la fase hacia adelante ya que se guardan datos de cada capas (entradas y valores intermedios) por lo que antes de realizar la propagación hacia atrás "backpropagation" es necesario una propagación hacia adelante "forward propagation".

3.1.7.2 Función de perdida

Como mencione anteriormente finalizando la fase directa o de propagación hacia adelante se comparan las salidas obtenidas con las deseadas y se calcula el error para cada una de ellas. Esto se realiza a través de la Función de perdida $J_{(W)}$.
Es una tarea importante, en este proceso, la elección de la función de perdida que se va a minimizar. Ésta relaciona el valor esperado con los valores obtenido en la propagación hacia adelante.

Entropía cruzada

La entropía cruzada es una función de perdida típicamente utilizada para la clasificación de imágenes.

$$\boxed{L = -ln(p_c)} \tag{1}$$

En donde "c" es la clase correcta y p_c es la probabilidad predicha para la clase "c".

3.1.7.3 Desarrollo matemático

Entonces, en la fase de la propagación hacia adelante, matemáticamente hablando se parte de una red en donde $a^{(l)} = f(z^{(l)})$ es la salida de la capa "l", tal que $z^{(l)}$ es el vector de pre-activación y se puede generalizar como $z^{(l)} = W^{(l)} * a^{(l-1)}$ siendo W los coeficientes de la capa y "a" la salida de la capa anterior y por ultimo $f()$, la función de activación de la capa l.

Por otro lado, en la fase de propagación hacia atrás, matemáticamente hablando se utiliza un método de calculo de gradiente conocido como retropropagación y para poder utilizar este método es necesario encontrar las derivadas parciales de la función de perdida respecto a cada uno de los parámetros de la red $\frac{\partial J(W)}{\partial W_{ij}^{l}}$.

- Se parte de Calcular el error y el delta de salida:
    1. $e^{(L)} = d - a^{(L)}$ tal que "L" es la ultima capa, "d" el valor que debería dar a la salida y "a" el valor de la salida obtenido previamente en la propagación hacia adelante.

2. $\delta^{(L)} = f'(z^{(L)}) \cdot e^{(L)}$ siendo $f'(z^{(L)}$ la derivada de la función de activación de la capa de salida.

- Para cada capa de la red neuronal se procede a calcular sus errores y deltas.

1. $e^{(L)} = W^{(l+1)^T} - \delta^{(l+1)}$ tal que "L" es la ultima capa, "d" el valor que debería dar a la salida y "a" el valor de la salida obtenido previamente en la propagación hacia adelante.
2. $\delta^{(L)} = f'(z^{(L)}) \cdot e^{(L)}$ siendo $f'(z^{(L)}$ la derivada de la función de activación de la capa de salida.

- Actualizamos las matrices de pesos.

1. $\Delta W^{(l)} = \lambda * (\delta^{(l)} \times a^{(l-1)})$ tal que λ se define como el factor de aprendizaje, y $a^{(l-1)}$ la salida de la neurona anterior.
2. Actualización: $W^{(l)} \leftarrow W^{(l)} + \Lambda W^{(l)}$

Este proceso se repite periódicamente (épocas) hasta alcanzar el entrenamiento deseado de la red.

3.2 Lógica Difusa

3.2.1 Introducción

A lo largo de los años hubo una gran cantidad de cambios en los paradigmas en las ciencias y matemáticas, en este caso me enfoco en el concepto de incertidumbre. Según la visión tradicional, la incertidumbre no era parte de la ciencia y esta se esforzaba en eliminarla. Según la visión alternativa, la incertidumbre juega un rol muy importante en la ciencia.

La lógica difusa fue investigada por primera vez por el ingeniero Lotfy A. Zadeh, en la Universidad de Berkeley (California) cuando logro comprender lo que el llamo principio de incompatibilidad: "A medida que aumenta la complejidad, las declaraciones precisas pierden significado y las declaraciones significativas pierden precisión". Lo que quiere decir es que cuando un sistema se vuelve mas complejo, nuestra capacidad de ser precisos disminuye (inherentemente debido a que la capacidad de la computación de datos no es infinita) mucho mas allá del cual la precisión y el significado son características indispensables.

La lógica difusa permite tomar decisiones mas o menos intensas en función de grados intermedios de cumplimiento de una premisa. Esta nueva lógica permite comprender nuestras expresiones del tipo «hace mucho frió», «él es un poco alto», «su ritmo es algo lento», entre otras.

Este es uno de los temas que siempre aparece en cualquier documento, libro, artículos y/o revistas dedicada a los sistemas de control y se caracteriza por ser un sistema de control sencillo, robusto, económico, de muy rápida implementación y con la ventaja de que permite agregar múltiples entradas sin complejizar demasiado su resolución.

3.2.2 Conjuntos difusos y funciones características

Lotfy A. Zadeh utilizo el ejemplo de los hombres altos para poder explicar el concepto de conjunto difuso. Según la teoría clásica los hombres que superen cierta altura pertenecen al conjunto de hombres "altos". Así por ejemplo, tendríamos que cualquier hombre que supere 1,7 mts es considerado alto, en cambio quien pida 1,69 mts deja de considerado alto lo cual no tiene mucho sentido que un hombre sea considerado mientras que otro no cuando su altura difiere 1 cm. La lógica difusa propone un conjunto que no tiene una frontera clara, y asigna un cierto grado de pertenencia a ese conjunto, entre 0 y 1. Ejemplo: Un hombre que mida 1,49 mts tiene un grado de pertenencia del 0.8 al conjunto de hombres bajos, mientras que un hombre que mida 1.6 mts tiene un grado de pertenencia del 0.5 a este mismo conjunto.

