Universidad Internacional de La Rioja (UNIR)

Escuela de Ingeniería

Máster en Análisis y Visualización de Datos Masivos

Estudio de arquitecturas para la extracción y explotación de datos de defectos superficiales mediante técnicas de Deep Learning

Trabajo Fin de Máster

presentado por: Saiz Álvaro, Fátima Aurora

Director: Pedraza Gómara, Luis

Codirector: Sánchez Tapia, Jairo Roberto

Ciudad: Bilbao
Fecha: 26 Julio 2018

Resumen

En este trabajo se realiza un estudio de arquitecturas de extracción y explotación de datos sobre defectos superficiales producidos en la laminación del acero mediante técnicas de *Deep Learning,* así como el almacenamiento de los datos en una arquitectura de *Big Data* y su explotación empleando herramientas de *Visual Analytics* que permiten tomar decisiones ágiles. Para ello se adquieren los datos con técnicas de visión por computador y se realizan experimentos para configurar la explotación de los datos empleando redes neuronales y se comparan los resultados y precisiones obtenidos con los del Estado del Arte actual, comprobando que son mejorados. Se diseña una arquitectura para almacenar los datos de la captura que se adapta a las necesidades de producción, garantizando la escalabilidad, la seguridad y la rapidez. Por último, se desarrollan visualizaciones enfocadas a diferentes roles de personas en la producción que aportan conocimiento sobre el estado de la fabricación y permiten mejorar el proceso.

Palabras Clave: Deep Learning, Big Data, Visual Analytics, Defectos Superficiales

Abstract

In this work a study of data extraction and mining architectures applied to superficial defects produced in the steel-rolling using Deep Learning techniques is carried out, as well as the storage of the data in a Big Data architecture and its exploitation using Visual Analytics tools that allow making agile decisions. For this purpose, the acquisition of data using computer vision is performed. Then, some experiments to configure the ideal acquisition of the data using neural networks are carried out. The obtained results and accuracies are compared with those of the current State of the Art, surpassing them. An architecture is designed according the production needs to store captured data. This architecture guarantees scalability, security and speed. Finally, some visualizations focused on different people-roles in production environment are developed to provide knowledge about the state of manufacturing that allows to improve the process.

Keywords: Deep Learning, Big Data, Visual Analytics, Superficial Defects

Índice de contenido

Índice de ilustraciones

Índice de tablas

1. Introducción

Durante este trabajo se ha realizado un estudio sobre la extracción de datos empleando técnicas de *Computer Vision* y la detección de defectos superficiales mediante ***Deep Learning*** para construir sistemas de ***Visual Analytics*** respaldados por infraestructuras de almacenamiento basadas en ***Big Data*** que faciliten la toma de decisiones en procesos de laminación de acero. En los siguientes apartados se expondrá la problemática a tratar, así como las características de la solución propuesta y los principales resultados obtenidos. Por último, se explicará la estructura de la memoria.

1.1. Motivación

En la actualidad, saber extraer conocimiento de los datos que posee la empresa es un factor estratégico que permite posicionarse en el sector y corregir errores en la producción. Este trabajo en concreto se centra en la producción del acero laminado por su gran volumen de fabricación y sus múltiples utilidades. En los últimos años, hay una mayor necesidad de realizar un control de calidad en el sector de la fabricación. Esto es debido a que las industrias que emplean material en sus productos como son la aeroespacial o la automotriz, rechazan los materiales defectuosos por el desastre que podría provocar un defecto menor en una pieza fabricada en una etapa posterior.

En la fabricación y en la manipulación del acero se generan diversos defectos, entre ellos defectos superficiales que son los más comunes, un ejemplo de estos son el óxido o las grietas. Además, la heterogeneidad de los defectos que pueden aparecer es un reto para su detección y clasificación. Por eso es necesario un control de estos defectos de manera automática y a tiempo real.

Actualmente, para realizar ese control de defectos, se emplean sistemas de visión artificial y de aprendizaje automático con el fin de descubrir patrones a partir del conjunto de datos que se genera en los procesos de producción. Estos sistemas ofrecen información valiosa que reducen la incertidumbre de la causa de los errores de fabricación. Sin embargo, las técnicas clásicas empleadas en la detección no se adaptan correctamente a la variabilidad de defectos existentes ni a las altas cadencias de producción.

En la actualidad se está evolucionando hacia un nuevo enfoque en las técnicas de fabricación que se conoce como Industria 4.0. Bajo este paradigma, cada vez se da más importancia a los datos que generan las máquinas debido a la información acerca del proceso que proporcionan. Para asumir este nuevo enfoque se requiere una arquitectura de almacenamiento sólida y capaz de adaptarse al ritmo de crecimiento. El empleo de esta arquitectura nace de la necesidad de almacenar todas las imágenes en continuo al emplear

sistemas de visión artificial, ya que estas son necesarias para el aprendizaje automático. Además, esta arquitectura *Big Data* debe permitir realizar consultas a la mayor rapidez posible, ya que los datos se actualizan de manera continua.

Para tratar estos datos, se abordan diferentes tendencias para gestionar grandes volúmenes de datos para lograr una gran productividad. [1]

El tener disponibles los datos de fabricación permite analizar los problemas que surgen en el proceso, pudiendo alcanzar el objetivo de adaptarse a los requerimientos de calidad del producto y las exigencias que impone la competencia. Para estudiar estos datos se emplean visualizaciones gráficas que permitan interpretar los datos sin tener grandes conocimientos en el sector.

1.2. Planteamiento del trabajo

Para mejorar en la extracción de defectos realizada normalmente hasta la actualidad empleando técnicas de *Machine Learning*, se propone el uso de técnicas de *Deep Learning*, como son las Redes Neuronales Convolucionales. Con esto se propone mejorar la velocidad y la precisión en la detección. Debido a la variabilidad de los defectos, se realizan diferentes experimentos para configurar correctamente los parámetros de la red neuronal, pudiendo así obtener el máximo rendimiento posible para esta casuística. Para constatar el incremento de la precisión con la configuración realizada, se utiliza una base de datos pública de imágenes que es empleada en varios artículos científicos y se contrastan los resultados.

Con el propósito de reducir los desperdicios en los procesos de producción y mejorar la calidad y el rendimiento de los productos fabricados, se propone el uso de una arquitectura *Big Data.* Esta debe ser capaz de absorber el volumen de datos que se genera en la línea de producción y debe poder vincular a otras plataformas, pudiendo así entrenar la red y generar las visualizaciones de manera directa. Este sistema debe contar con un respaldo de seguridad de los datos y un control a su acceso.

Debido a la complejidad de los procesos de producción, es necesario emplear técnicas de analítica avanzada para diagnosticar y corregir los defectos del proceso. Por ese motivo, para poder extraer conocimiento de los datos capturados, se propone la aplicación de estadísticas a los datos para evaluar el estado de la fábrica. Estos resultados serán plasmados en visualizaciones enfocadas a diferentes roles de personas en la producción que permitan tomar decisiones de forma ágil mejorando así la eficiencia de la línea.

Los resultados obtenidos demuestran que el uso de redes neuronales en la detección y clasificación de defectos superficiales en el acero supone una mejora respecto a las técnicas

empleadas hasta el momento. Además, los parámetros de configuración utilizados aumentan la precisión en la obtención, adaptándose a la problemática presentada.

La arquitectura elegida se adecua a los requerimientos de producción y al gran volumen de datos que se producen en la línea. Disponer de estos datos en un sistema estable y bien organizado facilita la tarea de analítica, permitiendo extraer conocimiento para resolver todo tipo de problemas de producción.

Este sistema *Big Data* junto a las visualizaciones son una herramienta crítica para realizar mejoras en el rendimiento, concretamente en este entorno de fabricación con procesos complejos. Se ha observado que el tener resultados continuamente actualizados y de manera visual permite tomar decisiones que mejoren la garantía de calidad, gestionar la cadena de suministro y mejorar los procesos de fabricación.

El esquema del planteamiento comentado en los párrafos anteriores se muestra en la Ilustración 1.

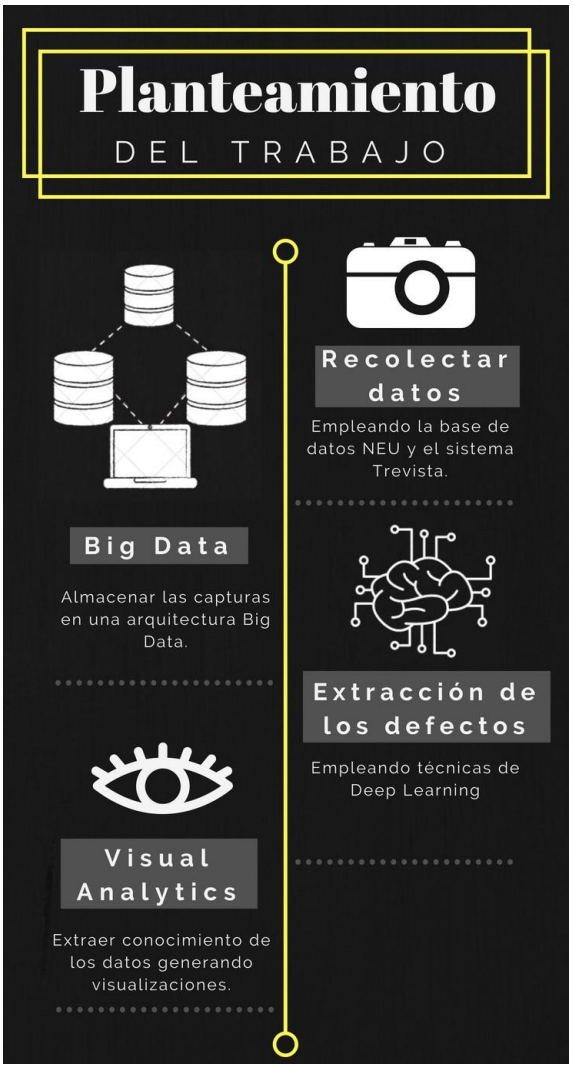


Ilustración 1. Esquema del planteamiento del trabajo.

1.3. Estructura del trabajo

El resto del documento está estructurado como aquí se indica:

En el Capítulo 2 se estudia el contexto de aplicación del trabajo revisando el control de calidad en la fabricación del acero laminado. Además, se estudian las técnicas de *Machine Learning* aplicadas a la inspección superficial y la visualización de los datos en un entorno de producción.

En el Capítulo 0 se analiza el Estado del Arte en temas de *Machine Learning*, *Deep Learning* y sus herramientas, los sistemas de almacenamiento y las técnicas de *Visual Analytics* actuales.

Posteriormente, en el Capítulo 4 se definen los objetivos del trabajo y la metodología propuesta para alcanzarlos.

En el Capítulo **Error! Reference source not found.** se explican las contribuciones del trabajo, así como el desarrollo de las comparativas realizadas y los resultados obtenidos de ellas. En él, se analizan las ventajas del uso de nuevas técnicas de captura y la adquisición de los datos a partir de *Computer Vision*. Se plantea una arquitectura de almacenamiento que sea capaz de albergar datos de estas dimensiones y se adapte a las altas cadencias de producción. Posteriormente, para la extracción de defectos, se realizan experimentos que permiten configurar las redes neuronales de manera óptima para esta problemática. Se realiza también una comparativa de las arquitecturas de Deep Learning más empleadas en el análisis de imágenes que mejor se adaptan al reconocimiento de estos defectos, superando en precisión y rapidez a los métodos empleados hasta la fecha. Por último, se realizan diferentes visualizaciones dirigidas a diferentes roles de personas en la producción que permiten generar conocimiento sobre el proceso.

Por último, se exponen las conclusiones obtenidas en el Capítulo 0 y las líneas futuras de trabajo en el Capítulo 7.

2. Análisis del contexto de aplicación

En este capítulo se presenta una serie de conceptos fundamentales para entender el resto del documento, todos ellos relacionados con la problemática tratada y el contexto de la aplicación abordada. En primer lugar, se describe y analiza el proceso de fabricación del acero, donde se resalta la importancia que tiene este material en la sociedad y sus múltiples campos de uso, los posibles problemas que se pueden producir en las etapas de fabricación y los procesos para el control de calidad.

2.1. La fabricación del acero laminado

La producción de acero bruto en España alcanzó los 14,4 millones de toneladas en 2017, suponiendo esto un incremento del 5,7% con respecto al año anterior según la Unión de Empresas Siderúrgicas (UNESID) [2]. Esto es debido a los múltiples usos que tiene este material, entre los que se pueden destacar la construcción de edificios e infraestructuras, o el sector de la manufactura. El reparto de estos usos en el 2016 se muestra en la Ilustración 2, donde se puede observar que en 2016 el principal uso de este material fue destinado a la construcción, mientras que otros sectores relevantes fueron la industria automotriz, la fabricación de maquinaria o la fabricación de objetos con componentes metálicos.

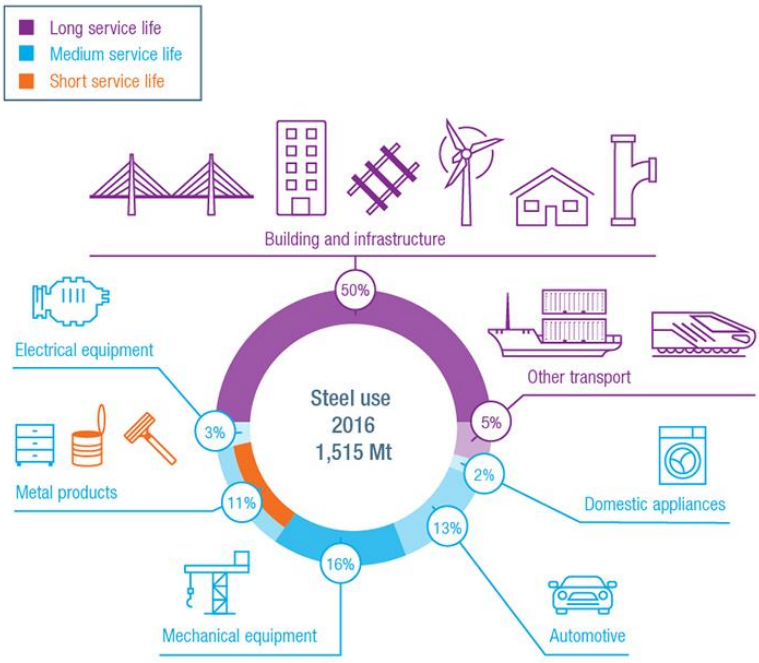


Ilustración 2. Uso del acero en el año 2016 a nivel mundial.
Fuente: WorldSteel (https://www.worldsteel.orgl)

En función del uso que se le vaya a dar, existen diferentes tipos de acero que ofrecen diversos resultados dependiendo de su composición, pudiendo conseguir así un material más resistente a la temperatura, impacto, corrosión, etc.

En este caso, el estudio se centra en el acero laminado. El proceso de laminación es un proceso de conformado de metales mediante deformación plástica que consiste en hacer pasar el material entre dos o más rodillos que giran en sentidos contrarios entre sí y ejercen presión. Un esquema de este proceso se muestra en la Ilustración 3. Existen dos tipos de laminación, la laminación en frío y en caliente. Estos dos tipos obtienen resultados diferentes, por tanto, su elección dependerá de la aplicación que se le quiera dar.

La principal diferencia entre ambos tipos es la temperatura a la que se produce, siendo muy elevada en la laminación en caliente (superando los 900ºC aproximadamente que es mayor a la temperatura de recristalización) y a temperatura ambiente en la laminación en frío. Debido a estas diferencias, en cada proceso el material sufre diferentes alteraciones. El acero laminado en caliente al enfriarse se contrae, lo que ocasiona rugosidades y ligeras distorsiones que hacen que no resulte apropiado en aplicaciones donde la precisión en las dimensiones del material es muy relevante. El acero laminado en frío se obtiene a partir del acero laminado en caliente. Caber resaltar el hecho de que el acabado superficial y la precisión dimensional conseguidas en el caso de la laminación en frío son más elevados, a costa de un mayor consumo energético y la máxima reducción de espesor es menor. Además, necesita de tratamientos térmicos posteriores para eliminar las tensiones internas de la estructura metálica obtenida. [3]

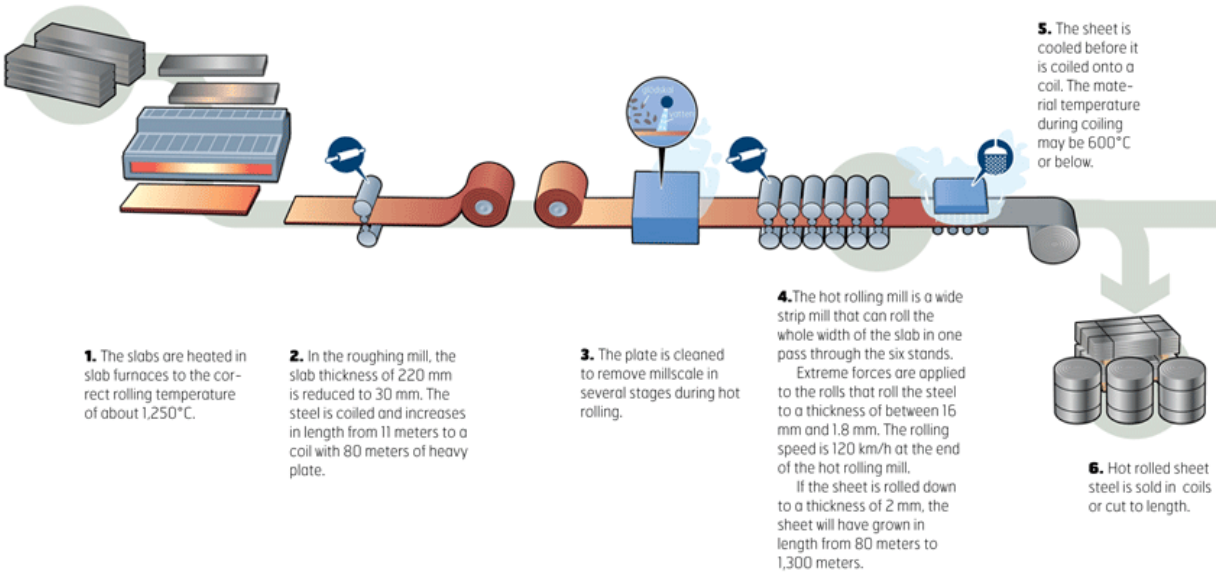


Ilustración 3. Proceso de fabricación del acero laminado en caliente.
Fuente: SSAB (https://www.ssab.es/)

En el proceso de fabricación pueden surgir diversos defectos:

- **Metalúrgicos**: afectan a la estructura del material, por ejemplo, las porosidades que pueden generar los gases como se muestra en la Ilustración 4.

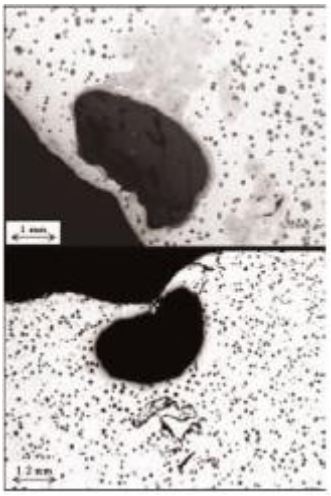


Ilustración 4. Poros de gas asociados a la presencia de escorias en superficies metálicas.
Fuente: [54]

- **Superficiales**: se refieren más al aspecto del material, pudiendo provocar daños mecánicos. Un ejemplo de ellos son el óxido o las manchas.
- **Dimensionales**: afectan a las dimensiones y a la forma geométrica del material, como ondulaciones o espesores incorrectos

2.1. El control de calidad en la fabricación del acero laminado

Los productos fabricados requieren de un control de calidad que asegure la condición óptima de los mismos. Para realizar estos controles, es fundamental tener un conocimiento del proceso de fabricación y la posibilidad de ejercer un cierto control sobre las posibles causas que puedan originar defectos en el material mencionados anteriormente en el Capítulo 2. El control realizado en los productos del acero comprende:

- Un análisis visual por parte del operario. (Defectos superficiales)
- Estudios sobre su composición química. (Defectos estructurales)
- Estudios sobre sus características mecánicas. (Defectos estructurales)
- Estudio de las características geométricas. (Defectos dimensionales)

Para obtener esta información se recogen una serie de muestras y se realizan diversos ensayos, cuyos resultados se compararán con unos criterios de aceptación o rechazo establecidos según la normativa o las exigencias del cliente.

Los ensayos se deben adecuar a las necesidades de producción. En este caso el trabajo se centrará de manera más concreta en aquellos ensayos destinados a analizar el estado superficial del material. Entre ellos, se puede citar el uso de la microscopía óptica, los análisis de acabado superficial en cuanto a rugosidad, o la inspección por líquidos penetrantes o partículas magnéticas para detectar grietas.

Unos de los ensayos practicados en el análisis de acabado superficial es el análisis visual. Una de las principales ventajas de emplear la capacidad visual que poseen los humanos para la localización de estos defectos es la habilidad precisa que tienen para buscar patrones o percibir tonos y colores. Debido a las velocidades a las cuáles se produce el acero laminado, uno de los métodos más empleados es **automatizar la inspección visual mediante cámaras**. Este es el método de inspección escogido en este trabajo, y por este motivo se describirá en mayor detalle en el apartado siguiente.

En el flujo de fabricación de acero, hay varias etapas en las que se pueden producir los defectos en la superficie. La gran variedad de defectos superficiales que pueden darse para todos los diferentes productos de acero existentes hace que la clasificación sea difícil. Los defectos más reseñables son: cascarilla, fricción, óxido, manchas, recubrimiento, grietas, sobrellenado, arañazos, picadas, escamas o suciedad, de los cuales se muestra un ejemplo de algunos de ellos en la Ilustración 5. Cabe mencionar que algunos de estos defectos implican algún tipo de rugosidad superficial, o información que debe ser reconstruida en el espacio 3D.

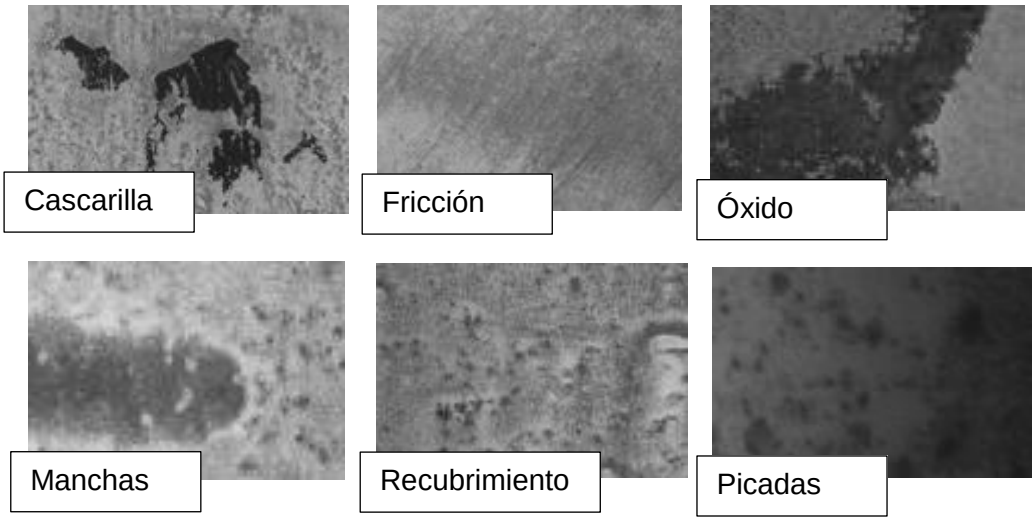


Ilustración 5. Ejemplo de algunos de los defectos superficiales del acero.

Estos defectos son aún más críticos cuando el producto fabricado se utiliza como una parte mecánica y afectan a su rendimiento funcional, llegando a poderse producir roturas en el

material que hagan que se tenga que sustituir la pieza que contenga estos defectos, lo que conduce a una gran pérdida de dinero y tiempo.

2.1.1. Inspección superficial con visión artificial

En la inspección superficial se emplean sistemas de visión artificial que están compuestos principalmente por dos elementos fundamentales, **la adquisición de las imágenes y el procesamiento**. El esquema común empleado en la inspección de defectos se muestra en la Ilustración 6. Esta muestra el proceso que se lleva a cabo en la inspección realizada sobre la lámina. En ella se puede observar una primera fase centrada en el proceso de adquisición de las imágenes superficiales, donde los elementos hardware, como las cámaras CCD o el sistema de iluminación son fundamentales, y una segunda etapa de procesamiento, donde las imágenes son analizadas empleando diferentes algoritmos y se producen los informes de calidad del proceso.

La fase de adquisición es decisiva en la calidad de los resultados obtenidos en el procesamiento, ya que una imagen de buena calidad que resalte los defectos simplificará el procesamiento posterior. Normalmente, se emplean cámaras de alta resolución junto a una fuente de iluminación. Debido a la gran variedad de defectos y de la dificultad de observar la información tridimensional de algunos de ellos, se emplean nuevas técnicas de captura que amplían la información que se obtiene en una manera tradicional.

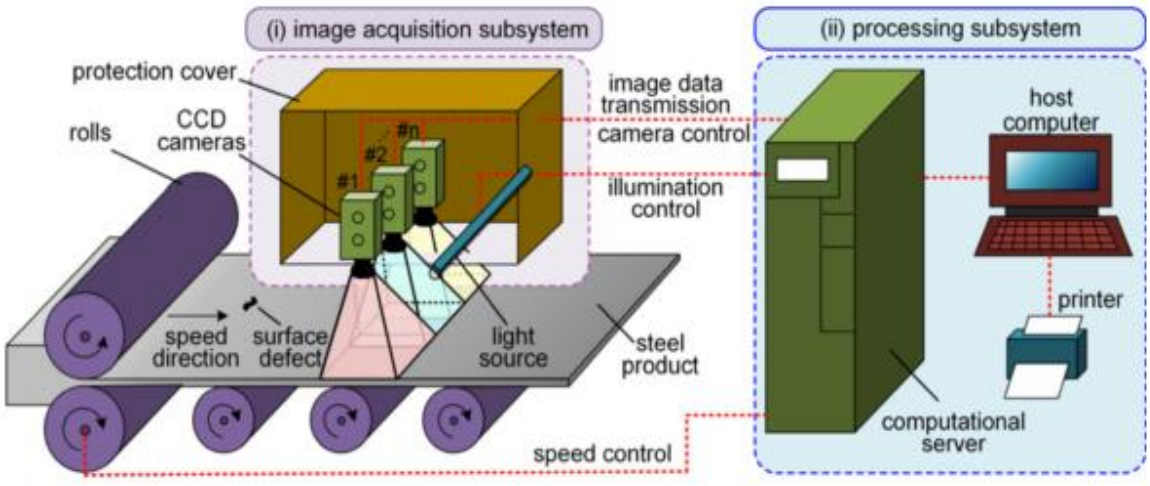


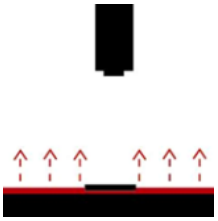
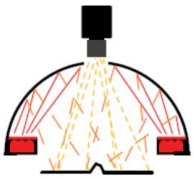
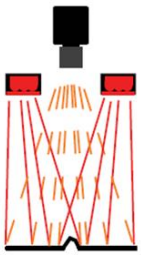
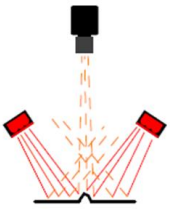
Ilustración 6. Esquema de visión artificial en la inspección de defectos superficiales en las bobinas de acero. Fuente: [55]

Existen diferentes **tipos de iluminación** en función del objetivo perseguido y de la geometría del objeto a capturar. Los parámetros principales de un iluminador son su color y su disposición. Cabe destacar que en el ámbito industrial es importante asegurar la estabilidad del iluminador ya que con el paso del tiempo su comportamiento puede variar de forma

significativa. En los casos extremos esta variación puede implicar que el sistema deje de hacer su función.

En la Tabla 1 se enumeran algunos de los tipos más significativos de iluminadores en función de su disposición.

Tabla 1 Tipos más significativos de iluminación.

Tipo de iluminación	Definición	Esquema
Back light	Generar un alto contraste entre el fondo y los objetos a inspeccionar. En esta configuración el iluminador se coloca tras el objeto en dirección a la cámara.	
Diffuse light	Ilumina el objeto de forma indirecta desde todas las direcciones. Resulta particularmente efectiva para inspeccionar superficies curvas y materiales altamente especulares.	
Bright field light	Ilumina el objeto de forma lateral de forma que el ángulo de incidencia de la luz se sitúa dentro del cono de visión de la cámara. Es muy útil para resaltar detalles topológicos de la pieza, como relieves. Presenta problemas con superficies altamente especulares debido al reflejo que produce en su superficie.	
Dark field light	Ilumina el objeto de forma lateral, pero con un ángulo de incidencia bajo para conseguir que la luz se refleje fuera del cono de visión de la cámara. Esta técnica resulta particularmente efectiva con materiales altamente especulares, como metales.	

Combinando las fuentes de iluminación y la captura de alta resolución se introduce una técnica de captura nueva como es el estéreo fotométrico, empleando el **sistema Trevista** [4], mostrado en la Ilustración 7, el cual está montado sobre un banco lineal ya que la cámara empleada captura de forma lineal.

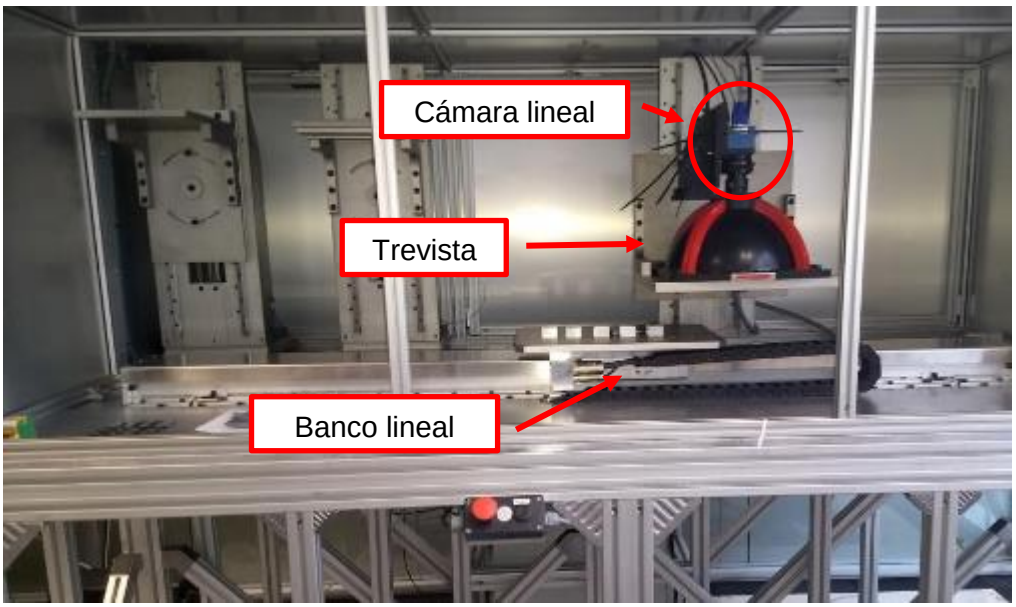
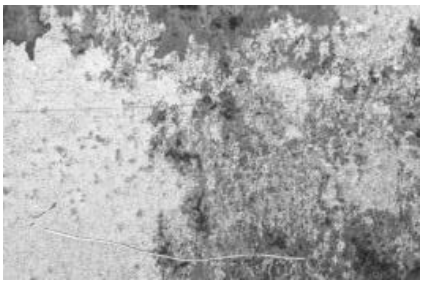
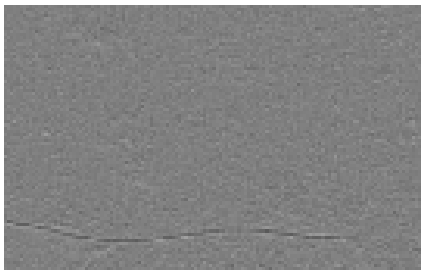
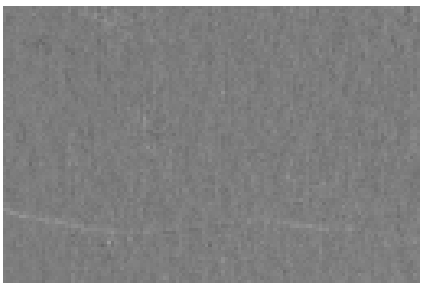
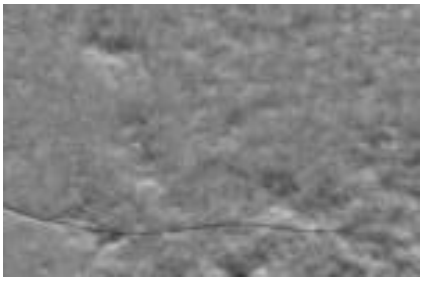


Ilustración 7. Sistema de captura Trevista.

Aun siendo un sistema 2D, es posible extraer información de la superficie de los objetos a partir de las sombras que se generan en la captura. De esta manera, se logra analizar de manera precisa los daños superficiales independientemente del material de la superficie. Para la iluminación se emplea un difusor en forma de domo que trabaja con diversas fuentes de luz con diferentes orientaciones. Este sistema obtiene cinco imágenes diferentes, mostradas en la Tabla 2, junto a una descripción de su adquisición.

Tabla 2. Ejemplo de imágenes obtenidas con la Trevista y descripción de su adquisición.

Nombre de la imagen	Imagen	Descripción
Textura		Imagen captada con iluminación coaxial
Curvatura		Iluminación estereofotométrica
Gradiente X		Imagen intermedia generada para componer las imágenes estereofotoméricas
Gradiente Y		Imagen intermedia generada para componer las imágenes estereofotoméricas
Rango		Iluminación estereofotométrica

2.1.2. *Machine Learning* en la inspección visual del acero

Para abordar la extracción de características y la clasificación se emplean técnicas de *Machine Learning*. El *Machine Learning* o aprendizaje automático es el grupo de técnicas que tienen como objetivo desarrollar métodos que permitan a los ordenadores aprender, creando programas que induzcan el conocimiento. Estos programas se basan en el análisis de datos y en la estadística, buscando patrones en esos datos para modelar el programa que permita el aprendizaje.

Estos programas tienen diferentes aplicaciones en muchos ámbitos que van desde la medicina, la robótica, la clasificación de imágenes, análisis de mercado o la detección de objetos, por ejemplo.

Existen diferentes enfoques del *Machine Learning* dependiendo de la problemática a solucionar. Se pueden emplear algoritmos genéticos, árboles de decisión, reglas de asociación, *Deep Learning*, etc.

En este trabajo se propone el uso del *Deep Learning* para solucionar el problema planteado.

2.1.3. Almacenamiento, visualización y análisis de la información

Debido a que la laminación es un proceso continuo de alta velocidad, se precisa de métodos de almacenamiento capaces de albergar todo el volumen de datos que se genera, así como métodos que permitan visualizar el estado de la producción de forma ágil.

Debido a estas características y a la capacidad de crecimiento de los datos capturados, este contexto de aplicación se puede considerar como *Big Data*.

Uno de los usos más importantes del *Big Data* es generar conocimiento a partir de los datos, ya que permite mediante herramientas analíticas predecir comportamientos y conocer mejor la empresa. En cuanto a los defectos superficiales, este análisis va a servir para conocer el estado de la producción del material y la calidad del material recibido, pudiendo sacar conclusiones sobre los procesos que puedan generar estos errores, como se puede apreciar en la Ilustración 8. El objetivo último es mejorar el proceso para evitar que estos defectos se produzcan, mejorando la calidad de producto final y reduciendo el volumen de chatarra.

Ilustración 8. Usos de la analítica de los datos de producción en una fábrica.
Fuente: https://grupogaratu.com/

Para extraer conclusiones de la gran cantidad de información recibida, es preciso emplear técnicas de visualización de la información que permitan conocer lo que los datos quieren transmitir de una manera sencilla y rápida. La evolución de las tecnologías permite analizar los datos recabados de forma masiva, solventando así diversos retos tecnológicos.

En el contexto de una planta de producción real, la visualización de los datos tiene aplicaciones diferentes. Como se muestra en la Ilustración 9**Error! Reference source not found.**, se necesita de un sistema que recoja todos los datos del sistema de producción, y los

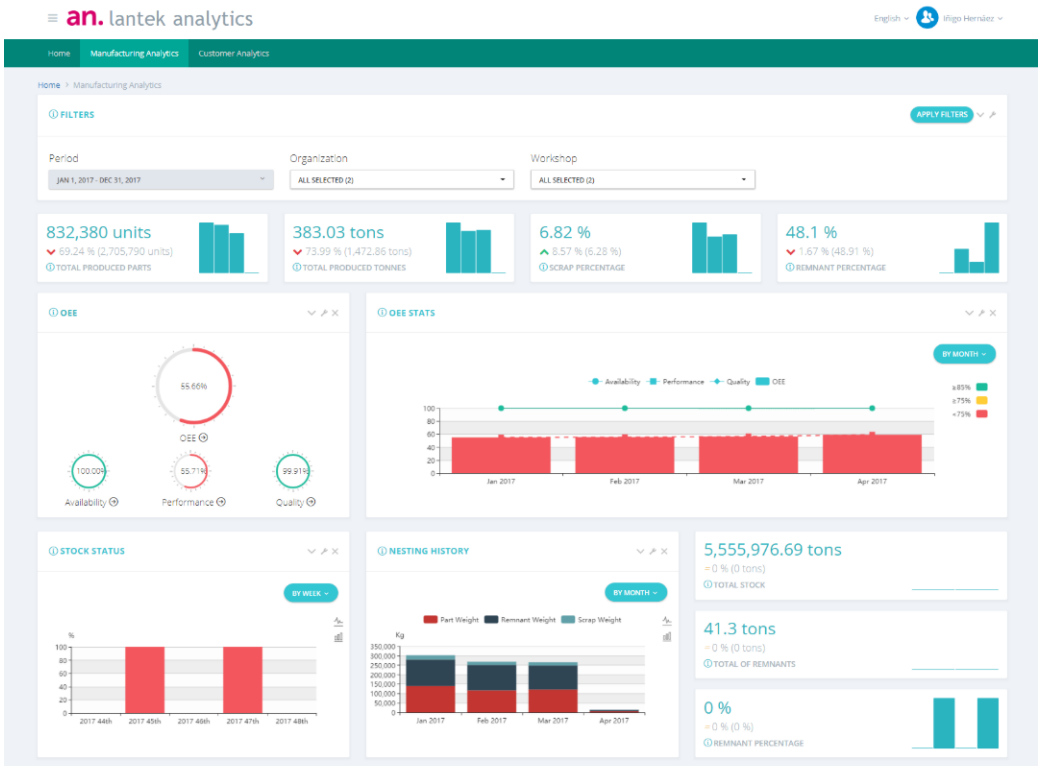


Ilustración 9. Módulo de gestión de datos en un entorno de producción.
Fuente: https://www.lanteksms.com/es/Manufacturing-Analytics

filtre y relacione para permitir la toma de decisiones basándose en información fiable y siempre actualizada.

Además, los datos obtenidos tienen diferentes objetivos en su representación. Es necesario obtener una visualización general que permita transmitir al gestor de planta información sobre el estado general de la producción en tiempo real. También es necesario tener una visualización de las incidencias que se puedan producir para poder subsanarlas lo antes posible. Otra visualización necesaria son los indicadores clave marcados por la empresa, para conseguir el nivel de calidad deseado. Existen más visualizaciones según la necesidad de la empresa y de la naturaleza del trabajo, que pueden requerir visualizaciones más concretas al detalle.

3. Estado del arte

En este trabajo se aborda el problema de la automatización en la detección y clasificación de defectos superficiales mediante la combinación de técnicas de visión artificial y algoritmos de Deep Learning. Los sistemas de visión artificial no sólo se emplean con el fin de automatizar y acelerar ciertos procesos de fabricación a nivel industrial, sino que permiten detectar y reducir las deficiencias en el proceso y en la calidad de los productos obtenidos.

El estado del arte se dividirá siguiendo las etapas marcadas para abordar la problemática. Primero se estudiará el *Machine Learning* y los métodos más importantes para la extracción de características en las imágenes y los métodos de clasificación más empleados. Posteriormente, se desarrollará el *Deep Learning*, y en concreto las redes neuronales convolucionales, así como las herramientas existentes para el entrenamiento. Seguidamente se analizarán las alternativas de almacenamiento masivo de información disponibles en el mercado. Finalmente, se revisará el estado de la visualización de los datos en fábrica.

3.1. Machine Learning

El *Machine Learning* es un conjunto amplísimo de técnicas que crean sistemas informáticos que aprenden automáticamente, ubicándose así dentro del ámbito de la Inteligencia Artificial. Con el uso de algoritmos matemáticos y estadísticos, se identifican patrones complejos en los datos. Esto permite que el ordenador sea capaz de predecir comportamientos en futuras ocasiones.

Estudiando la literatura actual sobre la detección y la clasificación de defectos superficiales, se han extraído los métodos más utilizados en la extracción de características y en la clasificación.

3.1.1. Extracción de características

Se explican a continuación los métodos más utilizados en la extracción de características en los defectos superficiales del acero.

Local binary pattern (LBP)

Local binary pattern (LBP) es un enfoque estadístico simple y muy eficiente para la clasificación de texturas y objetos introducido por Ojala et al. [5]. LBP filtra los píxeles adyacentes según determinadas consideraciones y obtiene un valor binario representativo. La idea básica es resumir la estructura local en una imagen comparando cada píxel con su los de su vecindad. Uno de los atributos más importantes de este operador es la robustez frente a la variación de luminosidad.

Con el LBP básico solo se trabaja con un canal de la imagen. Esta versión funciona en un bloque de 3 x 3 píxeles de una imagen en escala de grises, el píxel de este bloque está medido por su valor de píxel central, multiplicado por potencias de dos y sumado para obtener una etiqueta para el píxel central. Para cada pixel en una celda, se compara con cada uno de sus 8 vecinos, cuando el valor del pixel central sea mayor que el valor del vecino su valor es 0, si por el contrario es menor, el valor será 1.

A continuación, se calcula el histograma de cada celda y se concatenan los de todas ellas, obteniendo así un vector de características. El LBP básico está representado en la Ilustración 10. Años después, Ojala et al. [6] presentó una versión más genérica del operador, en contraste con la versión básica, esta versión no pone limitaciones al tamaño del vecindario o al número de puntos de muestra.

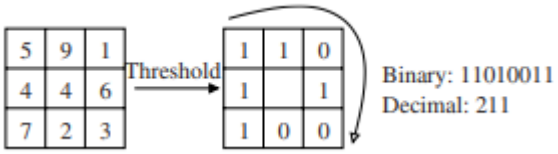


Ilustración 10. LBP básico.
Fuente: [56]

Existe una variante ampliamente utilizada de LBP, esta se conoce como *Uniform Patterns*. En este caso se usa una medida de uniformidad de un patrón, este patrón es el número de transiciones de bit a bit de 0 a 1 cuando el patrón de bits se considera circular. Cuando la medida de uniformidad es menor que 2, se denomina LBP.

En la actualidad, LBP se emplea en diversas investigaciones con algunas modificaciones. El uso de este está justificado por su insensibilidad a la luz, la invariancia en la rotación y la simplicidad en el cálculo. Sin embargo, este enfoque estadístico solo mantiene el patrón del signo para la textura, y no para la magnitud, por lo que Maoxiang Chu et al. [7] proponen un operador de LBP con signo y magnitud. Esos nuevos operadores describen la textura de la región del defecto a partir de los patrones de signo y magnitud, y esas características son insensibles a la transformación afín en escala y rotación.

En el caso del uso del *Uniform Local Binary Pattern* (ULBP), la investigación realizada por M. Xiao et al. [8] muestra la buena precisión obtenida usando este operador combinado con el clasificador Bayes Kernel (BYEC).

Transformación de Hough

La teoría del método de transformación Hough (HT) es que cualquier punto en una imagen binaria podría ser parte de un conjunto de una característica geométrica. Requiere que las características deseadas se especifiquen en alguna forma paramétrica. Este método de extracción de características se usa generalmente para la detección de curvas regulares (líneas, círculos, elipses, etc.), como se observa en la Ilustración 11. El uso de HT es ideal para la detección de rugosidades y precisa en la detección de grietas en las esquinas. [9]

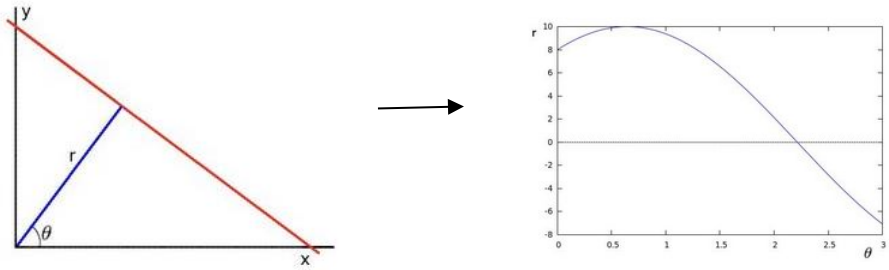


Ilustración 11. Transformada de Hough.
Fuente: https://docs.opencv.org

Luiz et al. [10] emplean HT en la detección y clasificación de las soldaduras (línea diagonal en relación con el borde de la tira), abrazaderas (línea perpendicular en la misma condición) y orificios de identificación (un círculo con radio fijo en el medio de la tira). Los valores de alta precisión (98%) se justifican por las características geométricas bien definidas de cada defecto.