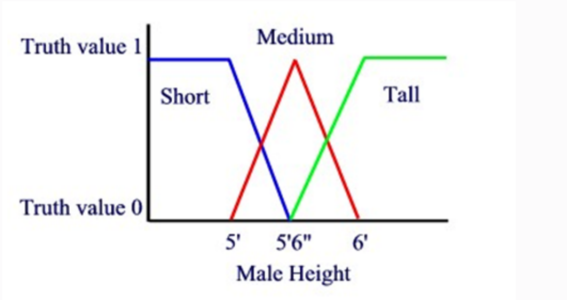


Figure 11: Conjuntos difusos y funciones características

Este grado de pertenencia de la persona en el conjunto se define mediante la función característica asociada al conjunto difuso. La función característica utilizada, depende del criterio que utilicemos para resolver nuestro problema. La única condición que debe cumplir una función característica es que tome valores entre 0 y 1 y sea continua.

Funciones características mas utilizadas: Gaussiana, Sigmoidal, Gamma, Pi, Campana, Trapezoidal, Triangular.

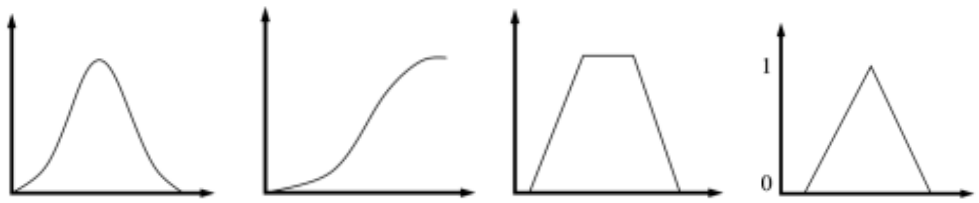


Figure 12: (a) Gaussiana (b) Sigmoidal (c) Rectangular (d) Triangular

3.2.2.1 Operaciones con conjuntos difusos

- Intersección:

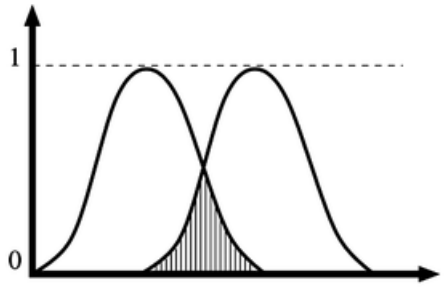


Figure 13: Intersección

$$\mu(A \cap B)(x) = min(\mu_A(x), \mu_B(x)) \tag{2}$$

- Unión:

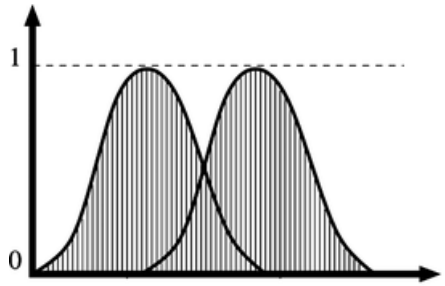


Figure 14: Unión

$$\mu(A \cup B)(x) = max(\mu_A(x), \mu_B(x)) \tag{3}$$

- Complemento:

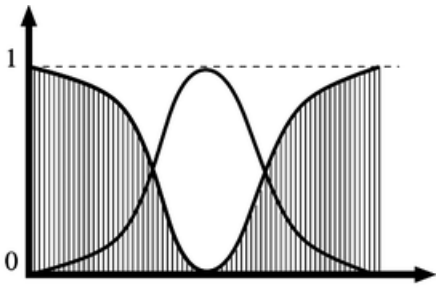


Figure 15: Complemento

$$\boxed{\mu_A(x) = 1 - \mu_A(x)} \tag{4}$$

- Inclusión: Un conjunto difuso, A, esta incluido en otro, B, si su función de pertenencia toma valores mas pequeños:

$$\boxed{\mu_B(x) \leq \mu_A(x)} \tag{5}$$

3.2.3 Reglas

Se llama reglas difusas al conjunto de proposiciones IF-THEN que modelan el problema que uno planea resolver. Típicamente tienen la forma:

"Si v es A entonces c es B"

Donde A y B son conjuntos difusos en los rangos de "v" y "c" respectivamente.

El conjunto de reglas se definen en lenguaje natural, según el grado de pertenencia de sus variables será el grado de verdad de la regla. El comportamiento de la salida dependerá de las reglas con mayor grado de verdad.

3.2.4 Control Difuso

Figure 16: Proceso de control difuso

Se puede descomponer el proceso de control difuso en tres etapas principales.

3.2.4.1 Fuzzificación Primera parte del proceso, dados los valores de entrada se calcula el grado de pertenencia a cada uno de los conjuntos difusos considerados, mediante las funciones características asociadas a estos conjuntos difusos.
Por ejemplo, tomando la variable velocidad.

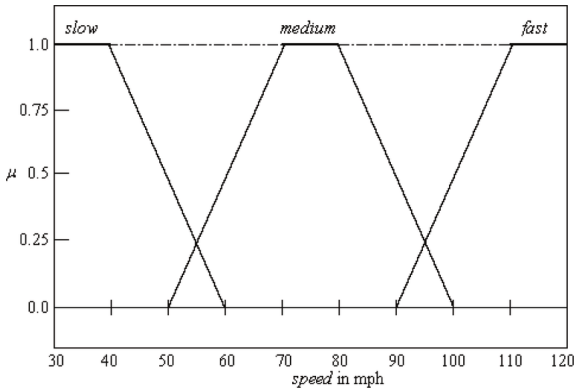


Figure 17: Ejemplo Fuzzificación

Con un valor de 65 mph, se cuantifica su grado de pertenencia a los conjuntos representados por algunas etiquetas lingüísticas: baja, media, alta. Para ellos se debió haber definido previamente una función de pertenencia para cada una de esas etiquetas que define que valores de la variable velocidad le pertenece y con que grado de pertenencia a cada una de ellas. Siguiendo con nuestro ejemplo, tiene un grado de pertenencia del 0 en la etiqueta ”baja”. Un grado de pertenencia del 0.75 en la etiqueta ”media” y un 0 en la etiqueta ”alta”.