Gray-level co-occurrence matrix

Gray Level Co-occurrence Matrix (GLCM) y los cálculos de las características de textura son técnicas de análisis de imágenes. Dada una imagen compuesta de píxeles, cada uno con una intensidad (un nivel de gris específico), el GLCM es una tabulación de la frecuencia con la que diferentes combinaciones de niveles de gris ocurren simultáneamente en una imagen o en una sección de imagen. Los cálculos de características de textura usan los contenidos del GLCM para dar una medida de la variación en intensidad en el píxel de interés, como se muestra en la Ilustración 12.

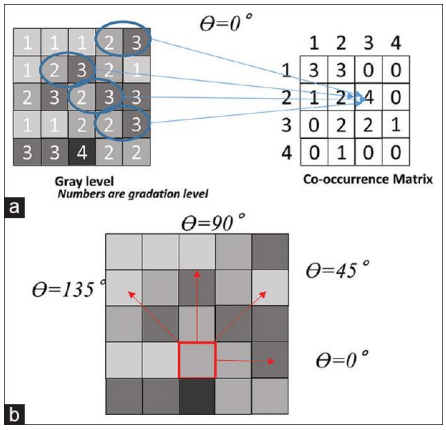


Ilustración 12. Proceso de GLCM.
Fuente: [57]

Gabor

Gabor es un filtro lineal utilizado generalmente para análisis de textura y detección de bordes. El filtro Gabor es una clase especial de filtro de paso de banda, se crea al multiplicar una función de envolvente gaussiana con una oscilación compleja. Con una dirección dada, da una respuesta para las ubicaciones de las imágenes objetivo que tienen estructuras en esta dirección.

Qu et al. [11] emplea una versión optimizada del filtro Gabor para la detección de defectos en tiras de acero, los resultados muestran que el algoritmo tiene baja complejidad, alta eficiencia de detección y la detección de la versión optimizada es 2,3 veces más rápida que el modelo tradicional.

En una comparación entre HOG, Gabor, GLCM, LBP y GLH hecha por Mang Xiao et al. [8] se concluye que la función Gabor es la más adecuada para la representación de texturas y la discriminación de la clasificación de defectos de acero. Por el contrario, la función GLH es la menos adecuada para esto. La razón principal de esto es que GLH no puede capturar textura o forma. Cabe destacar que los métodos HOG, GLCM y LBP también proporcionan un buen rendimiento.

Gray Level Histogram

El histograma muestra la frecuencia de cada nivel de gris. Una imagen tiene 256 niveles de gris distintos, estos valores generan un vector de 256 dimensiones. Este vector se usa de entrada para los clasificadores. Existen muchos estudios sobre el uso de *Gray Level Histogram* (GLH) para la extracción de características, en conclusión, este método no tiene la precisión necesaria para detectar correctamente los defectos superficiales [12] [8] [13].

Histogram of Oriented Gradient

El *Histogram of Oriented Gradient* (HOG) es un descriptor de características utilizado en la detección de objetos, HOG calcula los gradientes de cada píxel en una región de imagen local y construye un histograma de dirección de gradiente, un ejemplo del resultado que se obtiene al aplicar este descriptor se muestra en la Ilustración 13.

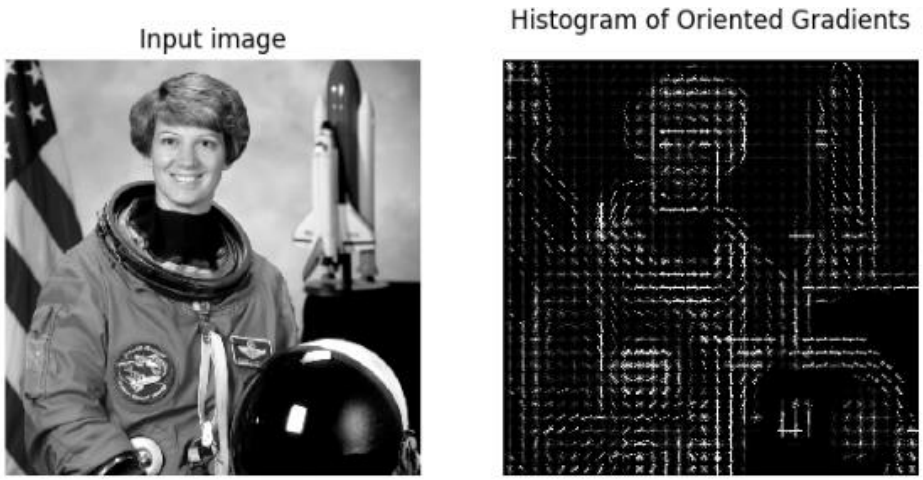


Ilustración 13. Histogram of Oriented Gradient.
Fuente: http://scikit-image.org

Yalin Wang et al. [14] mejoran su eficiencia combinando HOG y GLCM. El conjunto de características HOG describe la información de textura local, mientras que el conjunto de características GLCM, que se usa para extraer características de defectos locales, pudiendo capturar información de textura global. En este artículo se presentó un algoritmo óptimo de fusión de múltiples características (OMFF-RF) que fusiona los conjuntos de características HOG y GLCM para distinguir 5 tipos de defectos distribuidos en superficies de acero adquiridos en una línea real de producción de acero. Los resultados de los experimentos descritos mostraron que el algoritmo propuesto tiene un gran potencial para el reconocimiento de defectos distribuidos y que su tiempo de ejecución de procesamiento de imágenes es relativamente corto.

Principal Component Analysis

El análisis de componentes principales (PCA) convierte un conjunto de píxeles posiblemente correlacionados en un conjunto más pequeño de píxeles no correlacionados linealmente utilizando una transformación ortogonal.

En este artículo, Zhou et al. [15], seleccionan los ángulos de iluminación efectivos según la imagen y el vector de carga correspondiente obtenido por PCA. Se apilan y despliegan imágenes multivariadas con ángulos de iluminación efectivos, a partir de las cuales se pueden obtener las puntuaciones de los componentes principales de las imágenes de prueba.

Scale Invariant Feature Transform

Con *Scale Invariant Feature Transform* (SIFT), el contenido de la imagen se transforma en coordenadas de funciones locales que son invariables para la traslación, la rotación, la escala y otros parámetros de imágenes. SIFT se usa generalmente para la clasificación de objetos y el reconocimiento facial. La combinación de este método con los clasificadores BOF y SVM realizan tareas de clasificación con éxito en gestos manuales, imágenes naturales o imágenes de vehículos [16] [17].

3.1.2. Métodos de clasificación

Una vez extraídas las características más relevantes de las imágenes, se introducen en clasificadores para identificar su clase. A continuación, se muestran los clasificadores más empleados para la clasificación de estos defectos.

Support Vector Machines

Support Vector Machines (SVM) son un conjunto de métodos de aprendizaje supervisado aplicables a problemas de clasificación y regresión. Un clasificador SVM crea un hiperplano de margen máximo que se encuentra en un espacio de entrada transformado y divide las clases de ejemplo, al tiempo que maximiza la distancia hasta los ejemplos más cercanos [18].

Una de las razones del éxito de SVM es su capacidad de aprender correctamente con un número muy pequeño de parámetros, además de por su robustez contra valores atípicos y su eficiencia computacional en comparación con otros métodos [19].

En el campo de la clasificación de defectos del acero, se han realizado diversas modificaciones o combinaciones de este método obteniendo resultados precisos, siendo SVM uno de los más empleados en este ámbito [8] [20] [7] [21].

Naive Bayes

Naive Bayes (NB) es una técnica de clasificación basada en el teorema de Bayes donde se supone cierta independencia entre los predictores. En términos simples, NB supone que la presencia de una característica particular en una clase no está relacionada con la presencia de ninguna otra característica.

Huang et al. [22] proponen una combinación de NB con LBP obteniendo una detección de defectos muy rápida, siendo este factor crucial en un entorno real de producción industrial.

K-nearest neighbor

El algoritmo *K-nearest neighbor* (KNN) es uno de los algoritmos de clasificación más simples y uno de los algoritmos de aprendizaje más utilizados. Su propósito es utilizar una base de datos en la que los puntos se separan en varias clases para así predecir la clasificación de un nuevo punto.

En el trabajo realizado por Zoheir et al. [23] se emplean los clasificadores SVM y KNN en un proyecto de inspección de bandas de acero laminadas en caliente. En los tipos de defectos con los que ellos trabajan obtienen que KNN es más relevante en la identificación en términos de precisión y costo computacional en comparación con SVM.

Random Forest

El algoritmo *Random Forest* (RF) es un algoritmo de clasificación supervisado. Consiste en una combinación de clasificadores donde cada clasificador contribuye con un solo voto en la asignación de la clase más frecuente la instancia de entrada. El hecho de que sea una combinación de muchos clasificadores confiere a la RF algunas características especiales que la hacen sustancialmente diferente a los árboles de clasificación tradicionales. Un ejemplo de este clasificador se muestra en la Ilustración 14.

Wang et al. [14] combina los métodos de extracción de características HOG y GLCM a través de un factor de fusión de múltiples funciones, que cambia el número de árboles de decisión que corresponden a cada conjunto de características en el algoritmo de RF. Esta propuesta se verificó en 5 tipos de defectos recogidos en una línea de producción de acero, obteniendo una precisión de reconocimiento del 91%, superando así otros métodos convencionales como SVM.

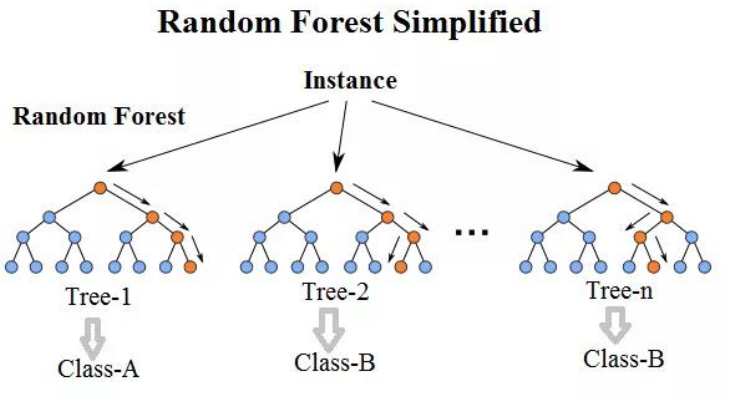


Ilustración 14. Clasificador Random Forest.
Fuente: https://sefiks.com

K-means

K-means es un tipo de clasificador de aprendizaje no supervisado utilizado cuando se tienen datos con clase desconocida. El objetivo de este algoritmo es encontrar k grupos dentro de esos datos. *K-means* trabaja de forma iterativa asignando un grupo a cada punto basándose en las características de estos, juntando así los datos por similitud.

Empleando este método junto a GGCM, Cui et al. [24] logran identificar soldaduras sin defecto o grietas en tuberías de acero. Riaz et al. [25] en cambio, combinan K-means con un filtro Gaussiano en el pre-proceso para distinguir agujeros y grietas, obteniendo buenos resultados en todos los niveles de resolución.

3.1.3. Resultados obtenidos empleando Machine Learning

Song et al. [26] proponen un método para abordar la influencia de la iluminación, los cambios de material, el ruido y la similitud de los defectos entre clases. Para ello, proponen una modificación de *Local Binary Patterns* llamado AECLBP. En esta propuesta se emplea una ventana de evaluación que abarca los píxeles vecinos y modifica el umbral del esquema para reducir la interferencia de ruido. Los resultados obtenidos demuestran que reduce los problemas mencionados.

Song et al. [27] evalúan la *Scattering Convolution Network* (SCN) propuesta por Bruna et al. [28]. En la SCN se emplean múltiples capas para mantener la información más importante. Este método es comparado con otros de extracción de características como LBP y con métodos de clasificación como NNC y SVM, obteniendo los mejores resultados en la combinación de SCN y SVM.

Pan et al. [29] proponen un nuevo método de clasificación de defectos superficiales llamado *Local SIFT Pattern* (LSP) basado en Dense SIFT+ *Improved Fisher Vector* (IFV). Ellos demuestran que su método es más eficaz que Dense SIFT+IFV, lo que supone una mejora en el método. Cómo el tiempo de clasificación es muy importante en los procesos industriales, analizan también el tiempo computacional obtenido.

En la comparación realizada por Xiao et al. [8] muestran que Gabor es un descriptor adecuado para identificar defectos en superficies de acero. En cambio, GLH es el que peor resultados ofrece en este caso, porque no puede capturar texturas o formas. HOG, GLCM y LBP ofrecen resultados buenos también.

3.2. Deep Learning

Dentro de la rama del aprendizaje automático, hay una gran cantidad de enfoques que dependen de la problemática a solucionar. En los últimos años, los métodos de *Deep Learning* se han extendido en el uso de extracción y clasificación de objetos.

La idea del *Deep Learning* surge de simular el funcionamiento de los cerebros biológicos, de forma que consta de múltiples neuronas interconectadas capaces de ejecutar operaciones complejas como la extracción de características. Esta estructura es capaz de aprender por sí misma sin interacción humana, obteniendo una mejora significativa en tareas de percepción computacional.

Las neuronas artificiales simulan el comportamiento de las neuronas biológicas. Las neuronas artificiales pueden recibir varios estímulos como entradas, pero únicamente tienen una salida. La salida conecta con entradas a otras neuronas a través de unos enlaces ponderados, esta es calculada empleando funciones de activación, como se muestra en la Ilustración 15. Se establecen unos pesos en los enlaces que conectan las neuronas, representando la relevancia de esa entrada a la neurona. Son precisamente estos pesos los que se han de ajustar en la tarea de aprendizaje mediante redes neuronales.

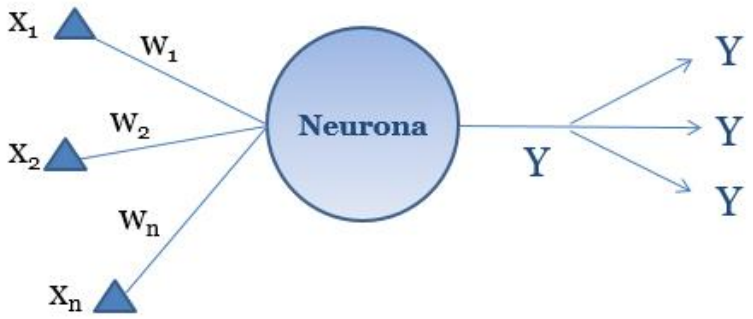


Ilustración 15. Estructura de una neurona artificial.
Fuente: [58]

Dado un conjunto de datos de entrenamiento, formado por una serie de entradas al sistema junto con las salidas conocidas correspondientes, el objetivo del aprendizaje consiste en determinar los pesos W_i que permiten a la red relacionar ambas series de datos entre sí de la mejor manera posible, Se espera así que, ante nuevas series de datos de entrada, el sistema sea capaz de inferir las salidas correctas. Estos pesos en un principio son desconocidos, por lo que se establecerán unos aleatorios. En cada iteración, la salida que se obtiene al aplicar dichos pesos sobre una entrada será diferente de la salida conocida o esperada. Esa

diferencia es el error que se va a utilizar para ir ajustando los pesos siguiendo la regla de aprendizaje del perceptrón. [30]

Las **redes neuronales** son una de las técnicas de aprendizaje automático más utilizadas, ya que se van adaptando a la información de entrada y van aprendiendo por sí mismas. El uso de estas ganó importancia cuando se aumentó el número de capas, junto al poder computacional de las tarjetas gráficas, creándose así el término de Deep Learning.

La primera red neuronal y la más simple creada es el perceptrón, que contaba con una sola neurona. Con posterioridad, se desarrollaron las redes neuronales multicapa, que son redes unidireccionales de alimentación hacia adelante, propagándose así la señal de entrada a lo largo de todas las capas hacia la salida de la red. Existen también las redes neuronales recurrentes, que tratan de emular las características asociativas de la memoria humana, alimentando la entrada de la red con las salidas que se van generando. Un ejemplo de un modelo básico de red neuronal se muestra en la Ilustración 16.

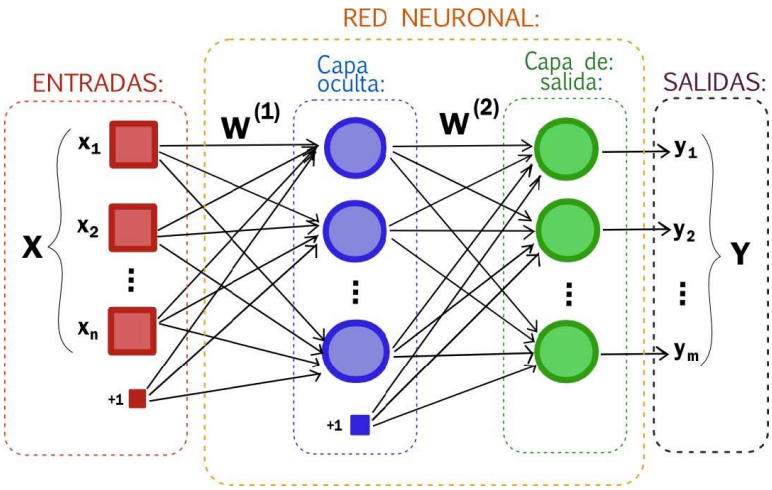


Ilustración 16. Modelo de red neuronal básica.
Fuente: http://ceres.ugr.es

La principal diferencia entre una red neuronal clásica y las utilizadas en el contexto de *Deep Learning*, es que en este último caso las redes están formadas por varias capas ocultas intermedias como se aprecia en la Ilustración 17. El desarrollo de este tipo de redes ha sido posible en los últimos años gracias al importante avance de las técnicas computacionales, y el desarrollo del hardware de soporte.

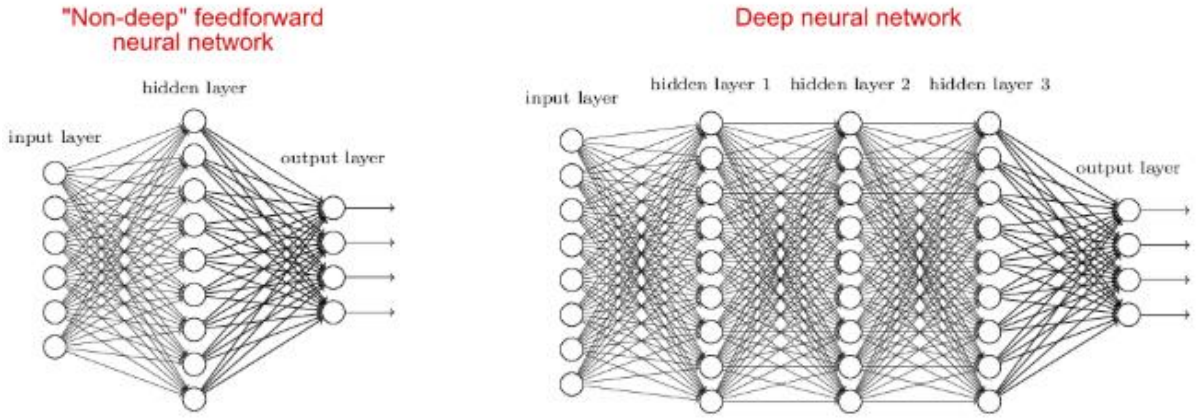


Ilustración 17. Comparación entre red neuronal y deep learning.
Fuente: http://neuralnetworksanddeeplearning.com.

El *Deep Learning* es bastante apropiado para aplicaciones en las que no se dispone de un modelo identificable que pueda ser programado, pero se dispone de un conjunto representativo de ejemplos de entrada. Asimismo, es robusto frente al ruido o la deformación de elementos y es fácilmente paralelizable. Es por eso por lo que se emplea bastante en problemas de clasificación y reconocimiento de patrones de voz, imágenes, señales, etc.

Dentro del contexto del procesamiento de imagen, la idea de usar redes neuronales proviene de la capacidad de adaptación frente a las situaciones amplias y variables que ocurren en el proceso de captura, como por ejemplo los cambios en la iluminación. Estos cambios dificultan la extracción de características. Por esta razón, es importante recalcar la importancia de la iluminación y la calidad de la captura, ya que no solo se necesitan una amplia colección de datos, si no que estos tienen que tener la calidad necesaria para obtener un modelo correcto.

Como se ha mencionado uno de los principales requerimientos de este método, es la necesidad de un gran volumen de datos para poder entrenar la red. Centrando el método en la detección y clasificación de defectos superficiales en el acero, obtener estos datos generalmente es difícil ya que los defectos ocurren con poca frecuencia. Otro factor muy importante es definir bien la ubicación del defecto en la imagen, ya que la diferencia física entre las clases de defectos es pequeña. El uso de redes neuronales para detectar la posición del efecto y clasificarlo es un desafío, ya que generalmente se utilizan únicamente para la clasificación.

Existen diferentes tipos de redes neuronales que se pueden emplear para solucionar este problema.

3.2.1. Redes Neuronales Convolucionales

Las redes neuronales convolucionales (CNN) son una arquitectura de red neuronal usada en el *Deep Learning*. Estas son capaces de aprender de imágenes, eliminando así el posible error que se genera en la extracción manual de características.

Las CNN están compuestas por multitud de capas que son capaces de aprender a detectar diferentes características de las imágenes. En cada capa se aplican una serie de filtros que generan como salida una imagen convolucionada siendo esta la entrada de la siguiente capa. Los filtros que se aplican extraen primero las características más generales, y a medida que se van a aplicando más, se van extrayendo características más concretas, logrando un reconocimiento de características cada vez más grande.

Este tipo de arquitectura necesita de un gran número de imágenes de entrenamiento, es por eso necesario el uso de una GPU capaz de procesar el entrenamiento del modelo.

En cuanto a su estructura, las CNN tienen 5 tipos distintos de capas:

- **Capa convolucional.** La capa convolucional es el núcleo de una CNN. Esta recibe la imagen como entrada, sobre la que se aplica un filtro de convolución obteniendo así un mapa de características. Este proceso reduce la imagen de entrada al aplicar una operación de convolución dimensional para la imagen de entrada. La activación de las características de salida se obtiene al sumar una o más respuestas convolucionales que pasan a través de una función de activación de píxel. Cada capa convolucional tiene muchos núcleos de filtro que generan algunos mapas de características de salida diferentes. Cada filtro puede extraer diferentes representaciones de características.
- **Capa de función rectificadora lineal uniforme (ReLU).** El objetivo de esta capa es introducir la no linealidad. Al mismo tiempo, ReLU tiene buenas propiedades de dispersión, ya que tiene un valor de activación cero, así como también limita la saturación de la salida durante el proceso de entrenamiento [31]. También se pueden usar otras funciones como tanh o sigmoid. Sin embargo, ReLU generalmente logra mejores resultados.
- **Capa de reducción o *pooling*.** La capa de agrupación se usa para realizar una representación de submuestreo. Normalmente se usa después de las capas convolucionales. Esta capa reduce los tamaños de vector de características y los parámetros de la CNN. Disminuye el tiempo de entrenamiento y los requisitos de memoria, y controla el sobreajuste. Las operaciones de agrupación más comunes son las de agrupación máxima y media.

- **Capa de normalización**. Esta capa es opcional, la respuesta de salida es localmente normalizada utilizando una función de promedio ponderado basado en la distancia.
- **Capa de clasificación totalmente conectada**. En esta capa se clasifican las características extraídas en las capas anteriores. Se compone por un número de neuronas igual al número de clases existentes. Se denomina totalmente conectada porque cada una de estas neuronas está conectada a todos los elementos de la capa anterior. Finalmente, se aplica generalmente la regresión *softmax* para clasificar las muestras.

Una alternativa al entrenamiento desde cero es emplear redes pre-entrenadas, que consiste en cambiar la última capa de la red con las clases que se desea clasificar. El modelo pre-entrenado es un modelo cuya arquitectura ha sido diseñada para solucionar un problema similar, y ha sido entrenado previamente por alguien para su problema en particular. Esto soluciona el problema de tener poca cantidad de imágenes para el entrenamiento evitando así un sobreajuste.

Ren et al. [32] proponen un método que testean en una base de datos pública. Su método denominado DeCAF está compuesto por una CNN que posee 5 capas convolucionales, tres capas de reducción y tres capas de clasificación totalmente conectadas. El método propuesto obtiene mejores resultados que los métodos tradicionales en comparación con las precisiones empleando los métodos de extracción de características LBP y GLCM combinados con los métodos de clasificación SVM y KNN.

Masci et al. [33] emplean también una CNN para clasificar defectos superficiales en el acero. Estos son capaces de clasificar 7 defectos recolectados en una línea de producción real con un error del 7%. Comparando su método con clasificadores comúnmente usados como es SVM obtienen unos resultados mejores y de una manera más optimizada y rápida.

Otro caso en el que se emplean CNNs es en el presentado por Soukup et al. [34], estos emplean imágenes de superficies metálicas obtenidas con una iluminación *dark-field* de diferentes colores. Este tipo de iluminación permite resaltar las cavidades en el material.

Zhou et al. [31] prueban la eficacia de emplear las CNN para la detección y clasificación de defectos en chapas de acero laminado en caliente. Aplicando pre-procesos y con un conjunto de datos medianamente pequeño de los defectos más comunes, obtienen una precisión del 99% configurando el número de capas y el tamaño de la imagen de entrada.

El uso de las redes neuronales está justificado también por tanto en el momento de la extracción de características. La precisión de la ubicación del defecto mejora cuando se

realiza automáticamente, de modo que de esta forma se minimizan los errores y se elimina el ruido de la muestra de entrenamiento.

3.3. Frameworks para Deep Learning

Como se ha dicho anteriormente, el uso del *Deep Learning* ha aumentado a medida que han aparecido tecnologías capaces de soportar la carga computacional que conlleva entrenar este tipo de redes. Para ello se han desarrollado *frameworks* con una programación de alto nivel que permiten diseñar, entrenar y validar las redes neuronales profundas.

A continuación, se mencionan los más utilizados junto a una pequeña descripción.

3.3.1. Caffe2

Caffe2 [35] es un *framework* cuya API está basada en Python. Sirve para diseñar de manera sencilla todo tipo de modelos de *Deep Learning* y ejecutarlos usando una capa de acceso a datos en C++ y CUDA. Este *framework* soporta ejecución multi-GPU.

3.3.2. Cognitive Toolkit

Cognitive Toolkit (CNTK) [36] es un kit de herramientas de código abierto desarrollado por Microsoft. Esta herramienta muestra las redes neuronales a través de un gráfico dirigido como una serie de pasos computacionales, donde los nodos representan valores de entrada o parámetros de red u operaciones de matriz sobre las entradas.

3.3.3. MATLAB

MATLAB [37] cuenta con una serie de herramientas y funciones para trabajar con grandes conjuntos de datos. Posee kits de herramientas para trabajar con redes neuronales, visión por computador, *Machine Learning*, etc. MATLAB permite crear y visualizar modelos y generar código CUDA [38] para aplicaciones de visión y de *Deep Learning* automáticamente desde el código de MATLAB. CUDA es una arquitectura de cálculo paralelo de NVIDIA que aprovecha la potencia de la GPU (unidad de procesamiento gráfico) para proporcionar un incremento del rendimiento del sistema.

3.3.4. MXNet

MXNet [39] es un *framework* de código abierto desarrollado por Apache empleado para entrenar e implementar redes neuronales profundas. Es escalable, permite generar modelos de una forma rápida y soporta diferentes lenguajes de programación.

3.3.5. NVIDIA Caffe

Caffe [40] es otro *framework* de código abierto desarrollado por la UC Berkeley. Está desarrollado en C++ y su interfaz está en Python.

3.3.6. PyTorch

PyTorch [41] es un paquete de Python para llevar acabo experimento de *Deep Learning* de una manera rápida y flexible. Este contiene el paquete Tensor, similar a numpy, con una fuerte aceleración GPU.

3.3.7. TensorFlow

TensorFlow [42] es una librería desarrollada por Google para computación numérica. Se emplea para el entrenamiento de redes neuronales, para descifrar patrones y para *Machine Learning*. Ofrece una arquitectura flexible, empleándose así con mayor frecuencia en investigación.

3.3.8. Chainer

Chainer [38] es un *framework* basado en Python que aporta flexibilidad. Posee un API de alto nivel orientado a objetos para construir y entrenar redes neuronales. También soporta CUDA.

Cabe destacar que los *frameworks* como CNTK, Caffe2, Chainer, MXNet y PyTorch permiten a los desarrolladores mover modelos entre estos ya que emplean una representación de modelo compartido llamado *Open Neural Network Exchange* (ONNX), permitiendo así la interoperabilidad y la optimización.

3.4. Sistemas de almacenamiento de los datos de entrenamiento

Para entrenar una red neuronal y obtener un modelo consistente se necesita de un sistema de almacenamiento que permita albergar grandes cantidades de información, sobre todo a la hora de la implementación de un sistema de visión en fábrica, debido a la cantidad de datos que se generarán.

En los últimos años, han surgido un gran número de arquitecturas basadas en el modelo *Big Data* para el análisis y almacenamiento de cadenas de datos. Dichas arquitecturas están usualmente desarrolladas sobre infraestructuras *cloud* y herramientas caracterizadas por su flexibilidad y escalabilidad. El modelo de negocio usual para tecnologías *Big Data* está basado en la creación de infraestructuras de tipo *cluster* con gran capacidad de procesamiento tomando como referencia el paradigma *cloud* IaaS (que ofrece infraestructura como servicio) de cara a ofrecer *Big Data* como SaaS (Software como servicio).

Su funcionamiento básico se basa en la ejecución de cinco procesos: captura de datos; almacenamiento; transformación; análisis y visualización, aunque en este proyecto no se abordará el proceso de la transformación de los datos. La Ilustración 18 muestra las diferentes capas incluidas en el procesamiento *Big Data* [43].

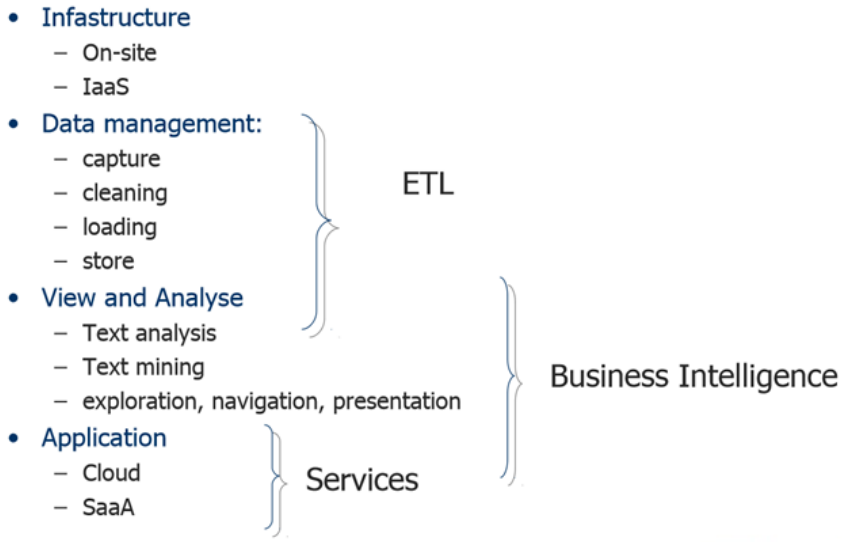


Ilustración 18. Capas de las tecnologías Big Data.
Fuente: https://www.spagobi.org/

Las bases de datos relacionales han sido empleadas tradicionalmente para el almacenamiento de datos. Sin embargo, con el auge del *Big Data* y debido a la gran cantidad de datos que se generan, estas se quedan obsoletas dando lugar a las bases de datos no relacionales o NoSQL.

Las bases de datos NoSQL son el tipo de bases de datos más utilizadas para el almacenamiento *Big Data*. Utilizan modelos de almacenamiento diferentes a los relacionales, entre los que se pueden mencionar los siguientes:

- Almacenes clave-valor. Permiten el almacenamiento de datos sin esquema, los objetos de datos pueden ser desestructurados o estructurados y su acceso se gestiona mediante el uso de una simple clave. Como no hay un esquema director, cada objeto puede tener una estructura diferente. La base de datos más empleada con esta estructura es Apache Cassandra [44].
- Almacenes de columnas. Se basan en el almacenamiento de tablas de datos como secciones de columnas de datos en lugar de como filas de datos (al contrario que las bases de datos relacionales). Estas bases de datos están dispersas, distribuidas y están indexadas por una clave triple: clave de fila, clave de columna y *timestamp*. Los

datos son accedidos mediante columnas y los valores se representan como *strings* ininterrumpidos de datos. El ejemplo más utilizado es HBase [45].

- Bases de datos documentales. Aunque se trata de bases de datos estructuradas, no hay un esquema común definido por lo que las estructuras pueden ser muy diversas definiéndose como semi-estructuradas. El acceso se maneja mediante *querys* dependientes de la codificación que use la base de datos, las codificaciones usuales suelen ser XML o JSON. Un ejemplo conocido es MongoDB [46].
- Bases de Datos de Grafos. Almacenan los datos en estructuras de grafos permitiendo el almacenamiento de datos altamente asociados. Usualmente ofrecen una interfaz propia para el acceso a datos. Neo4J [47] es una de las más utilizadas.

En cuanto a valorar la idoneidad de los tipos de bases de datos NoSQL, la Ilustración 19 muestra la mejor opción en función de la escalabilidad y la complejidad de los datos [43].

Finalmente, las bases de datos NewSQL ofrecen formas modernas de bases de datos relacionales que tienen como objetivo ofrecer una escalabilidad comparable a las bases de datos NoSQL pero manteniendo garantías transaccionales ofrecidas por las bases de datos relacionales.

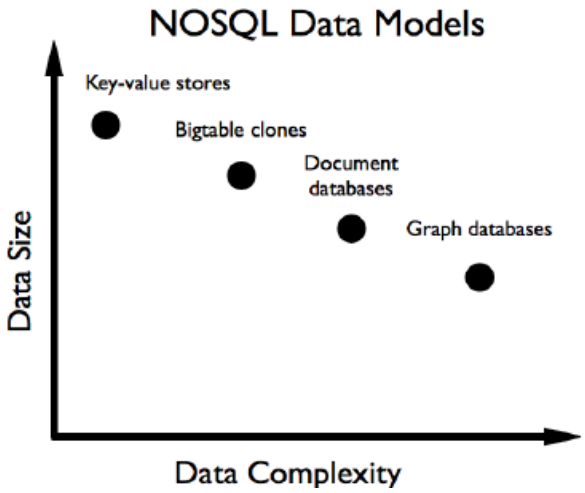


Ilustración 19. Bases de datos NoSQL en función de la complejidad y el volumen de datos. Fuente: [43]

A continuación, se muestra un listado de las bases de datos NoSQL más empleadas mencionadas anteriormente.

3.4.1. Apache Cassandra

Apache Cassandra[1] fue creado para el almacenamiento masivo de la información. Ofrece gran escalabilidad (lineal) y disponibilidad de los datos. Estos datos se almacenan en tablas con una arquitectura clave-valor. Este sistema se desarrolló inicialmente por Facebook, aunque ahora pertenece a Apache.

El sistema dispone de múltiples nodos distribuidos que se comunicas por protocolo P2P. Posee una alta redundancia quedando así el sistema protegido frente a fallos y asegurando la respuesta. En cuanto a la estructura, no es necesario un nodo maestro, ofreciendo una baja latencia. El lenguaje para comunicarse con el sistema es *Cassandra Query Language* (CQL), propio de Cassandra, que simula el lenguaje SQL tradicional mejorándolo.

3.4.2. HBase

HBase[2] está creado para soportar accesos de lectura y escritura en tiempo real. Es una base de datos abierta, distribuida, versionada y no relacional, modelada según el *Bigtable* de Google, que consiste en un sistema de almacenamiento distribuido para datos estructurados de Chang et al. [48].

3.4.3. Mongo DB

MongoDB[3] es una base de datos no relacional de código abierto orientada a almacenar documentos. El desarrollo de MongoDB empezó en octubre de 2007 por la compañía de software 10gen. Esta base de datos es muy ágil y permite a los esquemas cambiar rápidamente cuando las aplicaciones evolucionan. Esta base de datos ha sido creada para brindar escalabilidad, rendimiento y gran disponibilidad.

3.4.4. Neo4J

Sistema de gestión de bases de datos que está basada en grafos. Neo4J[4] está desarrollado por *Neo Technology* en Java y fue lanzado en 2010. Es una base de datos transaccional compatible con la regla ACID (Atomicity, Consistency, Isolation y Durability) con almacenamiento y procesamiento de grafos nativos.

Esta base de datos posee además una comunidad muy amplia de desarrollo, además de una alta disponibilidad y escalado en *cluster*. Dos de las propiedades más importantes de esta

[1] Página web del proyecto Apache Cassandra http://cassandra.apache.org/
[2] Página web del proyecto Apache HBase Project https://hbase.apache.org/
[3] Página web del sistema de base de datos NoSQL MongoDB https://www.mongodb.com/
[4] Página web del software libre para bases de datos de grafos Neo4j https://neo4j.com/

tecnología son el almacenamiento de grafos, el cual emplea un almacenamiento nativo diseñado específicamente para almacenar y administrar grafos, y el motor de procesamiento de grafos.

3.5. Visual Analytics

Uno de los factores críticos que afectan al proceso de toma de decisiones en la era de la información es cómo encontrar datos relevantes y cómo obtener información significativa a partir de estos datos. Para hacer frente a este problema surgió una nueva disciplina conocida como *Visual Analytics* (VA).

VA tiene como objetivo el soporte a la toma de decisiones basadas en grandes volúmenes de datos cuantitativos introduciendo al analista en el flujo y permitiendo así la combinación de técnicas de inteligencia artificial con la inteligencia real de los profesionales. Se define como la ciencia de combinar las visualizaciones interactivas con algoritmos de análisis para apoyar la exploración, el análisis y la presentación de grandes conjuntos de datos.

Estas son una aproximación multidisciplinar que se basa en diferentes áreas de investigación como la visualización, minería de datos, gestión de datos o estadística.

La Tabla 3 muestra una taxonomía de dinámicas interactivas que pueden favorecerse de la implementación de las analíticas visuales. Estas categorías incorporan las tareas críticas que permiten un análisis visual interactivo, incluyendo la creación de una forma de visualización, las consultas interactivas, la coordinación multipantalla, los historiales o la colaboración. Posada et al. [43] sitúan VA en su rol crucial para la Industria 4.0, presentando una visión de conjunto general a la vez que indican futuras líneas de investigación.

Tabla 3 Taxonomía de dinámicas interactivas para las analíticas visuales.

Especificación de los datos y el formato de visualización	• Visualización de datos mediante la utilización de codificaciones visuales • Filtrado de datos para la selección de elementos relevantes • Clasificación de los elementos para detectar patrones • Obtención de modelos a partir de los datos originales
Manipulación de la vista	• Selección de elementos para resaltar, filtrar o manipular • Navegación para analizar patrones de alto nivel y detalles de bajo nivel • Coordinación de vistas para una exploración multidimensional • Organización de múltiples ventanas y entornos de trabajo
Procesamiento y procedencia	• Almacenamiento de análisis históricos para su revisión y compartición • Anotación de patrones • Compartición de vistas y anotaciones para permitir la colaboración • Guiado de usuarios mediante el análisis de históricos

Actualmente se genera en todo el mundo grandes cantidades de información, que procede de los propios individuos, empresas y otras organizaciones. Esta información se genera de manera continua, desde una enorme variedad de ubicaciones y dispositivos. El estudio de dicha información ayuda en la toma de decisiones y apoya la inteligencia de negocio. Ahora bien, es necesario procesar y analizar esta gran cantidad de datos para convertirla en información útil fácilmente accesible.

Los datos a procesar en una empresa pueden provenir de fuentes muy diversas (redes sociales, datos de consumo, clientes u empleados), pudiendo estar o no estructurados.

Los datos pueden categorizarse en diferentes grupos para seleccionar las técnicas gráficas a utilizar. En primer lugar, dichos datos pueden dividirse en dos grupos según su naturaleza:

- Datos continuos que pueden tomar valores infinitos.
- Datos discretos, que especifican ítems independientes especificados en categorías cualitativas.

También se tendrá en cuenta la forma de medir estos datos (sobre una escala nominal, una escala ordinal, un intervalo o ratio).

Una de las técnicas más utilizadas para explotar la información es la de VA donde la representación de los datos se basa en un razonamiento facilitado por interfaces visuales interactivas. Cada elemento visual tendrá sus propias fortalezas, debilidades, limitaciones en la representación. El tipo de gráficos y técnicas de visualización concretas dependerán en general de las características de los datos analizados.

Los gráficos implementados con técnicas de analíticas visuales facilitan el acceso, comprensión y comunicación de los datos. Además, un gráfico permite estructurar la información y facilita su exploración de forma rápida e intuitiva.

Entre las principales tendencias actuales en el análisis visual, pueden mencionarse las siguientes metodologías:

- Utilización de diversos gráficos simultáneos. Se puede analizar los gráficos desde diferentes perspectivas o a partir de subconjuntos de datos. En la Ilustración 20 se muestra el *dashboard* ganador del concurso 2012 *Perceptual Edge Dashboard Design Competition*, en el que se representa un conjunto de datos sobre rendimiento estudiantil empleando distintas visualizaciones. Este tablero permite monitorear el desempeño y el comportamiento de los alumnos evaluándolos en diferentes campos como la asistencia a clase, las faltas o el histórico de notas.

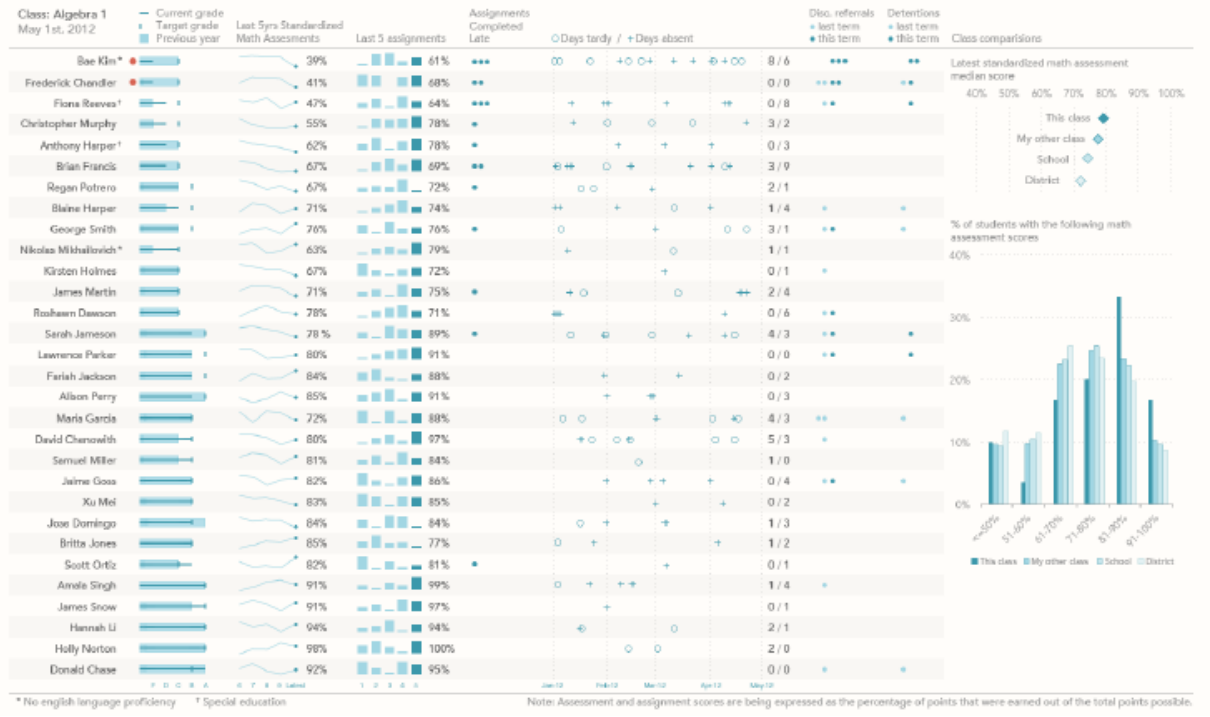


Ilustración 20. 2012 Perceptual Edge Dashboard Design Competition.
Fuente: https://www.perceptualedge.com/blog/?p=1374

- Visualización geo-espacial. Este tipo de técnica es útil cuando la información a representar tiene asociada localizaciones geográficas como por ejemplo códigos postales. En la Ilustración 21 se muestra como ejemplo el mapa de la ciudad de Barcelona a través del cual se puede comparar la edad de cada una de sus edificaciones y descubrir datos históricos y culturales.

Ilustración 21. Ejemplo de información geoposicionada. Mapa interactivo de Barcelona.
Fuente: http://bigtimebcn.300000kms.net/

- Animación visual temporal. Son representaciones de datos que forman parte de series temporales, y por tanto tienen una evolución en el tiempo. Esta evolución se suele representar generalmente empleando algún tipo de animación.
- Jerarquías o niveles. Esta técnica se utiliza cuando el volumen de datos a visualizar es muy grande y se puede categorizar en diferentes niveles o jerarquías. Un ejemplo de este tipo de gráficos es el *TreeMap* donde los datos se representan por jerarquías en rectángulos. Un ejemplo de este tipo de gráfico es el que se muestra en la Ilustración 22, donde se detalla la prevalencia del Virus de la Inmunodeficiencia Humana (VIH) en todo el mundo. La primera parte representa el estado de las personas que viven con el VIH, la segunda parte en cambio muestra las regiones geográficas donde viven las personas enfermas.

Ilustración 22. Ejemplo de visualización TreeMap. Informe anual sobre el SIDA de UNAIDS. Fuente: http://www.unaids.org/

- Representación de relaciones. Este tipo de gráficos representa relaciones entre entidades o nodos en lugar de valores cuantitativos. La interacción con estos gráficos permite navegar por los diferentes nodos y estudiar sus relaciones en detalle. En la Ilustración 23 se muestra un ejemplo donde se visualizan los miembros del Instituto de Crédito Oficial (ICO) y sus relaciones.