3.2.4.2 Evaluación de reglas Las reglas relacionan los conjuntos difusos de entrada mediante mecanismos de inferencia (reglas) con las salidas difusas.

Una vez terminada la etapa previa, la fuzzificación, se procede a evaluar los antecedentes de las reglas, obteniendo el grado de verdad o "peso" para cada una de ellas.

El peso de la regla estará dado por la veracidad del antecedente. Por ejemplo, si tiene una regla del tipo:

SI la velocidad es "baja" **ENTONCES** "aumente" la potencia del motor.

Se asigna como peso el grado de pertenencia del valor leído de velocidad a la etiqueta lingüística "baja".

En el caso de que tenga una regla con conectivos lógicos **Y**:

SI la velocidad es "baja" **Y** los obstáculos están "lejos" **ENTONCES** "aumente" la potencia del motor.

La regla sera tan verdadera como lo sea el menos verdadero de sus antecedentes. Por ende, se le asigna a la regla como peso, el menor de los grados de pertenencia de las variables de los antecedentes a las respectivas etiquetas lingüísticas.

Esto se repite para el resto de las reglas, entonces cada regla queda con su peso correspondiente.

Así como cada entrada tiene sus propias funciones de pertenencia, cada salida le corresponde su propio set de funciones de pertenencia. Cada una de ella es una salida difusa ("baje", "no modifique", "aumente" la potencia del motor).

A cada una de las salidas difusas se les asigna un valor (grado de aplicabilidad) es el valor máximo de cada uno de los pesos de las reglas que la mencionan. De esta manera, tenemos casa salida difusa con su valor.

3.2.4.3 Defuzzificación De la etapa anterior del proceso tenemos varias salidas difusas ("baje", "no modifique", "aumente") cada una de ellas con un valor de verdad o de aplicabilidad (un numero de 0 a 1) para cada variable de salida del sistema (en este caso la potencia del motor).

Ahora la pregunta es, cual es la potencia del motor? Una forma fácil y efectiva de determinarlo es realizando la operación denominada C.O.G (Centro de gravedad en ingles) el cual consiste en:

$$u = \frac{\int u.\mu(u) \cdot du}{\int \mu(u) \cdot du} \tag{6}$$

Ejemplo del método "Centro de gravedad"

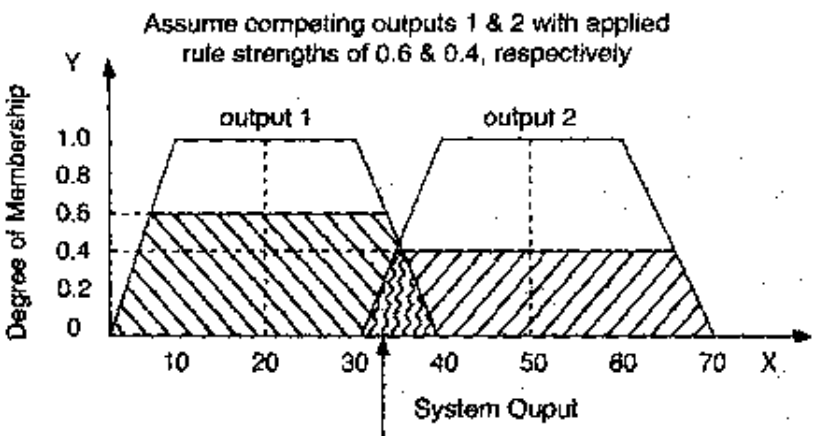


Figure 18: Ejemplo C.O.G

Método del Centro de gravedad (C.O.G) (asumiendo que los pesos de las reglas 1 y 2 son 0.6 y 0.4 respectivamente)

- Se determina el centroide de la función de pertenencia de salida. En este caso seria X=20 para la salida 1 y X=50 para la salida 2.
- Las funciones de pertenencia están limitadas en altura por el peso de la regla aplicada, y se calculan las áreas de las funciones de pertenencias (Área del trapecio $A = \frac{B+b}{2}.h$).

  Área de la salida 1: $A_1 = \frac{40+28}{2}.0.6 = 20.4$

  Área de la salida 2: $A_2 = \frac{40+32}{2}.0.4 = 14.4$
- Finalmente la salida defuzzificada es el promedio ponderado de los centroides y las áreas calculadas: Peso promedio: $W_{av} = \frac{20*20.4+14.4*50}{20.4+14.4} = 32.4$

Utilizando Singletons (asumiendo que los pesos de las reglas 1 y 2 son 0.6 y 0.4 respectivamente)

Con el propósito de simplificar el proceso de defuzzificación se utiliza un Singleton (una función de pertenencia de salida representada por una única linea vertical). De esta forma el calculo del centro de gravedad se traduce en un único calculo promedio ponderado entre los centroides y la aplicabilidad de las reglas: $W_{av} = \frac{20*0.6+50*0.4}{0.6+0.4} = 32.0$

3.3 Sistema Integrado

4 MODULO: PROCESAMIENTO DE IMÁGENES

5 MODULO: SISTEMA DE CONTROL DIFUSO

6 SISTEMA INTEGRADO

7 RESULTADOS

7.1 Protocolo de prueba

7.1.1 Evaluación del modulo: PROCESAMIENTO DE IMÁGENES

Existe distintas formas de evaluar un modelo de detección de objetos. Algunas de ellas son:

7.1.1.1 Matriz de confusión

Cada fila de la matriz representa el numero de predicciones de cada clase, mientras que cada columna representa el verdadero valor. Se evalúan todos los resultados y se dividen en verdaderos positivos, falsos negativos, falsos positivos y verdadero negativo.