Ilustración 23. Ejemplo de visualización de conexiones. Miembros de ICO y sus relaciones. Fuente: http://quienmanda.es/organizations/ico-1

- Gráficos 3D. Muchos de los gráficos realizados actualmente se realizan en tres dimensiones. Muchas veces su utilización no sólo no es necesaria, sino que además eclipsa el contenido del gráfico. Como se observa en la Ilustración 24, el ángulo no es la única forma en que un gráfico circular representa datos. El área también codifica los mismos datos que el ángulo, y haciendo un gráfico circular 3D se distorsiona el área.

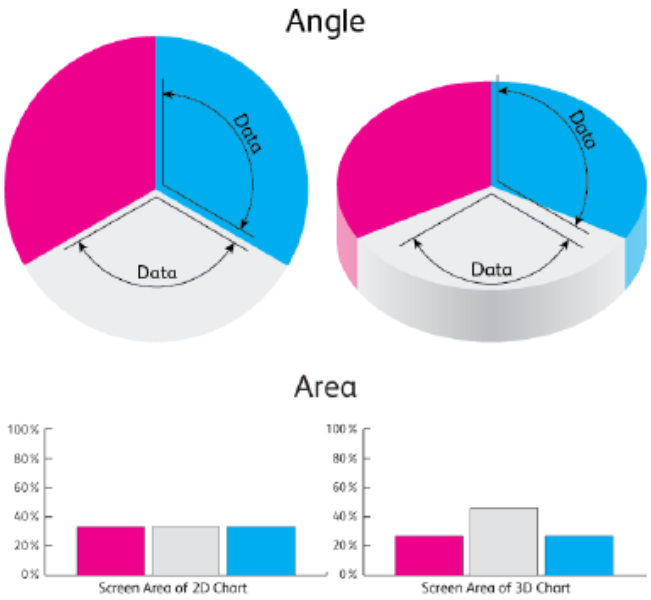


Ilustración 24. Distorsión en la dimensionalidad de los atributos gráficos. Fuente: https://visual.ly/blog/2ds-company-3ds-a-crowd/

Uno de los mayores problemas de visualización de los datos reside en que algunos gráficos no expresan las ideas claramente y conllevan a una interpretación y decisión erróneas. Esto es debido principalmente a la aplicación de técnicas de visualización erróneas y a la representación de datos innecesarios que recargan el gráfico innecesariamente. Aunque es difícil ver algún dato que no haya sido visualizado en un gráfico anteriormente, éstos no siempre se representan correctamente.

En la actualidad, se han explorado otro tipo de alternativas relacionadas con los gráficos de flujo de datos, en los que el proceso de visualización se deconstruye en un conjunto de operaciones relacionadas con la importación, transformación y diseño de los datos. Otros sistemas están basados en gramáticas formales para la construcción de sistemas de visualización. Se trata de lenguajes de alto nivel que describen la forma en la que los datos se mapean a las características visuales. La combinación de diferentes programas permite construir formatos de visualización complejos y customizados.

Otro de los aspectos importantes es el filtrado de los datos, ya que rara vez se visualizan todos los datos, sino que se generan un conjunto de visualizaciones de datos seleccionados. Existe un conjunto extenso de técnicas de interacción para limitar el número de elementos en una pantalla.

Una opción es la utilización de un conjunto de controles auxiliares (*dynamic query widgets*) para controlar la visibilidad de los objetos, de modo que la elección del control adecuado depende del tipo de datos.

3.6. Conclusiones del estado del arte

Como conclusión, se creará la base de datos extrayendo la información de los defectos empleando técnicas de *Deep Learning* ya que se obtienen mejores resultados que con las técnicas tradicionales. Estos datos se albergarán en una arquitectura de almacenamiento que soporte imágenes y que sea capaz de adaptarse a las necesidades de la producción. Para darle sentido a los datos y extraer conclusiones de lo obtenido, son necesarias las visualizaciones. Estas deben ser sencillas, sin animaciones o elementos estéticos que distraigan la atención del lector, y deben ser claras sin dar lugar a dudas o engaños sobre los resultados.

4. Objetivos y metodología de trabajo

En este capítulo se enuncian los objetivos generales y específicos del trabajo y la metodología planteada para alcanzarlos.

4.1. Objetivo general

El objetivo general de este trabajo es **realizar un estudio de arquitecturas de extracción y explotación de datos mediante técnicas de *Computer Vision* y *Deep Learning* para la toma ágil de decisiones relativas a defectos superficiales en procesos continuos de laminación de acero**.

Para ello, se estudiará el control de calidad en entornos de producción industrial y se verán las soluciones actuales de automatización existentes. Se propondrán técnicas para automatizar de una manera fiable la tarea de la detección visual de los defectos superficiales aumentando así la velocidad y la precisión.

En cuanto al almacenamiento de los datos, se investigará sobre los diferentes tipos de bases de datos existentes para albergar las capturas de los defectos y se implementará un prototipo que pueda funcionar en planta.

Para sacar conclusiones de los datos generados, se propondrá e implementará una solución que permita evaluar los resultados de producción de manera rápida y visual. Estas herramientas tendrán como objetivo mostrar la información de forma gráfica para dar claridad en la solución a los problemas ocurridos. Las visualizaciones planteadas deberán ser capaces de gestionar un gran volumen de datos y de ofrecer interacciones que sean intuitivas para el usuario.

4.2. Objetivos específicos

- Investigar sobre las técnicas de extracción y explotación de datos mediante *Computer Vision*, en especial las técnicas de *Deep Learning* para la inspección de defectos superficiales.
- Emplear metodologías que permitan obtener altas precisiones en la detección de defectos independientemente del material y del tipo de defecto.
- Facilitar la detección de defectos en las imágenes aplicando pre-procesos.
- Comprobar la estabilidad de la CNN generando oclusiones que aseguren la robustez y la obtención de resultados precisos y de calidad.
- Investigar nuevos métodos de adquisición de imágenes que eliminen los problemas generados por el material y con una rapidez que se ajuste a las necesidades de los entornos industriales.

- Emplear bases de datos de imágenes que permitan realizar una comprobación de las precisiones obtenidas en la extracción de los defectos.
- Establecer una arquitectura de almacenamiento *Big Data* que se adecue a la cadencia de producción y sea segura
- Emplear técnicas de *Visual Analytics* que permitan obtener a diversos perfiles con diferentes roles en la producción el estado en el que se encuentra la fabricación y localizar los problemas que puedan estar causando los defectos

4.3. Metodología

La hipótesis sobre la que se sustenta el trabajo se basa en que **el uso de técnicas de extracción de datos mediante *Deep Learning* permite construir sistemas de *Visual Analytics* que facilitan la toma de decisiones relacionadas con defectos superficiales**. Para demostrar esta hipótesis se han realizado una serie de experimentos.

Como punto de partida para establecer una posible solución al problema, en el Capítulo 3 se ha presentado el Estado del Arte sobre el control de defectos superficiales empleando técnicas de visión artificial, así como la problemática asociada a ellos. Para dimensionar los problemas y elegir las técnicas se consultarán artículos en revistas indexadas.

A continuación, se trabajará en el proceso de adquisición de imágenes para crear la base de datos con la que entrenar el sistema de *Deep Learning*. Para solventar los problemas que se tiene por la iluminación y la reflectividad del material se probarán distintos tipos de iluminación, y se elegirá la más adecuada.

Junto al requisito de la iluminación, se debe tener en cuenta la frecuencia de captura, ya que el tiempo de adquisición debe ser bajo para poder ser empleado en un proceso de continuo de alta velocidad. Se emplearán técnicas novedosas de adquisición como es el sistema Trevista junto a un banco de adquisición lineal que simula el proceso de laminación en continuo. Éste cuenta con una plataforma robotizada para desplazar la pieza y un sistema de iluminación estéreo-fotométrico que resalta los defectos relevantes del material, junto a un sensor lineal que capta en alta resolución cada parte de la pieza.

Una vez obtenidas las imágenes con las que trabajar e identificados los defectos con claridad en ellas, se emplearán diferentes arquitecturas de *Deep Learning* para intentar mejorar los métodos empleados hasta la actualidad. Para poder realizar una comparación de los resultados que se van obteniendo se empleará la base de datos pública NEU que contiene imágenes de defectos superficiales de acero. Esta base de datos es empleada por algunos métodos estudiados en el estado del arte, por lo que la precisión obtenida será contrastada con los demás métodos.

Para mejorar el proceso y la precisión de la clasificación, se aplicarán diferentes experimentos como alterar el número de imágenes de entrenamiento, realizar aumentado de datos o aplicar diferentes pre-procesos que ayuden a resaltar mejor los defectos y a tener un número equilibrado de muestras de cada tipo de defecto.

Con los parámetros de configuración obtenidos a partir de los experimentos, se emplearán diferentes arquitecturas de *Deep Learning* como AlexNet, GoogleNet y ResNet, para conseguir un resultado más preciso. Para ello se realizará el entrenamiento con 4 *folds*, esto significa que se dividirán los datos en 4 grupos. Dos de estos grupos se emplearán en el entrenamiento, el cual se era validando con otro de los grupos. Para saber la precisión obtenida, se realizará el testeo con el cuarto grupo, teniendo así resultados de clasificación de imágenes que no ha visto nunca el modelo.

A continuación, se elegirá el sistema de *Big Data* más oportuno para albergar las imágenes y que sea compatible con los sistemas de *Deep Learning* empleados, además para su elección se tendrá en cuenta la arquitectura del sistema. Se tendrá en cuenta la escalabilidad y la capacidad de almacenamiento que ofrezcan y la naturaleza de los datos para las que están diseñadas.

Por último, se crearán diferentes visualizaciones en JavaScript empleando diferentes librerías como d3 o canvaJS. Las visualizaciones deberán señalar las relaciones entre la producción y los defectos generados, para poder detectar tendencias o patrones. Estas facilitarán el entendimiento de la generación de los defectos, infiriendo conocimiento sobre el proceso de producción que se posea en la planta y poder tomar decisiones que lo mejoren. Permitirán la observación de la realidad desde diferentes puntos de vista, es decir, se adaptarán a diferentes roles presentes en la planta como por ejemplo el operario o el gerente.

5. Desarrollo de la contribución

En este capítulo se realizará el planteamiento seguido en la comparativa realizada en el trabajo y se detallarán los experimentos realizados para encontrar el método adecuado a la problemática expuesta. Por último, se expondrán los resultados obtenidos.

5.1. Planteamiento de la comparativa

El control y la explotación de los datos generados en tiempo real en una empresa es un arma estratégica que ayuda en la gestión y en la toma de decisiones. Así mismo, permite adelantarse a futuros acontecimientos y a realizar cambios que soluciones problemas que afecten negativamente a la empresa.

Tras el estudio realizado sobre la importancia de los defectos superficiales generados en la fabricación del acero y las posibles soluciones en la detección, se ha visto que con el empleo de técnicas de *Deep Learning* se obtienen resultados muy buenos en cuanto a la clasificación de objetos. Estas técnicas unen la extracción de características y la clasificación, por lo que últimamente su uso está en auge. Estas técnicas permitirán extraer los datos acerca del estado del material componiendo así una base de datos para controlar su calidad y corregir errores de producción.

La **primera contribución** de este trabajo es emplear un sistema de captura que evite introducir ruido en la adquisición. Esto es debido a la variabilidad de los defectos y la diferencia de las superficies, así como su manera de reaccionar a la captura y la iluminación. Este sistema de captura es el llamado Trevista, explicado en más detalle en la Sección 2.1.1. Empleando las imágenes que más información ofrecen acerca del defecto, se compondrá una imagen de falso color introduciendo en cada canal las imágenes elegidas.

Antes de entrenar una red neuronal, se debe realizar una serie de configuraciones de esta que sean las óptimas para la problemática que se posee. Para tener una base con la que compararse y ver que las elecciones elegidas son las correctas, se emplean imágenes de una base de datos pública con defectos superficiales en el acero de Northeastern University (NEU) [49]. Estas imágenes son empleadas por algunos de los autores mencionados en el Estado del Arte, por lo que la **segunda contribución** será comparar los resultados obtenidos con el método propuesto.

En cuanto a los experimentos elegidos para obtener la mejor configuración en la extracción de datos y que esta sea fiable, se realiza primero una serie de operaciones de mejora de la imagen, como por ejemplo el ecualizado de histogramas o el resalto de bordes. Con este pre-proceso se observa si la precisión en la clasificación sufre una mejora.

En la siguiente fase se prueban las arquitecturas más empleadas en la detección de elementos similares, como son AlexNet, GoogLeNet y ResNet, observando cómo responden al caso presentado. Adicionalmente, habrá que adaptar estas arquitecturas para conseguir mejores resultados.

Con el propósito de obtener una base de datos amplia y evitar el *overfitting* se realiza ampliado de datos, aplicando giros, rotaciones, escalados y cambios en la iluminación, ya que son casos que pueden darse en la realidad puesto que la orientación y tamaño de los defectos es aleatoria. Con el término *overfitting* se hace referencia al sobreajuste que puede sufrir el modelo al entrenar con datos con valores iguales.

Posteriormente, como cada arquitectura tiene un tamaño de entrada de imagen diferente, se prueban diferentes tamaños para averiguar cuál es el tamaño óptimo en este caso. Tras esto, se evalúa el número de imágenes de entrenamiento para ver si se obtiene más precisión en la clasificación al aumentar el conjunto hasta alcanzar el punto máximo.

Una vez configurados los parámetros ideales de entrenamiento, se procede a entrenar la red y a evaluar su robustez generando cambios en la iluminación mínimos y oclusiones aleatorias.

La **tercera contribución** es emplear las nuevas imágenes generadas con la información de Trevista con las que se pretende mejorar en la detección y la clasificación ya que ofrecen información diferente a la que se puede captar empleando métodos de captura tradicional y anotar los defectos para crear parches que mejoren el entrenamiento de la red neuronal. Se realiza una red neuronal binaria que sea capaz de detectar si existe o no defecto en la imagen a testear, ya que no existen tantas muestras disponibles como para clasificar por tipología los defectos.

La **cuarta contribución** es emplear la arquitectura de *Big Data* adecuada que ayude a simplificar la tarea de conectar y explorar los datos no estructurados generados en la captura. Esta arquitectura se adapta al volumen, complejidad y velocidad de crecimiento de los datos de los defectos de la producción. También asegura la seguridad y el respaldo ante perdidas o fallos del sistema necesarios en entornos de fabricación. Permite además el acceso de manera sencilla a los datos desde otras aplicaciones para el entrenamiento del modelo de extracción de datos y la visualización.

La **quinta contribución** es, una vez obtenido el modelo y la arquitectura de almacenamiento correcta, plantear una serie de visualizaciones que permitan ir mostrando los datos de la fabricación a los diferentes roles de trabajadores de la fábrica. Las visualizaciones estarán enfocadas desde que el gerente tenga la visión de lo que está ocurriendo de manera general,

hasta que el operario vea exactamente y de manera más detallada los defectos generados para poder ajustar el proceso. Estas permiten que cualquier persona sea capaz de obtener información acerca del proceso y una explicación sobre los resultados de producción.

La **sexta contribución** es obtener un sistema completo que va desde la extracción de los datos, la arquitectura de almacenamiento y la explotación de los mismos que genera conocimiento extra de la empresa, generando inteligencia de negocio. Las conclusiones que se extraen de este proceso permiten sacar evidencias que facilitan la toma de decisiones, ahorran costos y tiempos, incrementan la eficiencia y aumentan la fiabilidad de la calidad en los productos.

5.2. Desarrollo de la comparativa

En este capítulo se desarrollará la comparativa realizada y los experimentos desarrollados para obtener los resultados y las conclusiones. El código empleado para llevar los experimentos a cabo se encuentra accesible públicamente en Github[5]. El contenido de esta sección se ha enviado a la conferencia *9th International Conference on Intelligent Systems IS'18* [6] y se encuentra en proceso de revisión.

En primer lugar, el propósito ha sido generar una base de datos con la que trabajar para modelar el sistema. Tras el análisis de cómo se genera de forma óptima esta base de datos, se han escaneado con el sistema Trevista empleado probetas de una línea de producción real, simulando así el escenario de fabricación. Para entrenar las redes neuronales se necesita un amplio número de muestras de cada defecto que cubra lo más posible todas las opciones en las que el defecto pueda aparecer. Las bases de datos de entrenamiento deben ser equilibradas para que el resultado no se vea sesgado. Por eso, cómo el número de muestras que se tiene por cada defecto no es equilibrado y para poder obtener una base con la que compararse, se utilizará la base de datos pública NEU, que contiene imágenes de defectos en el acero.

5.2.1. Bases de datos de defectos del acero NEU

Song et al. [27] de la Northeastern University han compuesto una base de datos pública llamada NEU. En esta base de datos existen seis defectos típicos que suceden en las superficies de acero, son los siguientes: arañazos, inclusión, manchas, picaduras, grietas y rallones. La base de datos en total cuenta con 1800 imágenes, 300 por cada tipo de defecto.

[5] https://github.com/fatimasaiz/NEU_dataset_experiments
[6] Página wed de 9th International Conference on Intelligent Systems IS'18 http://www.ieee-is2018.com/

Estas imágenes tienen una resolución de 200 x 200 pixels. En la Ilustración 25 se muestran dos ejemplos por cada clase.

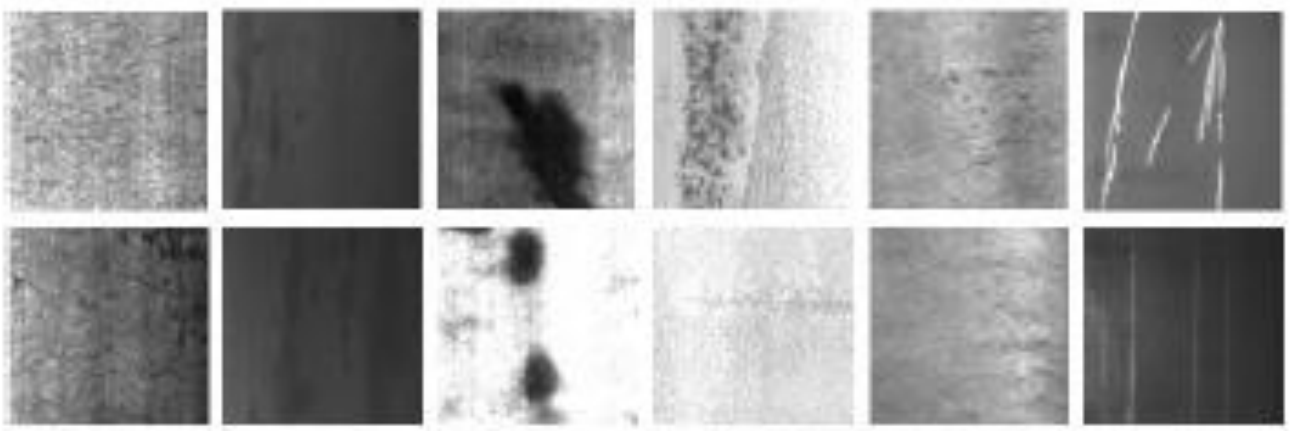


Ilustración 25. Ejemplo de defectos en la base de datos NEU.

Este conjunto de datos está disponible en su página web oficial [49]. Además de las imágenes, también se encuentran los ficheros XML que indican la situación del defecto para facilitar la tarea de detección.

Las dificultades principales que presenta este conjunto de datos son que los defectos tienen aspecto similar y que hay cambios en la iluminación y en el material que afectan a la escala de grises de los defectos.

Como se ha mencionado, se irá evaluando el resultado obtenido en cada experimento con los demás autores para tomar decisiones correctas en la configuración. Por este motivo, en un primer momento se ha seleccionado como arquitectura AlexNet [50]. Se ha elegido esta arquitectura por ser común en el reconocimiento de objetos en imágenes y por su simplicidad. Esta arquitectura es la ganadora de ILSVRC 2012[7] reduciendo el error top-5 a un 15,3%. AlexNet tiene un tamaño de entrada de 227 x 227 x 3. La arquitectura completa se muestra en la Ilustración 26.

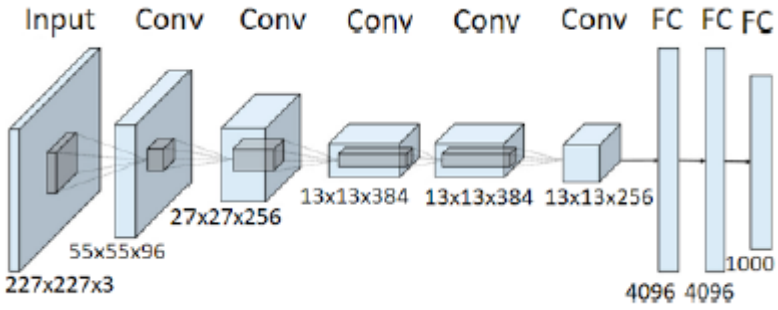


Ilustración 26. Arquitectura original de AlexNet.
Fuente: [50].

[7] *Imagenet Large Scale Visual Recognition Challenge* 2012. El objetivo general de este concurso es identificar automaticamente los objetos principales presentes en unas imágenes de prueba dadas. http://www.image-net.org/challenges/LSVRC/2012/index

5.2.2. Primer experimento: Ajuste del tamaño óptimo de las imágenes de entrada para el entrenamiento.

Empleando AlexNet y los datos de NEU se variará el tamaño de entrada de las imágenes para conseguir captar información significativa para la clasificación. El tamaño original de entrada de esta arquitectura es 227 x 227. En este experimento se va a evaluar la relación entre el tamaño de entrada de la CNN y los tamaños de lado de las imágenes de entrenamiento. Los tamaños de lado a evaluar son 75 x 75, 100 x 100, 125 x 125, 150 x 150, 175 x 175, 200 x 200 y 227 x 227. Los resultados obtenidos se muestran en la Tabla 4, donde aparece la precisión de clasificación, la desviación estándar y el tiempo de entrenamiento.

Tabla 4. Precisión obtenida en la clasificación y desviación estándar con diferentes tamaños de entrada usando AlexNet.

Tamaño de entrada	Precisión	Tiempo de entrenamiento
75 x 75	94.33 ± 0.734	2.08min
100 x 100	94.39 ± 0.840	2.32min
125 x 125	93.83 ± 0.840	2.46min
150 x 150	**95.11 ± 1.244**	**2.72min**
175 x 175	93.89 ± 0.968	3.07min
200 x 200	94.17 ± 0.493	3.1min
227 x 227	93.11 ± 1.267	3.88min

Observando los resultados obtenidos se muestra que el re-escalar las imágenes a un tamaño más pequeño ofrece mejores resultados que empleando el tamaño original. Este resultado se debe a que esta CNN tiene un campo receptivo pequeño en la primera capa (11 x 11), por tanto, cuando la imagen es re-escalada, este campo receptivo es capaz de extraer más información en este caso. Por los resultados obtenidos, las imágenes de entrada con las que se entrenará tendrán el tamaño 150 x 150.

5.2.3. Segundo experimento: Análisis de operaciones de pre-proceso a las imágenes.

El segundo experimento consiste en aplicar operaciones de pre-proceso a las imágenes de entrada para resaltar los defectos y mejorar la clasificación. De este modo, se pretende reducir la similitud entre las clases.

Las operaciones efectuadas son las siguientes:

- **Imadjustn**: aumenta el contraste de la imagen volumétrica de salida.
- **Histeq:** mejora el contraste usando la ecualización de histograma.
- **Localcontrast:** realza los bordes en las imágenes.

En la Ilustración 27 se muestra el resultado de las operaciones efectuadas comparadas con la imagen original.

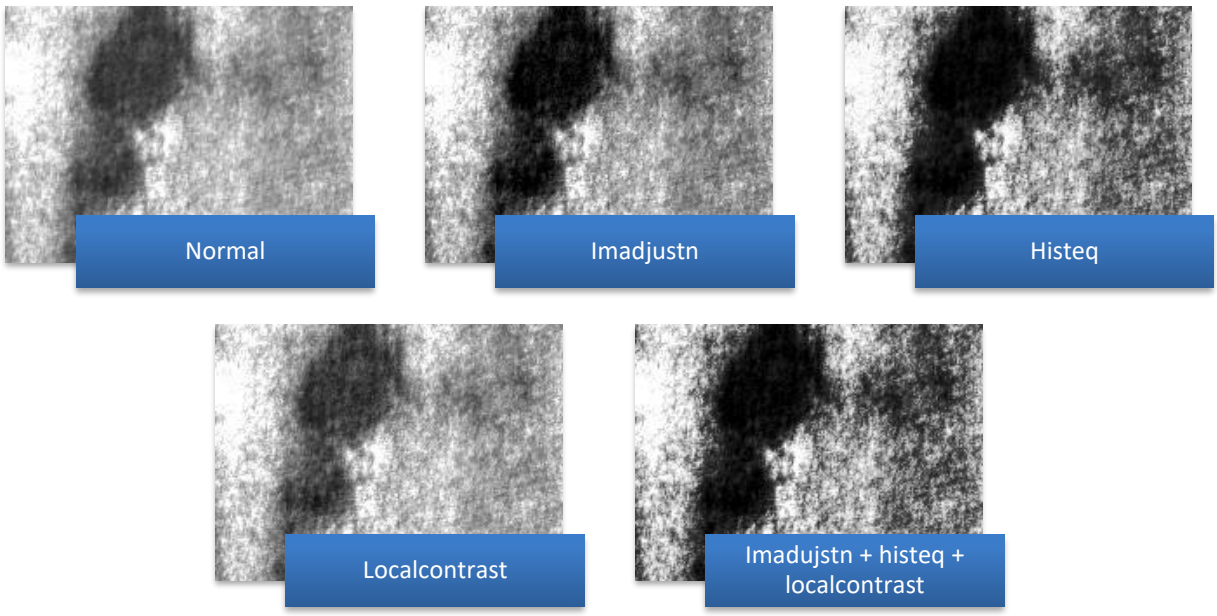


Ilustración 27. Efecto de los pre-procesos aplicados en una imagen concreta.

Observando los resultados obtenidos, mostrados en la Tabla 5, aplicando la operación de aumento de contraste **imadjustn** se consigue una precisión en la clasificación mayor (97,33%). Esta operación permite diferenciar mejor los defectos sin resaltarlos en exceso.

Tabla 5. Precisión de clasificación obtenida al aplicar diferentes pre-procesos en las imágenes de entrenamiento.

Tipo de pre-proceso	Precisión
normal	95.11
histeq	95.95
imadjustn	**97.33**
localcontrast	96.74
histeq + imadjustn + localcontrast	95.83

5.2.4. Tercer experimento: Evaluación del número óptimo de imágenes de entrenamiento

A continuación, se evaluará la precisión obtenida al incrementar el número de imágenes de entrenamiento. El número de imágenes será evaluado en un rango entre 10 y 150. Los resultados obtenidos se muestran en la Ilustración 28. Estos muestran que la precisión empeora a medida que el número de muestras de entrenamiento decrece. Aun así, la CNN obtiene una precisión del 86,17% con solo 40 imágenes por cada clase.

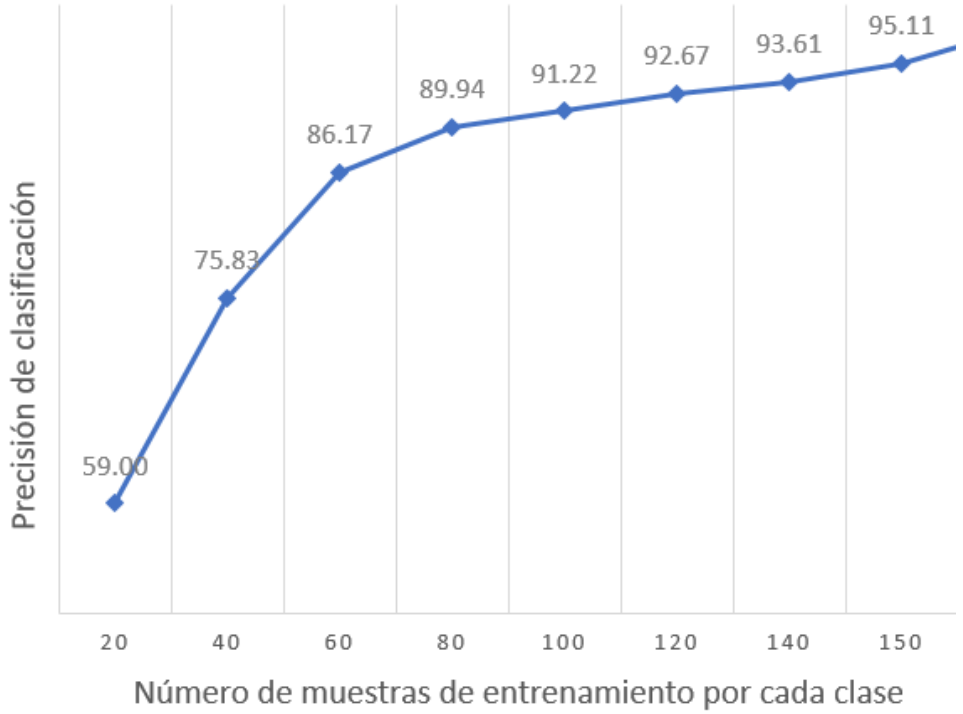


Ilustración 28. Precisión en la clasificación obtenida con un número de muestras de entrenamiento diferente para cada clase.

5.2.5. Cuarto experimento: Evaluación de la precisión obtenida aplicando las operaciones óptimas de aumentado de datos

Partiendo de la información obtenida en el experimento anterior, se va a aumentar el número de imágenes aplicando diferentes operaciones en las imágenes. El aumentado de datos consiste en generar nuevas imágenes a partir de las que se poseen para generar a partir de transformaciones nuevas situaciones de captura que se pueden dar. El objetivo de este experimento es encontrar las operaciones ideales de aumentado de datos para este conjunto de imágenes y obtener así un modelo más robusto.

Con esto se quiere probar si el generar imágenes nuevas empleando las que se tienen mejora la precisión de la red neuronal. Las operaciones que se realizan permiten hacer modificaciones menores al conjunto de datos existente que hagan creer a la red neuronal que las imágenes son distintas a las originales, y crear así más escenarios de aprendizaje que se pueden dar en el mundo real.

Una CNN puede clasificar robustamente objetos colocados en diferentes orientaciones, siendo invariante a la traslación, el punto de vista, el tamaño o la iluminación. Existen muchas operaciones posibles, pero en este caso se evaluará la rotación, escalado, traslación, reflexión y cortado. Por tanto, se aplican estas operaciones y una combinación de estas a las imágenes de entrenamiento y se vuelve a entrenar la CNN comparando las precisiones obtenidas con cada una de ellas.

Las pruebas realizadas demuestran que las rotaciones y los cambios en la escala tienen mayor impacto en el resultado final, es por eso por lo que el aumentado de datos se ha realizado aplicando rotaciones aleatorias en el rango de ±180º y el factor de cambios en la escala va desde 0,5 a 2.

La posible razón por la cual las demás operaciones disminuyesen la precisión puede ser debido a que, al aumentar y cortar las imágenes, el defecto desapareciese de esta y produjese falsos positivos en la clasificación.

Empleando estas dos operaciones se ha probado a incrementar el conjunto de datos N veces. En la parte derecha de la Ilustración 29 se muestra las precisiones obtenidas con los N aumentos, y se realiza una comparación con las precisiones obtenidas en el experimento anterior con un número más reducido de imágenes. Este experimento demuestra que la precisión mejora a medida que se aumenta el conjunto de datos hasta que llega un punto en el que se estabiliza. Cabe destacar que el testeo de la red se ha realizado con imágenes originales, por lo que se demuestra la robustez del modelo obtenido.

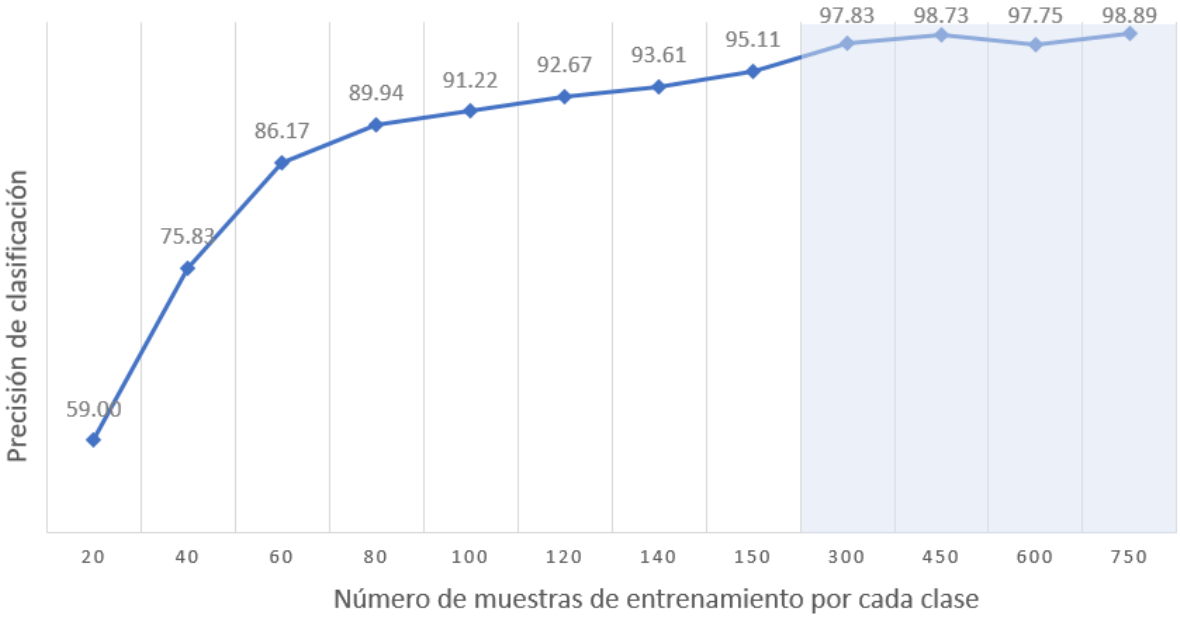


Ilustración 29. Precisión en la clasificación con un número diferente de muestras de entrenamiento. La parte derecha muestra los resultados del aumentado de datos.

5.2.6. Quinto experimento: generación de oclusiones y cambios en la iluminación

Para demostrar la robustez del modelo se realiza este experimento que consiste en generar oclusiones aleatorias a lo largo de las imágenes y con diferentes porcentajes de oclusión y una simulación de cambios en la iluminación aplicando cambios en el brillo.

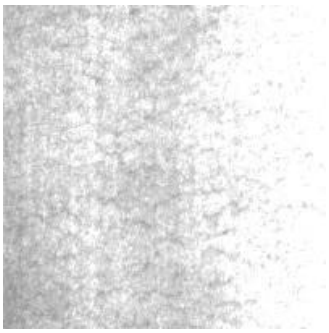


Ilustración 31. Aumento en el brillo del 40%.

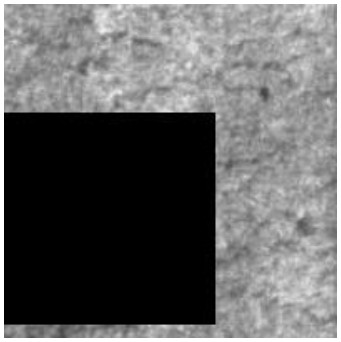


Ilustración 30. Oclusión aleatoria del 40%.

La Ilustración 31Ilustración 30 muestra un ejemplo del incremento en el brillo de la imagen en un 40%, mientras que la Ilustración 31 muestra la oclusión generada con una cobertura del 40%.

El clasificador obtenido es muy robusto ya que aun con un 40% oclusión y un aumento del brillo en un 70% la precisión obtenida es del 92% como se muestra en la Ilustración 32.

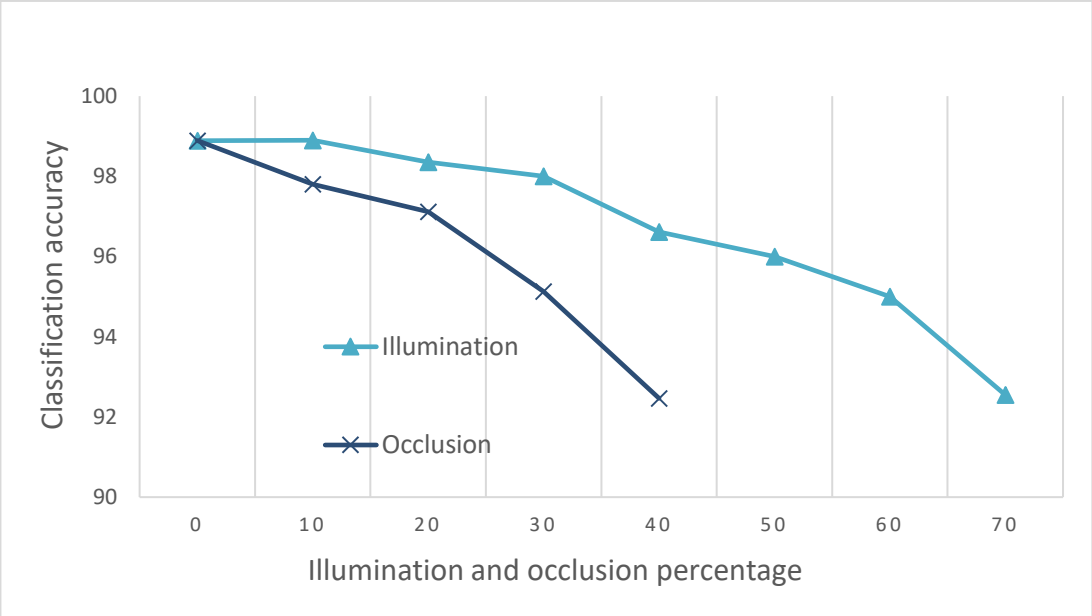


Ilustración 32. Precisión de la clasificación generando oclusiones aleatorias y cambios en la iluminación.

5.2.7. Sexto experimento: cambios en la arquitectura de la CNN

Por último, se evaluarán otras arquitecturas conocidas empleando los mismos parámetros de configuración que se han ido evaluando en los anteriores experimentos. Las arquitecturas que se van a utilizar en la comparativa son las siguientes:

- **GoogLeNet** [51] fue la arquitectura ganadora de la competición ILSVRC 2014[8]. Esta acumuló un error top-5 del 6,67%, el cual está cerca de la capacidad humana. GoogLeNet es una CNN con 22 capas, mostradas en la Ilustración 33, pero con un número reducido de parámetros (4 millones) en comparación con AlexNet (60 millones). El tamaño de entrada de esta arquitectura es 224 x 224 x 3.

[8] *Imagenet Large Scale Visual Recognition Challenge* 2014. El objetivo general de este concurso es identificar automáticamente los objetos principales presentes en unas imágenes de prueba dadas y su localización.
http://www.image-net.org/challenges/LSVRC/2014/index

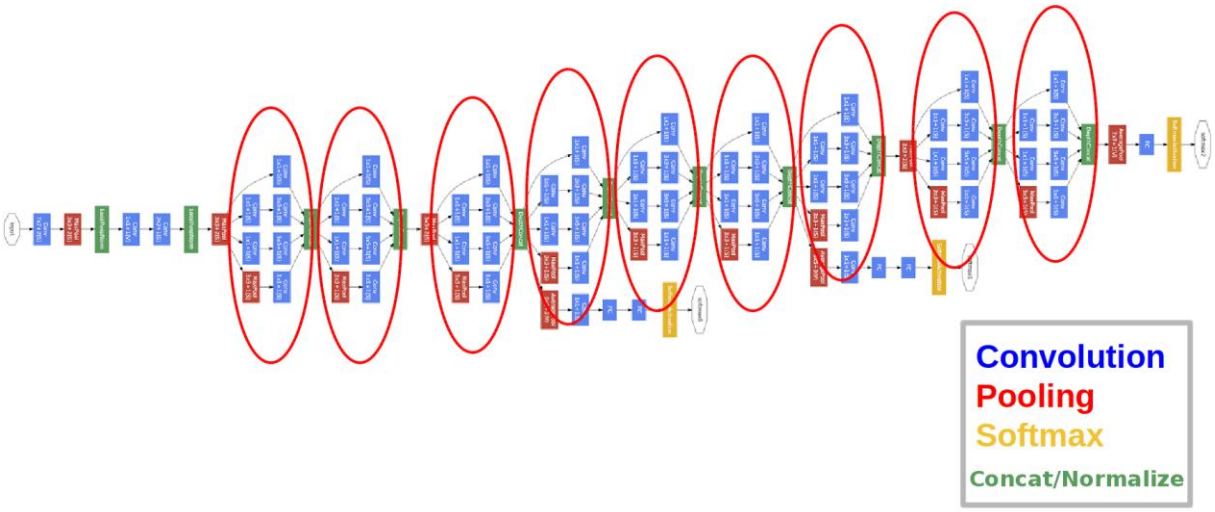


Ilustración 33. Arquitectura de GoogLeNet.
Fuente: https://leonardoaraujosantos.gitbooks.io

- **Resnet** [52] fue la ganadora en ILSVRC 2015[9]. Esta arquitectura introduce una capa que omite conexiones. Esta arquitectura tiene 152 capas con una baja complejidad, como se muestra en la Ilustración 34. Resnet obtuvo un error de 3,57% en top-5, batiendo la capacidad humana en el dataset con el que se probó. El tamaño de entrada de las imágenes en la red es igual al de GoogLeNet, 224 x 224 x 3.

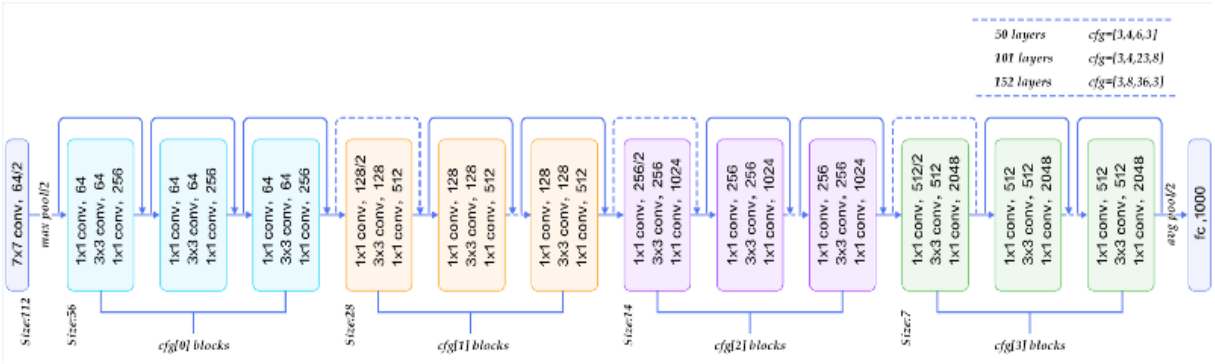


Ilustración 34. Arquitectura Resnet.
Fuente: https://medium.com/

Estudiando el campo receptivo de estas dos arquitecturas, se observa que es menor en los dos casos al empleado en AlexNet. Al tener un campo menor, la capacidad de extraer detalles

[9] *Imagenet Large Scale Visual Recognition Challenge* 2015. El objetivo general de este concurso es identificar y localizar automáticamente los objetos principales presentes en unas imágenes y vídeos de prueba dados.
http://www.image-net.org/challenges/LSVRC/2015/index

más complejos de las imágenes aumenta, por lo que se va a mantener el tamaño de entrada empleado y las imágenes obtenidas con el preprocesamiento y el aumentado de datos en el entrenamiento con AlexNet para poder realizar una comparación en las mismas condiciones.

Cómo las arquitecturas tienen un tamaño predefinido por los autores para la competición, se han tenido que re-escalar las imágenes. En la Tabla 6 se comparan los resultados obtenidos con los demás métodos mencionados en el estado del arte que usan la misma base de datos. En ella se realiza una comparación entre los métodos de *Machine Learning* y los de *Deep Learning* y las precisiones obtenidas. Cada autor ha realizado una serie de procesos para obtener esas precisiones de clasificación, por lo que se compara con los tres métodos propuestos en este trabajo.

Resnet es la arquitectura que mejor precisión ha obtenido (99,95%), aunque también el mayor tiempo de entrenamiento por su complejidad y su número de capas. En cuanto al tiempo de clasificación, los resultados obtenidos son positivos ya que cada imagen se clasifica en 0.006-0,019 segundos en cualquiera de las arquitecturas.

Tabla 6. Comparación de los resultados obtenidos.

Referencia	Método		Precisión	Tiempo	
	Características	Clasificación		Entrenamiento (por fold)	Test GPU/CPU (por imagen)
[26]	LBP	SVM	97.93		
		NNC	95.07		
	LTP	SVM	98.22		
		NNC	95.93		
	CLBP	SVM	97.93		
		NNC	98.28		
	AECLBP	SVM	98.87		
		NNC	97.93		
[27]	SCN	SVM	98.6		
[29]	DenSIFT+IFV	SVM	99.56		
	LSP		99.87		--/1.07s
[8]	HOG + Gabor + GLCM + LBP	SVM + BYEC	83.33		
	Gabor + GLCM + LBP + GLH		83.56		
	HOG + GLCM + LBP + GLH		84.18		
	HOG + Gabor + LBP + GLH		84.69		
	HOG + Gabor + GLCM + GLH		85.24		
[32]	DeCAF		99.21		
	Método propuesto + AlexNet		98.89	16.9min	0.006/0.008s
	Método propuesto + GoogLeNet		99.8	39.57min	0.012/0.069s
	Método propuesto + ResNet		99.95	280min	0.019/0.167s

5.2.8. Séptimo experimento: Emplear imágenes capturadas con el sistema Trevista

Con la configuración y los parámetros obtenidos en los anteriores experimentos, se quiere comprobar si el sistema de captura propuesto mejora los resultados obtenidos y aumenta la precisión en la clasificación.