		Valor verdadero	
		Tomate	Next
Valor predicho	Tomate	Verdadero Positivo (VP)	Falso Positivo (FP)
	Next	Falso Negativo (FN)	Verdadero Negativo (VN)

- Verdaderos positivos (VP): Predicciones correctas respecto a los datos verdaderos.
- Falso positivo (FP): Se detecta una clase que no existe en los datos verdaderos.
- Falso Negativo (FN): Una clase se detecta como otra clase o simplemente no se detecta pero si existe en los datos.
- Verdadero Negativo (VN): Una clase que no se detecta o se detecta como otra y no existe en los datos.

Métrica

Precisión La precisión mide el porcentaje de predicciones positivas correctas entre todos los casos positivos que existen.

$$Precisin = \frac{VP}{VP + FP}$$

Sensibilidad La sensibilidad (Recall) mide las predicciones positivas correctas entre todas las predicciones hechas.

$$Sensibilidad = \frac{VP}{VP + FN}$$

Exactitud Pocentaje total de los aciertos del modelo.

$$Exactitud = \frac{TP + TN}{TP + TN + FP + FN}$$

7.1.1.2 Intersección sobre unión

Mide la precisión de un detector en un conjunto de datos en particular.

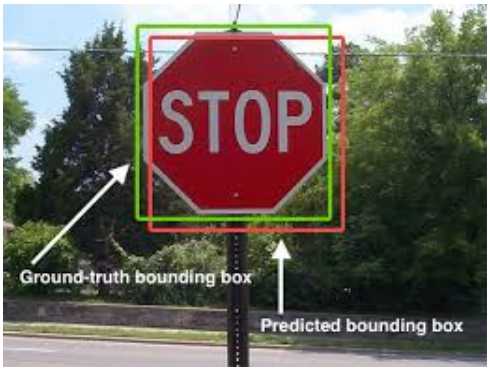


Figure 19: Ejemplo IoU

$$IoU = \frac{\text{Área de solapamiento}}{\text{Área de unión}}$$

Donde, "Área de solapamiento" entre la predicción y la realidad, mientras que el "Área de unión" es el área sumada entre la predicción y la realidad.

7.1.1.3 Precisión promediada (AP: Average Precision)

Es una métrica comúnmente utilizada como resumen de una curva que forma la relación entre dos métricas, la Precisión y la Sensibilidad de la red.

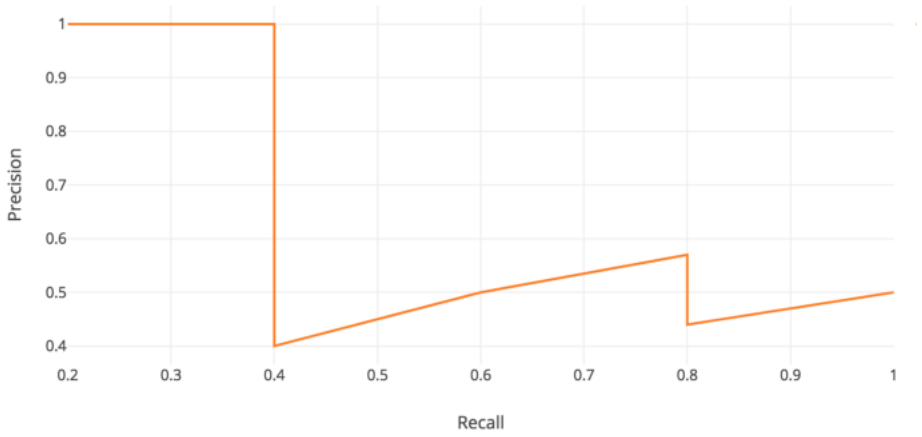


Figure 20: Precisión-Sensibilidad

Algo a tener en cuenta es que la curva es decreciente ya que si uno entrena a su clasificador para aumentar la precisión, la sensibilidad disminuirá.

Como por definición la precisión promediada (AP) es el área bajo la curva y queremos hacer mas sencillo el proceso, antes de calcular la precisión promediada, ponderamos la curva.

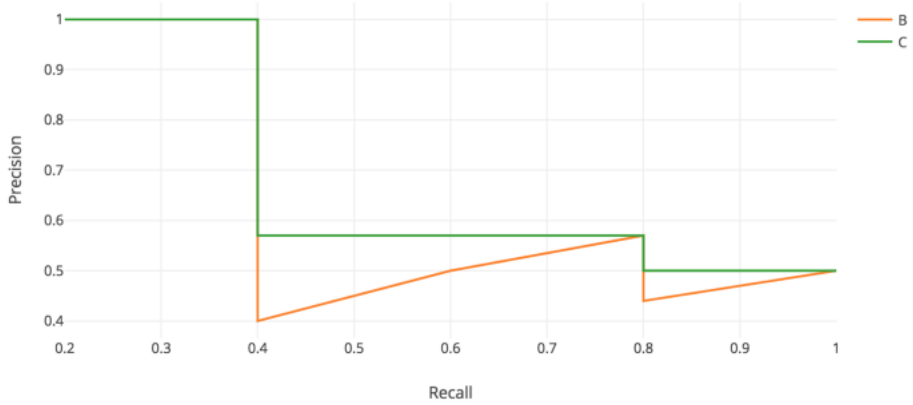


Figure 21: Precisión-Sensibilidad ponderada

Matemáticamente se calcula, para aquellas predicciones consideradas correctas (IoU mayor a un determinado threshold), como:

$$AP = \sum (r_n - r_{n-1}) P_{interp(r_n)}$$

Donde r_n es la sensibilidad para un determinado valor de IoU. Y $p_{interp}(r) = max_{r>r_n} p(r)$ ya que gráficamente reemplazamos cada valor de precisión con el valor máximo de precisión a la derecha de ese nivel de Sensibilidad.

7.1.1.4 Precisión promediada media (mAP: Mean Average Precisión)

Si el conjunto de datos contiene N clases:

$$mAP = \frac{1}{N} \sum_{class=1}^{N} (AP_{class})$$

7.1.1.5 Sensibilidad promediada (AR: Average Recall)

En vez de calcular la Sensibilidad en un valor particular de IoU, calculamos la Sesnbilidad promediada (AR) con IoU desde 0.5 a 1. Matemáticamente AR es definido como:

$$AR = 2 \int_{0.5}^{1} recall(IoU)\, d(IoU)$$

7.1.1.6 Sensibilidad promediada media (mAR: Mean Average Recall)

Si el conjunto de datos contiene N clases:

$$mAR = \frac{\sum_{i=1}^{K} (AR_i)}{K}$$

7.1.1.7 Puntaje F1 (F1 score)

El valor F1 combina las medidas de Precisión y Sensibilidad en un solo valor. Esto resultas practico ya que en un solo valor se puede comparar el rendimiento combinado por la Precisión y la Sensibilidad.

$$F1 = 2 \times \frac{Precisin \times Sensibilidad}{Precisin + Sensibilidad}$$

7.1.2 Evaluación del modulo: SISTEMA DE CONTROL DIFUSO

A partir de la simulación desarrollada se podrá chequear el funcionamiento del sistema, y reglas de control difuso.

Además se realizarán múltiples pruebas testeando el sistema de control difuso. Se colocaran obstáculos frente al robot en todas las distancias posibles según las etiquetas lingüísticas de las variables de entrada (cerca, lejos) y se evaluara la salida del sistema (ángulo a corregir) buscando obtener otra vez, todas las etiquetas lingüísticas de las variables de salida (izquierda, centro, derecha).

7.2 Resultados, versión del proyecto: 1.0

Entre las características principales de estas versión se destaca:

- El sistema de procesamiento de imágenes únicamente reconoce un tomate.
- El sistema de control difuso únicamente controla el ángulo de salida a corregir para evitar colisionar con un obstáculo.

8 CONCLUSIÓN

asdasd

9 APÉNDICE A

9.1 Matriz de requisitos

Matriz de requisitos							
ID	Requisito	Estado completo			Prioridad	Complejidad	Objetivo
		Estado	Versión	Fecha			
0	Tiene que reconocer obstáculos.	EN CURSO	1.0	2022/01/17	ALTA	ALTA	Capacidad de reconocer algunos de los obstáculos.
1	Tiene que evadir obstáculos.	EN CURSO	1.0	2022/01/17	ALTA	ALTA	Capacidad de evadir obstáculos, si el obstáculo es reconocido la evasión se hará en función de sus dimensiones.
2	Capacidad de desplazarse.	SIN COMENZAR	1.0	2022/01/17	ALTA	BAJA	Capacidad de desplazarse a través de ciertos entornos.
3	Capacidad de sensar distancia a obstáculos.	SIN COMENZAR	1.0	2022/01/17	ALTA	BAJA	Necesidad de obtener la distancia al obstáculo en su cono de visión.
4	Capacidad de conocer dirección de desplazamiento.	SIN COMENZAR	1.0	2022/01/17	ALTA	MEDIA	Necesidad de saber el sentido al cual se dirige.
5	Capacidad de medir la distancia recorrida.	SIN COMENZAR	1.0	2022/01/17	ALTA	MEDIA	Decoders ayudarán a medir la distancia recorrida.
6	Tiene que ser implementado en la Zybo.	SIN COMENZAR	1.0	2022/01/17	BAJA	MEDIA	Implementado en la placa de desarrollo Zybo.
7	Tiene que ser alimentado por batería.	SIN COMENZAR	1.0	2022/01/17	BAJA	BAJA	-.
8	Operaciones de mayor consumo computacional tienen que ser implementadas en una FPGA.	EN CURSO	1.0	2022/01/17	BAJA	ALTA	Con el propósito de ahorrar energía, acelerar las operaciones, etc.