Por ese motivo, primero se entrena un clasificador binario que solo emplea las imágenes de textura obtenidas con la Trevista, sin añadirle ninguna información extra del estereofotométrico. Este clasificador determinará si la imagen tiene o no defecto. La arquitectura empleada para el entrenamiento es Resnet debido a los resultados positivos previos obtenidos.

A continuación, se crean imágenes de falso color introduciendo en cada canal la información capturada en las imágenes de textura, rango y curvatura, y se crean parches de entrenamiento. Un ejemplo de un parche en falso color empleado se muestra en la Ilustración 35.

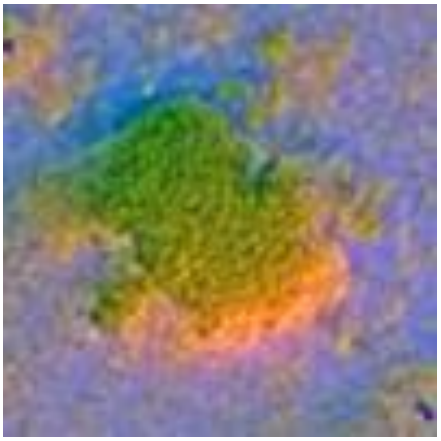


Ilustración 35. Imagen de falso color de un defecto de mancha.

Con la misma arquitectura y los mismos parámetros se repite el entrenamiento del modelo. Gracias a la información extra que aporta acerca de la superficie este sistema de captura, se logra conseguir una mejora de aproximadamente 10% en la clasificación. Este porcentaje puede parecer bajo, pero supone una mejora considerable ya que aumentar la precisión cuando el valor esta aproximadamente en el 80% es complicado.

Debido a la diferencia en el número de muestras de normalidad y de defecto, no se ha podido entrenar un modelo consistente en el que se diferencie cada uno de los defectos, ya que por estadística la clasificación en caso de duda tenderá hacía la clase que más muestras posea.

5.2.9. Arquitectura de almacenamiento

Tras elegir el método de extracción de datos adecuado, es necesario establecer ahora un sistema de almacenamiento. Con este sistema se debe garantizar que los datos se almacenen de una manera correcta y sean transformados tras su captura para añadirles valor y obtener información.

Los datos se almacenan cumpliendo una serie de requisitos de calidad, que se basan en:

- La completitud: los datos obtenidos cubren un porcentaje elevado de la población total.
- La credibilidad: están obtenidos a partir de un método fiable y contrastado con otros métodos.
- La precisión: los datos son correctos y hay un porcentaje bajo de errores.
- La interpretabilidad: los datos son fácilmente entendibles por una persona que no sea experta en el tema.

La organización de los datos es fundamental para conseguir resultados, para gestionarlos se suelen emplear ficheros. En este caso, los datos son imágenes, y en el caso del entrenamiento con las imágenes capturadas se cuenta también con anotaciones en formato XML.

El formato XML (*eXtended Markup Language)* es un formato de fichero plano de los más comunes, este utiliza etiquetas como parte de la estructura y formato de los datos que contiene.

Todos estos datos se vuelcan en una base de datos. Cómo se ha estudiado en el estado del arte, se emplea una base de datos NoSQL que permita el almacenamiento de documentos, ya que los datos que se poseen son relativamente complejos. Por este motivo se elige MongoDB, que ofrece agilidad y tiene un lenguaje completo de búsquedas.

La arquitectura planteada es la siguiente:

- Para proteger los datos se han implantado medidas de seguridad que restringen el acceso a la base de datos mediante autenticación.
- Se ha restringido los accesos a nivel de red y se ha empleado un firewall para restringir las comunicaciones del exterior y de otros dominios.
- Para que el sistema sea capaz de soportar la inmensa cantidad de información que le llega, se ha instalado un mecanismo que permita incrementar el rendimiento y escale la base de datos de manera horizontal. Para ello se ha empleado *Sharding,* que consiste en dividir el conjunto de datos y distribuirlo en varios servidores. La ventaja de este mecanismo, aparte de aumentar la capacidad de almacenamiento,

es que permite ejecutar consultas rápidas sin necesidad de adaptarlas por estas distribuidas en diferentes servidores. Un esquema de esto se muestra en la Ilustración 36.

- Cada servidor de la arquitectura es un *replica set*, esto significa que permite una recuperación a fallos de forma automática asegurando así alta disponibilidad y consistencia. Los nodos llevan control sobre el funcionamiento y la accesibilidad de los otros nodos mediante el envío de pings cada dos segundos.
- Para acceder a los datos y poder así entrenar la red neuronal se conecta MongoDB con Matlab, pudiendo acceder a todos los servidores.
- Esta base de datos permite también generar visualizaciones de una manera sencilla ya que se pueden ejecutar peticiones que devuelven JSON, por lo tanto, se pueden importar de manera sencilla en ficheros HTML empleando librerías básicas de JavaScript.
- Se puede realizar un incremento en el rendimiento a través de la creación de índices que agilicen el tiempo de las consultas.

La configuración de estos parámetros se muestra en el Anexo 1.

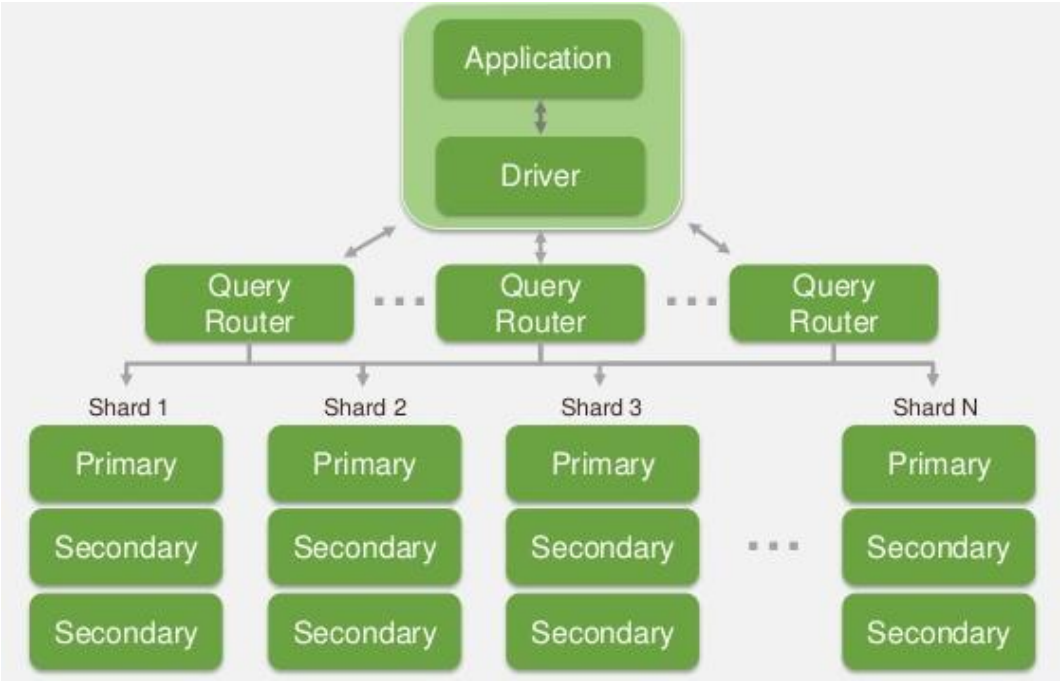


Ilustración 36. Arquitectura de sharding en MongoDB.
Fuente: https://www.slideshare.net/mongodb

5.2.10. Visualización de los resultados

En esta sección se quiere recalcar la importancia de la visualización de datos y la importancia de interactuar con los datos para obtener conocimiento de calidad. Una vez que se dispone la información se debe hacer uso de ella. El uso y el acceso a esta información suponen actualmente un cambio transcendental en la manera de pensar en la producción. El manejo adecuado de los datos resulta de gran relevancia dentro de la gestión empresarial y posee un gran potencial, por eso es necesario crear un buen sistema de información.

Con la visualización se pretende obtener conocimiento a partir de la información obtenida de los datos capturados de una manera sencilla. Estas visualizaciones tienen que estar alineadas con el objetivo y las necesidades del negocio. En este caso, es necesario tener un conocimiento de los defectos generados, buscar posibles causas de la generación de estos y controlar que el material empleado este en buen estado. Para lograr esto es necesario gestionar el conocimiento que encierran los datos disponibles y aprovechables de la información.

Estos objetivos se logran empleando analítica aplicada al negocio. Se deben disponer de visualizaciones interactivas e intuitivas que permitan a los usuarios acceder y analizar a os datos sin necesidad de programar.

Para lograr cualquier objetivo planteado y tener conocimiento de los procesos es necesario medir el estado de estos con el fin de tomar decisiones. Para conseguir las metas propuestas y mejorar se propone el uso de un ***dashboard*** que permita interpretar de una manera rápida todos los datos generados. Un *dashboard* es una representación gráfica de las métricas que se crean importantes y relevantes. Este tipo de representación permite visualizar errores o prácticas erróneas, conocer el rendimiento, ver el cumplimiento de la estrategia planteada o corregir de una manera rápida acciones a partir de los resultados generados.

Se han diseñado dos visualizaciones de los resultados dependiendo de los intereses de la persona a las que estén dirigidas, ya que no todos los roles de la empresa necesitan recibir la misma información.

La primera visualización es un *dashboard* en el que se resumen el número de defectos que se han dado en la producción en un día, turno y línea concretos, mostrado en la Ilustración 37.

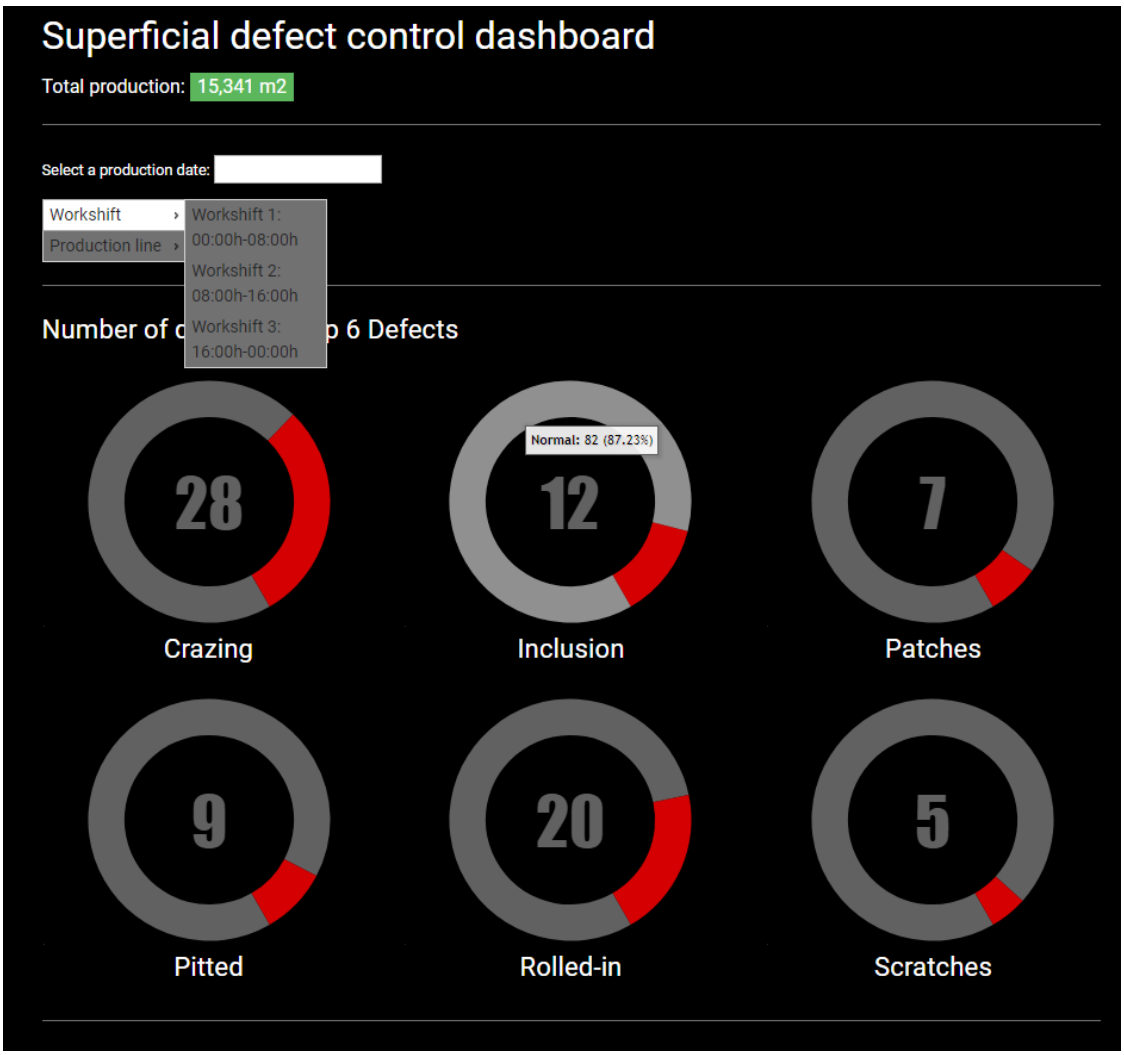


Ilustración 37. Dashboard planteado para la visualización global de resultados.

Esta visualización además ofrece el número de metros cuadrados de material analizado e información adicional sobre los porcentajes de los defectos ocurridos. Esta visualización se propone por su sencillez en la interpretación. A pesar de que los gráficos de donuts no son precisos, se considera adecuado en este caso porque se representan solo dos clases y además permite conocer el porcentaje exacto al deslizar el ratón por encima. En el centro de ellos además se visualiza de manera rápida el número exacto de defectos ocurridos.

La visualización se basa en las librerías de JavaScript jQuery [53] y CanvasJS [53]. JavaScript es un lenguaje de programación inventado por Brendan Eich, cofundador del proyecto Mozilla, en 1995 que permite crear visualizaciones dinámicas e interactivas en páginas web. Este lenguaje puede ser aplicado en un documento HTML.

- **jQuery**: es una librería de JavaScript multiplataforma cuyo principal objetivo es facilitar la programación, ofreciendo funcionalidades que están basadas en

JavaScript que permiten ahorrar tiempo y espacio. Fue publicada en "BarCamp NYC" en el año 2006 por John Resign. Es software libre y de código abierto.

- **CanvasJS:** en una librería de gráficos en HTML5 con una API sencilla que permite desarrollar *dashboards* con mayor rapidez que empleando Flash o SVG. Soporta múltiples navegadores como Chrome, Firefox, Safari o IE8+ y es multiplataforma.

La segunda visualización está más orientada a tener un control en tiempo real y más detallado de la producción, mostrada en la Ilustración 38. Consiste en un visor en el que se va mostrando la imagen captada en tiempo real de la línea con los defectos remarcados en ella. Estos están diferenciados por colores, pero para que esto no sea una limitación para personas con impedimentos en distinguir colores, se visualiza al lado de cada cuadrado el número de defecto correspondiente.

En ella existe un apartado de anotación, que permite corregir en las imágenes los defectos que se hayan clasificado mal, o anotar los que no han sido detectados seleccionando el tipo al que pertenece. Con esos datos nuevos actualizados se permite el reentrenamiento de la CNN. Existe también la posibilidad de cambiar de modelo de CNN.

Para tener una idea del estado de producción en el turno, se crea una gráfica en la que se representa la cantidad de defecto por tipo generado en el turno de trabajo. Con esta herramienta se pretende dar al usuario el conocimiento necesario para tomar decisiones que mejoren el proceso de producción al momento y permita corregir errores mejorando así la calidad del producto final y evitando desperdicios.

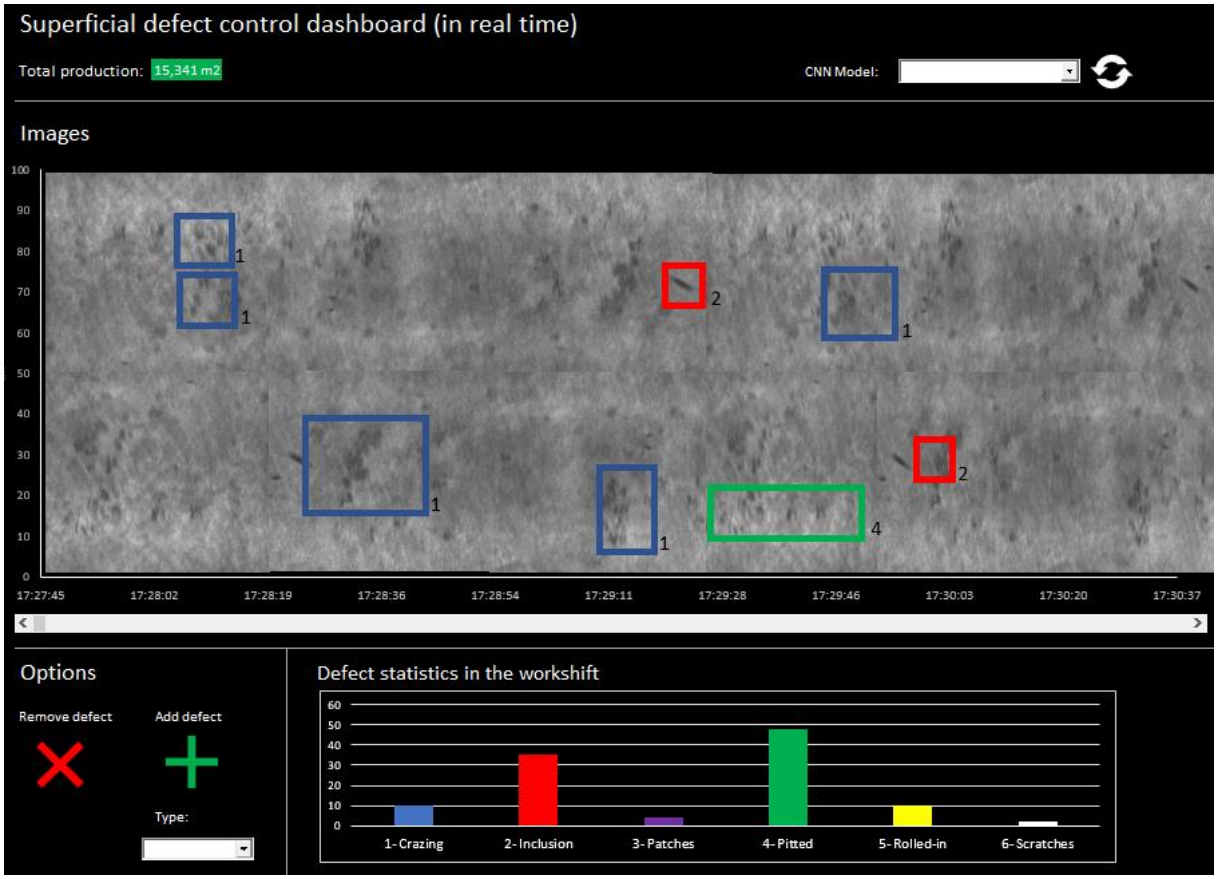


Ilustración 38. Segunda visualización propuesta.

5.3. Discusión y análisis de resultados

Tras los experimentos realizados, se puede afirmar que el uso de técnicas de *Deep Learning* es adecuado en la extracción de los datos de los defectos en la producción del acero, obteniendo precisiones mejores que empleando técnicas clásicas de *Machine Learning*. Es importante crear una buena configuración del sistema de entrenamiento y adaptarlo a la necesidad de la situación, además de crear una arquitectura de almacenamiento que sea escalable y ofrezca rapidez en las consultas. Esta a su vez tiene que ser fácilmente accesible desde otras plataformas para explotar los datos y tiene que garantizar la seguridad de los mismos.

Se ha observado que el tamaño óptimo de entrenamiento de la red para la extracción de los defectos no es el original que la arquitectura posee, sino que es menor. Esto es debido a que hasta el mínimo detalle que se posee en las imágenes es significativo para la clasificación y que la arquitectura empleada posee un campo receptivo en la primera capa muy pequeño (11 x 11). Por tanto, cuando la imagen es re-escalada a un tamaño menor, este campo receptivo es capaz de extraer más información espacial para el entrenamiento.

Se ha detectado también la necesidad de aplicar pre-procesos a las imágenes que permitan diferenciar más las anomalías respecto a la normalidad del material. En este caso se han obtenido mejores resultados al aplicar filtros más suaves que no contrastasen en exceso el defecto. Este exceso podría generar falsos positivos al resaltar cualquier marca que estuviera dentro de la normalidad.

El número de imágenes de entrenamiento es un factor muy importante a tener en cuenta. Al entrenar con menos muestras la precisión en la clasificación decrece. Pero es importante tener en cuenta que no solo afecta el número de imágenes, también su calidad. Esto se demuestra tras realizar el aumentado de imágenes, en el que llegado a un punto la precisión en la clasificación se vuelve constante ya que las imágenes sintéticas generadas no ofrecen más información relevante que permita enseñar a la red. Por lo tanto, es bueno realizar un aumentado de imágenes con operaciones que representen situaciones que se puedan dar ya que aumenta la robustez del modelo, pero no en exceso. Algunas de las operaciones realizadas no aumentaban la precisión en la clasificación ya que generaba situaciones que distorsionaban el defecto a clasificar. Un ejemplo de estas situaciones son el cambio de escala o los cambios en la iluminación. El sistema de captura como se ha descrito anteriormente tiene que asegurarse de que la iluminación sea constante, por lo tanto, esa operación emplearía datos que no ocurrirían en la realidad. El cambio de escala en cambio puede producir falsos positivos al considerar una imagen como defectuosa cuando por efectuar la operación el defecto ha podido desaparecer.

La configuración elegida ha resultado adecuada ya que, a pesar de generar oclusiones aleatorias y cambios en la iluminación de las imágenes de test, aunque no deberían de generarse en la realidad, se ha obtenido una precisión en la clasificación alta. Es por eso que se han evaluado las demás arquitecturas.

La precisión obtenida con GoogLeNet (99,8%) y ResNet (99,95%) es mayor debido a que son redes pre-entrenadas y se ha realizado la adaptación a las necesidades concretas de la clasificación de defectos superficiales en el acero. Los parámetros que se han establecido en el inicio para AlexNet han sido adecuados para estas arquitecturas, ya que son capaces de extraer detalles más pequeños de las imágenes de entrenamiento. En cuanto a la velocidad de la clasificación de los defectos, empleando una GPU GeForce GTX 960 se consiguen tiempos de entre 0,006s a 0,019s por imagen, lo cual cumple con las exigencias de rapidez que exige la producción industrial. Para realizar una comparación con el tiempo obtenido por [29], se ha evaluado el testeo en una CPU 3.20GHz Intel(R) Core(TM) i5-6500, ya que ellos usan procesador 3.5GHz Intel Xeon. El método propuesto en este trabajo es aproximadamente 10 veces más rápido que el propuesto por ellos.