9.2 Diagrama de Gantt

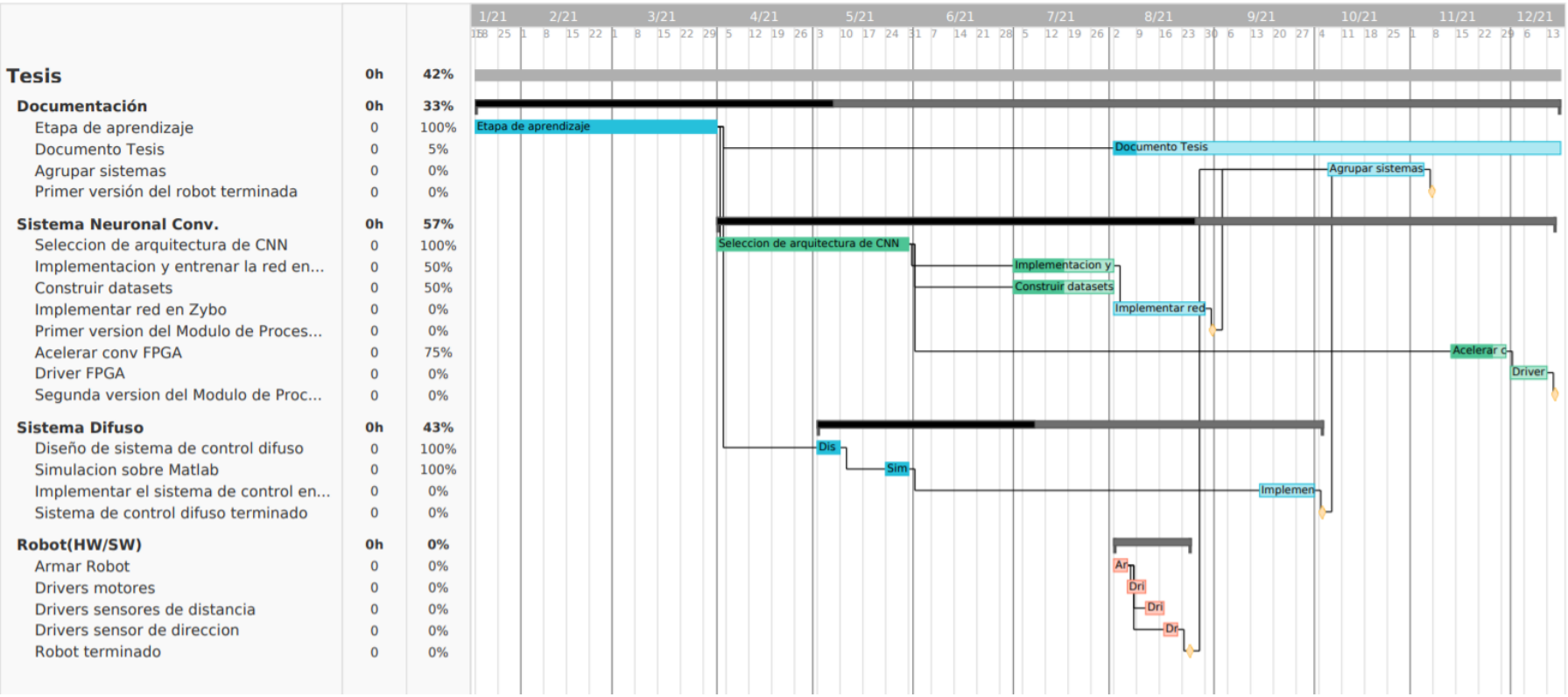


Figure 22: Diagrama de Gantt

9.3 Gestión del riesgo

La matriz de probabilidad-impacto es una herramienta de análisis cualitativo de riesgos que nos permite establecer prioridades en cuanto a los posibles riesgos de un proyecto en función tanto de la probabilidad de que ocurran como de las repercusiones que podrían tener sobre nuestro proyecto en caso de que ocurrieran.

RIESGO = Probabilidad x Impacto

			Probabilidad				
			Excepcional	Poco probable	Probable	Muy probable	Inminente
			(2)	(4)	(6)	(8)	(10)
Impacto	Extensiva	(10)	20	40	60	80	100
	Mayor	(8)	16	32	48	64	80
	Localizada	(6)	12	24	36	48	60
	Menor	(4)	08	16	24	32	40
	Leve	(2)	04	08	12	16	20

Table 1: Matriz de probabilidad por impacto

Colour	**Legenda**
	No aceptable, se requiere reducción del riesgo.
	Aceptable pero considere reducción del riesgo.
	Aceptable.

Table 2: Leyenda de colores de la matriz de riesgo

9.3.1 Riesgos técnicos

- Incorrecta selección de arquitectura de la red neuronal convolucional y/o modificación de las etapas necesarias (24)

  · Si bien es muy probable que la red a utilizar sea seleccionada en función a la cantidad de recursos disponible para poder implementarla, este riesgo puede ocasionar una reimplementación a nivel de software o hardware. Por lo tanto este riesgo tiene una probabilidad de ocurrencia **PROBABLE** y un impacto **MENOR**.
  Gestión: Mitigación, se reducirán las probabilidad de ocurrencia al mínimo con mucha investigación previa.

- Aprendizaje mayor al esperado (80)

  · Gran parte del desarrollo se llevara acabo sobre la placa de desarrollo ZYBO (Zynq Board). Durante la carrera el complejo tipo de SOC jamás fue utilizado. Además el conocimiento requerido en temas como Fuzzy Logic, redes neuronales, redes neuronales convolucionales, y HDL fueron temas solo alcanzados en algunas materias de forma muy limitada y/o jamas vistos. Este riesgo tiene una probabilidad de ocurrencia **INMINENTE** y un impacto **MAYOR**.
  Gestión: Mitigación, se reducirán las probabilidad de ocurrencia al mínimo con mucha investigación previa.

- Sensibilidad, precisión, y calidad de los sensores (32)

· Este riesgo puede ocasionar que los resultados esperados del sistema de control difuso no sea el esperado. Este riesgo tiene una probabilidad de ocurrencia **MUY PROBABLE** y un impacto **MENOR**.
Gestión: El propio sistema de control difuso mitiga la precisión de los sensores por ser un sistema de control robusto.

- Capacidad de reconocer la dirección de avance del robot (60)

  · Este riesgo es uno de los mas grandes a enfrentar, ya que el gps no brinda buenos resultados en interiores, además es necesario conocer la orientación del robot y la ubicación del target o objetivo a alcanzar. Este riesgo también puede ocasionar que los resultados esperados del sistema de control difuso no sea el esperado. Este riesgo tiene una probabilidad de ocurrencia **INMINENTE** y un impacto **LOCALIZADO**.
  Gestión: Se intentará buscar una forma óptima de geolocalización o bien simplemente el usuario definirá un punto relativo a la orientación y ubicación inicial del robot, utilizando unos decoders en los motores del robot y una brújula para conocer la dirección de avance.

- Imposibilidad de implementar el acelerado del sistema de reconocimiento del robot por hardware (16)

  · Previendo problemas al realizar la implementación y por falta de tiempo. Este riesgo tiene una probabilidad de ocurrencia **MUY PROBABLE** y un impacto **LEVE**.
  Gestión: Aceptación, se aceptaran las consecuencias del riesgo.

- Reducción de la exactitud de la red neuronal convolucional al utilizar aritmética de punto fijo (12)

  · Normalmente los parámetros de una red neuronal convolucional son flotantes de 32 o 64 bits. En caso de implementar algunas operaciones sobre la FPGA sera necesario reducir la representación a 8 bits con punto fijo, lo que provocara una reducción de la calidad de la precisión de la red. Se puede mitigar con correctas simulaciones sobre la PC y reentrenando la red. Este riesgo tiene una probabilidad de ocurrencia **PROBABLE** y un impacto **LEVE**.
  Gestión: Mitigación, existen distintos programas para prever la precisión que tendrá la red al ser implementada con aritmética de 8 bits de punto fijo.

- Dificultad al integrar los módulos desarrollados por separado y de alcanzar el objetivo esperado (80)

  · Una vez finalizado el sistema de control difuso y el modulo de procesamiento de imágenes existe el riesgo de redefinir algunas partes de los módulos para poder integrarlos. A su vez existe la posibilidad, una vez integrados los sistemas de no lograr el resultado esperado del proyecto. Este riesgo tiene una probabilidad de ocurrencia **MUY PROBABLE** y un impacto **EXTENSIVO**.
  Gestión: En caso de que mi sistema de navegación no obtenga los resultados esperado y el sistema de procesamiento de imágenes no complemente y/o realce los beneficios del sistema difuso (extienda sus capacidades) no se va a asumir ninguna acción ya que es un resultado completamente válido para un proyecto de investigación. Y simplemente se remarcara en la tesis que por ahí no es el camino.

9.3.2 Riesgos organizativos

- Disponibilidad del hardware (16)

- Al ser un hardware caro afrontado por mi, y quiza eventualmente provisto por la facultad. Este riesgo tiene una probabilidad de ocurrencia **EXCEPCIONAL** y un impacto **MAYOR**.
  Gestión: Prevención, esta amenaza se eliminara adquiriendo la placa de desarrollo.

- Errores de estimación del cronograma (48)

  - Este riesgo puede aparecer por desconocimiento y falta de experiencia en este tipo de desarrollos. Este riesgo puede ocasionar que no se llegue a cumplir con las fechas de los hitos del proyecto. Este riesgo tiene una probabilidad de ocurrencia **PROBABLE** y un impacto **MAYOR**.

- Priorizar inadecuadamente las tareas del proyecto (48)

  - Este riesgo puede aparecer si se elige incorrectamente el orden de prioridad de las tareas del proyecto, ocasionando que ciertas tareas que deberían ser necesarias para el desarrollo de otras no estén terminadas. Este riesgo tiene una probabilidad de ocurrencia **PROBABLE** y un impacto **MAYOR**.

- Omisión de tareas en el cronograma (48)

  - Pueden existir tareas que debido a su poca carga de trabajo, no hayan sido tomadas en cuenta dentro del cronograma. Esto puede ocasionar retrasos no contemplados dentro del proyecto. Este riesgo tiene una probabilidad de ocurrencia **PROBABLE** y un impacto **MAYOR**.

9.3.3 Riesgos externos

- Cuarentena (80)

  - Ante la actual situación epidemiológica nacional e internacional en relación con la infección por coronavirus (Covid-19) se pueden encontrar problemas a la hora de acceder a recursos necesarios que podría proveer la facultad. Este riesgo tiene una probabilidad de ocurrencia **INMINENTE** y un impacto **MAYOR**.

Riesgo	ID	Probabilidad	Impacto	Riesgo	Detalle	Gestión
Incorrecta selección de arquitectura de la red neuronal convolucional y/o modificación de las etapas necesarias	0	PROBABLE	MENOR	24	Si bien es muy probable que la red a utilizar sea seleccionada en función a la cantidad de recursos disponible para poder implementarla, este riesgo puede ocasionar una reimplementación a nivel de software o hardware.	Mitigación, se reducirán las probabilidad de ocurrencia al mínimo con mucha investigación previa.
Aprendizaje mayor al esperado	1	INMINENTE	MAYOR	80	Gran parte del desarrollo se llevara acabo sobre la placa de desarrollo ZYBO (Zynq Board). Durante la carrera el complejo tipo de SOC jamás fue utilizado. Además el conocimiento requerido en temas como Fuzzy Logic, redes neuronales, redes neuronales convolucionales, y HDL fueron temas solo alcanzados en algunas materias de forma muy limitada y/o jamas vistos.	Mitigación, se reducirán las probabilidad de ocurrencia al mínimo con mucha investigación previa.
Sensibilidad, precisión, y calidad de los sensores	2	MUY PROBABLE	MENOR	32	Este riesgo puede ocasionar que los resultados esperados del sistema de control difuso no sea el esperado.	El propio sistema de control difuso mitiga la precisión de los sensores por ser un sistema de control robusto.
Capacidad de reconocer la dirección de avance del robot	3	INMINENTE	LOCALIZADO	60	Este riesgo es uno de los mas grandes a enfrentar, ya que el gps no brinda buenos resultados en interiores, además es necesario conocer la orientación del robot y la ubicación del target o objetivo a alcanzar. Este riesgo también puede ocasionar que los resultados esperados del sistema de control difuso no sea el esperado.	Se intentará buscar una forma óptima de geolocalización o bien simplemente el usuario definirá un punto relativo a la orientación y ubicación inicial del robot, utilizando unos decoders en los motores del robot y una brújula para conocer la dirección de avance.
Imposibilidad de implementar el acelerado del sistema de reconocimiento del robot por hardware	4	PROBABLE	LEVE	12	Previendo problemas al realizar la implementación y por falta de tiempo.	Aceptación, se aceptaran las consecuencias del riesgo.
Reducción de la exactitud de la red neuronal convolucional al utilizar aritmética de punto fijo	5	PROBABLE	LEVE	12	Normalmente los parámetros de una red neuronal convolucional son flotantes de 32 o 64 bits. En caso de implementar algunas operaciones sobre la FPGA sera necesario reducir la representación a 8 bits con punto fijo, lo que provocara una reducción de la calidad de la precisión de la red. Se puede mitigar con correctas simulaciones sobre la PC y reentrenando la red.	Mitigación, existen distintos programas para prever la precisión que tendrá la red al ser implementada con aritmética de 8 bits de punto fijo.

Riesgo	ID	Probabilidad	Impacto	Riesgo	Detalle	Gestión
Dificultad al integrar los módulos desarrollados por separado y de alcanzar el objetivo esperado	6	MUY PROBABLE	EXTENSIVO	80	Una vez finalizado el sistema de control difuso y el modulo de procesamiento de imágenes existe el riesgo de redefinir algunas partes de los módulos para poder integrarlos. A su vez existe la posibilidad, una vez integrados los sistemas de no lograr el resultado esperado del proyecto.	En caso de que mi sistema de navegación no obtenga los resultados esperado y el sistema de procesamiento de imágenes no complemente y/o realce los beneficios del sistema difuso (extienda sus capacidades) no se va a asumir ninguna acción ya que es un resultado completamente válido para un proyecto de investigación. Y simplemente se remarcara en la tesis que por ahí no es el camino.
Disponibilidad del hardware	7	EXCEPCIONAL	MAYOR	16	Al ser un hardware caro afrontado por mi, y quiza eventualmente provisto por la facultad.	Prevención, esta amenaza se eliminara adquiriendo la placa de desarrollo.
Errores de estimación del cronograma	8	PROBABLE	MAYOR	48	Este riesgo puede aparecer por desconocimiento y falta de experiencia en este tipo de desarrollos.	
Priorizar inadecuadamente las tareas del proyecto	9	PROBABLE	MAYOR	48	Este riesgo puede aparecer si se elige incorrectamente el orden de prioridad de las tareas del proyecto, ocasionando que ciertas tareas que deberían ser necesarias para el desarrollo de otras no estén terminadas.	
Omisión de tareas en el cronograma	10	PROBABLE	MAYOR	48	Pueden existir tareas que debido a su poca carga de trabajo, no hayan sido tomadas en cuenta dentro del cronograma. Esto puede ocasionar retrasos no contemplados dentro del proyecto.	
Cuarentena	11	INMINENTE	MAYOR	80	Ante la actual situación epidemiológica nacional e internacional en relación con la infección por coronavirus (Covid-19) se pueden encontrar problemas a la hora de acceder a recursos necesarios que podría proveer la facultad.	

9.4 Plan de calidad

Cuando hablamos de calidad hablamos del grado de cumplimiento que tiene un proyecto respecto a sus requisitos. Es importante remarcar que un proyecto no cumple con los requisitos tanto cuando no llega a conseguir estos, como cuando los excede.
Los requisitos normalmente se pueden dividir en dos grupos.

9.4.1 Requisitos del proyecto

Aquellos requisitos relativos al proceso de trabajo o forma de gestionar el proyecto que este debe seguir por el hecho de ser llevado a cabo dentro de la institución.

9.4.2 Requisitos del producto

Aquellas características que debe cumplir el producto resultante del proyecto. Con el fin de satisfacer ciertos requerimientos puestos por el proyecto de investigación y llevar a cabo la gestión del proyecto se comienza por definir el alcance del proyecto con relación a dos aspectos.

9.4.3 ¿Qué características debe cumplir el producto? ¿Cómo se comprobara que este cumple con estas características?

Con la intención de definir las características que se deben cumplir de forma cuantificable y medible se lleva a acabo la descripción de los requisitos.

* El robot móvil a desarrollar deberá de poder reconocer al menos 3 obstáculos de distintas dimensiones y formas capaz de evadir, a su vez como el propósito del proyecto de investigación es mejorar la toma de decisión a la hora de evasión en función de las dimensiones del obstáculo y de la meta alcanzable por el robot se harán numerosas pruebas para testear que evada los distintos obstáculos de la forma adecuada.

  A su vez, como el robot se ira desplazando, el modulo de procesamiento de imágenes sera sometido a distintos tipos de entornos. Por este motivo se hará un sistema robusto intentando que el robot sea capaz de reconocer el o los obstáculos frente la mayor cantidad de veces posibles antes de que tome la decisión de como esquivarlo. Se va a estudiar la exactitud de la red propuesta, la referencia [1] promete una exactitud del 80.3% para el Top-5 y una exactitud del 57.5% para el Top-1.

* En cuanto a la movilidad y capacidades sensitivas el robot tendrá que ser capaz de desplazarse, medir la distancia recorrida, la dirección y evaluar constantemente la distancia a los obstáculos mas cercanos por delante de el. Se estudiará la precisión de los sensores de distancia, la precisión al medir la distancia recorrida y del sensor que indicara la dirección de movimiento.

* Por cuestiones de recursos, el sistema tiene que ser implementado sobre una placa de desarrollo Zybo.

* Por cuestiones de consumo de energía y aceleración de toma de decisiones el sistema se procesarán ciertas operaciones en la lógica programable de la Zybo.

10 APÉNDICE B

List of Figures

List of Tables

References

[1] Forrest N. Iandola1 - Song Han2 - Matthew W. Moskewicz1 - Khalid Ashraf1 - William J. Dally2 - Kurt Keutzer1. ""SQUEEZENET"". In: _ (2016).