El uso de las imágenes obtenidas con el sistema de captura alternativo planteado (Trevista) ha demostrado ser capaz de mejorar la precisión en la clasificación gracias a la información adicional aportada sobre la superficie. Se ha conseguido una mejora del 10% aproximadamente empleando las imágenes de falso color generadas con las tres imágenes que más información revelaban sobre los defectos, por lo que supone un avance en los sistemas de captura empleados hasta la fecha. Además, el emplear una cámara lineal permite tener capturas continuas de la producción y ofrece más rapidez que una cámara matricial.

La base de datos elegida para almacenar las imágenes de la captura ha sido la adecuada ya que es óptima para este tipo de archivos, ofrece una velocidad muy alta permitiendo acceder a mucha información en poco tiempo. El modelo elegido, al tratarse de una arquitectura NoSQL, responde a la necesidad de la escalabilidad horizontal. El uso de *Sharding* evita cuellos de botella y permite manejar grandes volúmenes de datos pudiendo abordar la cantidad de capturas que se realizan en cada línea.

Por último, las visualizaciones planteadas permiten extraer el conocimiento necesario de la cantidad de datos que se producen continuamente en la fábrica, desde un enfoque general en cuanto a todos los defectos hasta un enfoque minucioso que permite analizar cada imagen y rectificar el modelo de clasificación para mejorar en la producción. En el diseño de estas se ha tenido en cuenta la psicología de percepción del usuario, permitiendo extraer la información sin necesidad de procesamientos mentales posteriores. Se han seguido además principios del diseño gráfico, que son claves para crear visualizaciones efectivas.

6. Conclusiones

Tras realizar el trabajo, se pueden concluir respecto al objetivo de la investigación sobre las técnicas de extracción y explotación de datos mediante *Computer Vision*, en especial las técnicas de *Deep Learning* para la inspección de defectos superficiales, que se ha investigado sobre las técnicas de automatización de visión artificial empleadas, relatadas en el estado del arte haciendo un análisis de las empleadas hasta la actualidad. En especial se ha enfocado en las técnicas de *Deep Learning* aplicadas en la inspección de defectos superficiales donde se ha revisado el trabajo hecho hasta ahora por otros investigadores, así como los experimentos realizados y las arquitecturas y bases de datos empleadas.

Acerca del empleo de metodologías que permitan obtener altas precisiones en la detección de defectos independientemente del material y del tipo de defecto, se ha profundizado en emplear metodologías que permitan obtener altas precisiones en la detección de defectos como son las CNN, en concreto las arquitecturas empleadas (AlexNet, GoogLeNet y Resnet), pudiéndose adaptar estas mismas realizando una configuración a partir de los experimentos dados que permiten adaptar el sistema a cualquier tipo de material o defecto.

En cuanto a facilitar la detección de defectos, se han aplicado pre-procesos que mejoran esta tarea y ofrecen un resultado más preciso y de mayor calidad.

Se ha demostrado la estabilidad y robustez del sistema y del modelo propuesto generando ruido en las imágenes aplicando oclusiones. Por la precisión obtenida este permite dar un resultado fiable que incrementa la calidad de la producción por su alta tasa de aciertos.

Sobre investigar nuevos métodos de adquisición de imágenes que eliminen los problemas generados por el material y con una rapidez que se ajuste a las necesidades de los entornos industriales, se han estudiado los diferentes tipos de iluminación y el objetivo que tiene cada uno. Por ello se ha elegido el método que más se adecua al tipo de material y a los defectos que se tienen en él. Se ha entrenado el modelo empleando diferentes tipos de iluminación en la captura para verificar la mejora del sistema propuesto. Además, al ser una captura lineal, se adapta a la necesidad del entorno industrial en el que se emplean rodillos continuos.

Se ha contrastado también los resultados obtenidos con los obtenidos con diferentes métodos. Por ello se ha empleado la base de datos pública NEU, sobre la que se han realizado los experimentos.

Acerca del objetivo de establecer una arquitectura de almacenamiento Big Data que se adecue a la cadencia de producción y sea segura, se han estudiado los sistemas de almacenamiento existentes según su tipología y su finalidad, así como las ventajas que tienen

en cuanto a rapidez y capacidad de almacenamiento. Por ello se ha elegido la base de datos de MongoDB, que permite adaptarse al volumen de datos que se generaría en el entorno industrial descrito y que además es idónea para imágenes. Se han implementado protocolos de seguridad y de respaldo de la información. Además, se ha establecido una arquitectura que permita afrontar el ritmo de crecimiento de los datos y su procesado de manera eficiente.

Respecto a emplear técnicas de *Visual Analytics* que permitan obtener a diversos perfiles con diferentes roles en la producción el estado en el que se encuentra la fabricación y localizar los problemas que puedan estar causando los defectos, se han empleado técnicas de *Visual Analytics* creando visualizaciones que permitan conocer el estado de fabricación de una manera rápida y sencilla, además de tener opciones interactivas que hagan llegar la información al usuario de una manera más comprensible. Estas visualizaciones se han adaptado a diversos perfiles con diferentes roles en la producción permitiendo obtener la información necesaria para cada uno.

Por último, en vista a las necesidades que se han ido planteando por el camino, se propone en el Capítulo 7 las líneas futuras de trabajo.

7. Líneas futuras de trabajo

Siguiendo la línea de trabajo sobe la que va el proyecto, se plantea como trabajo futuro las siguientes tareas:

- Completar la base de datos de imágenes capturadas con la Trevista con más muestras de defectos. Como se ha mencionado en el Capítulo 5.2.8, el número de muestras a capturar con la Trevista no era equilibrado, por lo que el resultado del entrenamiento no era fiable en la clasificación de cada defecto. Tras los resultados obtenidos con el clasificador binario y las imágenes obtenidas con este sistema, se quiere entrenar un modelo que clasifique el defecto según su naturaleza. Con esto se quiere dar un paso más en verificar que el sistema de captura Trevista ofrece verdaderas ventajas en la clasificación de los defectos y mejora la precisión respecto a las imágenes convencionales.
- Se plantea la mejora de las herramientas de visualización adecuándose más a las necesidades reales que tengan los trabajadores de este sector, así como la creación de más herramientas para otros roles. A pesar de haber diseñado las visualizaciones teniendo en cuenta las herramientas que se emplean en la actualidad en entornos industriales, se quiere recibir un *feedback* sobre cómo de útil y adecuada es la herramienta desde el punto de vista de usabilidad, comodidad, etc.
- Aplicar los conocimientos adquiridos en la configuración de las arquitecturas de *Deep Learning* para entrenar modelos que permitan detectar defectos en otros materiales.
- Validar mediante prototipo el impacto de esta arquitectura en un entorno de producción real mediante estudio de indicadores de rendimiento (KPIs), rechazos, reducción de chatarra, etc.

Referencias bibliográficas

[1] J. Lee, H.-A. Kao, and S. Yang, “Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment,” *Procedia CIRP*, vol. 16, pp. 3–8, Jan. 2014.

[2] “UNESID.” [Online]. Available: https://unesid.org/. [Accessed: 16-May-2018].

[3] J. L. Enríquez Berciano, E. Tremps Guerra, D. Fernández Segovia, and S. de Elío de Bengy, “Monografías sobre Tecnología del Acero. Parte 1,” *Univ. Politec. Madrid*, p. 244, 2009.

[4] Sirius Advanced Cybernetics (SAC), “Machine Vision for Industry. Trevista.” [Online]. Available: https://www.sac-vision.net/en/Products/trevista.php. [Accessed: 16-May-2018].

[5] T. Ojala, M. Pietikäinen, and D. Harwood, “A comparative study of texture measures with classification based on featured distributions,” *Pattern Recognit.*, vol. 29, no. 1, pp. 51–59, Jan. 1996.

[6] T. Ojala, M. Pietikäinen, and T. Mäenpää, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, 2002.

[7] M. Chu, R. Gong, S. Gao, and J. Zhao, “Steel surface defects recognition based on multi-type statistical features and enhanced twin support vector machine,” *Chemom. Intell. Lab. Syst.*, vol. 171, pp. 140–150, Dec. 2017.

[8] M. Xiao, M. Jiang, G. Li, L. Xie, and L. Yi, “An evolutionary classifier for steel surface defects with small sample set,” *EURASIP J. Image Video Process.*, vol. 2017, no. 1, p. 48, Dec. 2017.

[9] C. Y. Hsu, B. S. Ho, L. W. Kang, M. F. Weng, and C. Y. Lin, “Fast vision-based surface inspection of defects for steel billets,” in *2016 IEEE International Conference on Consumer Electronics-Asia, ICCE-Asia 2016*, 2017, pp. 1–2.

[10] A. O. M. Luiz, L. C. P. Flavio, and E. M. A. Paulo, “Automatic detection of surface defects on rolled steel using Computer Vision and Artificial Neural Networks,”

IECON 2010 - 36th Annu. Conf. IEEE Ind. Electron. Soc., no. c, pp. 1081–1086, 2010.

[11] E. Qu, Y. Cui, S. Xu, and H. Sun, “Saliency defect detection in strip steel by improved Gabor filter,” vol. 45, no. 10, pp. 12–17, Oct. 2017.

[12] E. A. Kholief, S. H. Darwish, and M. N. Fors, “Detection of Steel Surface Defect Based on Machine Learning Using Deep Auto-encoder Network,” pp. 218–229, 2017.

[13] M. B. Tayel and M. E. El-Bouridy, “ECG images classification using artificial neural network based on several feature extraction methods,” in *2008 International Conference on Computer Engineering & Systems*, 2008, pp. 113–115.

[14] Y. Wang, H. Xia, X. Yuan, L. Li, and B. Sun, “Distributed defect recognition on steel surfaces using an improved random forest algorithm with optimal multi-feature-set fusion,” *Multimed. Tools Appl.*, pp. 1–30, Oct. 2017.

[15] S. Zhou, D. Liang, and Y. Wei, “Automatic detection of metal surface defects using multi-Angle lighting multivariate image analysis,” in *2016 IEEE International Conference on Information and Automation, IEEE ICIA 2016*, 2017, pp. 1588–1593.

[16] R. Azhar, D. Tuwohingide, D. Kamudi, Sarimuddin, and N. Suciati, “Batik Image Classification Using SIFT Feature Extraction, Bag of Features and Support Vector Machine,” *Procedia Comput. Sci.*, vol. 72, pp. 24–30, Jan. 2015.

[17] L. Lenc and P. Král, “Automatic face recognition system based on the SIFT features,” *Comput. Electr. Eng.*, vol. 46, pp. 256–272, Aug. 2015.

[18] A. Shmilovici, “Support Vector Machines,” in *Data Mining and Knowledge Discovery Handbook*, Boston, MA: Springer US, 2009, pp. 231–247.

[19] I. Steinwart and A. Christmann, *Support vector machines*. Springer, 2008.

[20] R. Gong, C. Wu, and M. Chu, “Steel surface defect classification using multiple hyper-spheres support vector machine with additional information,” *Chemom. Intell. Lab. Syst.*, vol. 172, pp. 109–117, Jan. 2018.

[21] H. Hu, Y. Liu, M. Liu, and L. Nie, “Surface defect classification in large-scale strip steel image collection via hybrid chromosome genetic algorithm,” *Neurocomputing*, vol. 181, pp. 86–95, 2016.

[22] H. Huang, C. Hu, T. Wang, L. Zhang, F. Li, and P. Guo, “Surface defects detection for mobilephone panel workpieces based on machine vision and machine learning,” in *2017 IEEE International Conference on Information and Automation (ICIA)*, 2017, pp. 370–375.

[23] M. Zoheir, M. Abdelkrim, B. Djalil, and B. Adel, “Identification approaches for steel strip surface defects in hot rolling Process,” 2018.

[24] W. Cui, K. Wang, Q. Zhang, and P. Zhang, “A RECOGNITION ALGORITHM TO DETECT PIPE WELD DEFECTS,” vol. 3651, pp. 1969–1975, 2017.

[25] F. Riaz, K. Kamal, T. Zafar, and R. Qayyum, “An inspection approach for casting defects detection using image segmentation,” in *2017 International Conference on Mechanical, System and Control Engineering (ICMSC)*, 2017, pp. 101–105.

[26] K. Song and Y. Yan, “A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects,” *Appl. Surf. Sci.*, vol. 285, no. PARTB, pp. 858–864, Nov. 2013.

[27] K. Song, S. Hu, and Y. Yan, “Automatic recognition of surface defects on hot-rolled steel strip using scattering convolution network,” *J. Comput. Inf. Syst.*, vol. 10, no. 7, pp. 3049–3055, 2014.

[28] J. Bruna and S. Mallat, “Classification with scattering operators,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, no. 2, pp. 1561–1566, 2011.

[29] S. Pan, T.-Y. Hung, and L.-T. Chia, “Using material classification methods for steel surface defect inspection,” in *2016 IEEE 25th International Symposium on Industrial Electronics (ISIE)*, 2016, pp. 40–45.

[30] Claudia Villalonga, “Técnicas de Inteligencia artificial.” [Online]. Available: https://campusvirtual.unir.net/portal/site/PER3-416-7832/tool/2df2120c-b502-4d95-976a-ecf3b90131bd. [Accessed: 24-May-2018].

[31] S. Zhou, Y. Chen, D. Zhang, J. Xie, and Y. Zhou, “Classification of surface

defects on steel sheet using convolutional neural networks," *Mater. Tehnol.*, vol. 51, no. 1, pp. 123–131, 2017.

[32] R. Ren, T. Hung, and K. C. Tan, "A Generic Deep-Learning-Based Approach for Automated Surface Inspection," *IEEE Trans. Cybern.*, vol. 48, no. 3, pp. 929–940, Mar. 2018.

[33] J. Masci, U. Meier, D. Ciresan, J. Schmidhuber, and G. Fricout, "Steel defect classification with Max-Pooling Convolutional Neural Networks," *Proc. Int. Jt. Conf. Neural Networks*, 2012.

[34] D. Soukup and R. Huber-Mork, "Convolutional Neural Networks for Steel Surface Defect Detection from Photometric Stereo Images," *Isvc 2014*, pp. 668–677, 2014.

[35] Caffe2, "Caffe2 | A New Lightweight, Modular, and Scalable Deep Learning Framework." [Online]. Available: https://caffe2.ai/. [Accessed: 24-May-2018].

[36] Microsoft, "Microsoft Cognitive Toolkit." [Online]. Available: https://www.microsoft.com/en-us/cognitive-toolkit/. [Accessed: 24-May-2018].

[37] MathWorks, "MATLAB - El lenguaje del cálculo técnico - MATLAB & Simulink." [Online]. Available: https://es.mathworks.com/products/matlab.html. [Accessed: 24-May-2018].

[38] Chainer, "Chainer: A flexible framework for neural networks." [Online]. Available: https://chainer.org/. [Accessed: 24-May-2018].

[39] MXNet, "MXNet: A Scalable Deep Learning Framework." [Online]. Available: https://mxnet.incubator.apache.org/. [Accessed: 24-May-2018].

[40] Y. Jia *et al.*, "Caffe: Convolutional Architecture for Fast Feature Embedding," Jun. 2014.

[41] PyTorch, "PyTorch." [Online]. Available: https://pytorch.org/. [Accessed: 24-May-2018].

[42] Google, "TensorFlow." [Online]. Available: https://www.tensorflow.org/. [Accessed: 24-May-2018].

[43] M. Franceschini and Monica, “How to maximize the value of big data with the open source SpagoBI suite through a comprehensive approach,” *Proc. VLDB Endow.*, vol. 6, no. 11, pp. 1170–1171, Aug. 2013.

[44] “Apache Cassandra.” [Online]. Available: http://cassandra.apache.org/. [Accessed: 19-Jun-2018].

[45] Apache, “Apache HBase – Apache HBase™ Home.” [Online]. Available: https://hbase.apache.org/. [Accessed: 19-Jun-2018].

[46] “MongoDB for GIANT Ideas | MongoDB.” [Online]. Available: https://www.mongodb.com/. [Accessed: 19-Jun-2018].

[47] Neo Technology, “The Neo4j Graph Platform – The #1 Platform for Connected Data.” [Online]. Available: https://neo4j.com/. [Accessed: 19-Jun-2018].

[48] F. Chang *et al.*, “Bigtable: A Distributed Storage System for Structured Data.” 2006.

[49] “NEU_surface_defect_database.” [Online]. Available: http://faculty.neu.edu.cn/yunhyan/NEU_surface_defect_database.html. [Accessed: 27-Jun-2018].

[50] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Adv. Neural Inf. Process. Syst.*, pp. 1–9, 2012.

[51] A. Krizhevsky, “One weird trick for parallelizing convolutional neural networks,” 2014.

[52] S. Wu, S. Zhong, and Y. Liu, “Deep residual learning for image steganalysis,” *Multimed. Tools Appl.*, pp. 1–17, 2017.

[53] “HTML5 JavaScript Charts | CanvasJS.” [Online]. Available: https://canvasjs.com/. [Accessed: 05-Jul-2018].

[54] A. Loizaga and J. Sertucha, “Defectos metalúrgicos generados por la presencia de gases en el metal fundido,” *An. Química*, vol. 2, no. June 2014, pp. 111–119, 2008.

[55] Y. J. Zhao, Y. H. Yan, and K. C. Song, “Vision-based automatic detection of steel surface defects in the cold rolling process: considering the influence of industrial liquids and surface textures,” *Int. J. Adv. Manuf. Technol.*, vol. 90, no. 5–8, pp. 1665–1678, 2017.

[56] T. Ahonen, A. Hadid, and M. Pietik\\"ainen, “Face recognition with local binary patterns,” *Eur. Conf. Comput. Vis.*, pp. 469–481, 2004.

[57] A. Saito *et al.*, “A novel method for morphological pleomorphism and heterogeneity quantitative measurement: Named cell feature level co-occurrence matrix,” *J. Pathol. Inform.*, vol. 7, no. 1, p. 36, 2016.

[58] Claudia Villalonga, “Técnicas de Inteligencia Artificial.” [Online]. Available: https://campusvirtual.unir.net/portal/site/PER3-416-7832/tool/2df2120c-b502-4d95-976a-ecf3b90131bd. [Accessed: 10-Jul-2018].

Anexo 1. Arquitectura de almacenamiento propuesta

La información a almacenar es sumamente importante para la empresa, por lo que se plantea la existencia de al menos dos copias de la base de datos, actualizadas inmediatamente después de cada cambio.

Para ello es necesario crear un **respaldo con una copia de seguridad dinámica** a dos secundarios. Con el comando **replSet** se configura la replicación. Se debe especificar un nombre de conjunto de réplica como argumento para este conjunto. Todos los hosts en el conjunto de réplicas deben tener el mismo nombre de conjunto. Para ello hay que generar los dos **config server** secundarios. La creación de estos config server se explicará en el último punto, en el que se crea la arquitectura de la base de datos.

```
mongod --replSet <nombre de conjunto de réplicas>/<dirección de otro host>

mongod --replSet data --port 27117
mongod --replSet data --port 27217
```

Se requiere que se establezcan **perfiles en el acceso a los datos**. Por ello, se crea un administrador y contraseña para el acceso.

Crear usuario administrador y adjudicar derechos.

```
admin = db.getSiblingDB("admin")
admin.createUser(
{
user: "admin",
pwd: "Password1",
roles: [{ role: "userAdminAnyDatabase", db: "admin"}, {role: "clusterAdmin", db:
"admin"}, { role: "readAnyDatabase", db: "admin"}, { role: "readWriteAnyDatabase",
db: "admin"}, { role: "dbAdminAnyDatabase", db: "admin"}]
}
)
```

Se para el servidor y se lanza con la autenticación, ahora ya se tendrá un superusuario y será necesario la credencial a no ser que se trabaje en local.

```
mongod --auth --port 27017 --dbpath /data/db1
security:
authorization: enabled
```

Se entra como administrador a MongoDB, se selecciona la base de datos en la que se vaya a crear el usuario y se crea un usuario con el rol deseado.

```
db.getSiblingDB("admin").auth("admin", "Password1")
use data
db.createUser({
user: "usuario0002",
pwd: "ghT67gHK",
roles: [ { role: "dbadmin", db: "Data "},
})

})
```

Se **restringe los servidores** que tienen conexión a internet el acceso a la base de datos. En los ficheros de configuración de los servidores mostrados en el siguiente punto, se configura la restricción en las siguientes líneas:

```
net:
   bindIp: 127.0.0.1
   port: 28017
```

Se elige el IP de la interfaz de red que quiera que escuche, solo se podrían conectar clientes que estén en la misma máquina que el servidor.

Se estima que en un futuro se manejará una gran cantidad de datos, por lo que se requiere **una arquitectura escalable**. Para ello se utilizará ***sharding***.

Primero se generará el **replica set** de los **config server**. Para ello se generan los 3 **ficheros de configuración** de los config server y se incluyen en la carpeta etc.

Los ficheros son los siguientes: cfgsvr1.cfg, cfgsvr2.cfg y cfgsvr3.cgf, con puertos 28017, 28117 y 28217. Se muestra a continuación el fichero cfgsvr1.cfg.

```
security:
   keyFile: "C:/MongoDB/etc/keyfile.txt"
net:
   bindIp: 127.0.0.1
   port: 28017
sharding:
   clusterRole: configsvr
replication:
   replSetName: "replicaConfSvr"
storage:
   dbPath: "C:/MongoDB/data/cluster/cfgsvr1/"
systemLog:
   destination: file
   path: "C:/MongoDB/log/cfgsvr1.log"
   logAppend: true
```

A continuación, se inician los 3 config servers en modo administrador:

```
>C:\Program Files\MongoDB\Server\3.4\bin>mongod --config
"C:\MongoDB\etc\cfgsvr1.cfg" --install --serviceName Mongocfgsvr1 --
serviceDisplayName "MongoDB Config Server 1"

>C:\Program Files\MongoDB\Server\3.4\bin>mongod --config
"C:\MongoDB\etc\cfgsvr2.cfg" --install --serviceName Mongocfgsvr2 --
serviceDisplayName "MongoDB Config Server 2"

>C:\Program Files\MongoDB\Server\3.4\bin>mongod --config
"C:\MongoDB\etc\cfgsvr3.cfg" --install --serviceName Mongocfgsvr3 --
serviceDisplayName "MongoDB Config Server 3"
```

Se inicializa el replica set de los config server. Por ejemplo, se realizará la conexión con el primero. (Antes iniciarlos desde Servicios de Windows.)

```
>mongo localhost:28017
```

Seguidamente, se configuran los config servers

```
>rs.initiate(
         {
         _id: "replicaConfSvr",
         configsvr: true,
         members: [
         { _id : 0, host : "localhost:28017" },
         { _id : 1, host : "localhost:28117" },
         { _id : 2, host : "localhost:28217" }
         ]
         }
)
```

Se va a generar el primer ***shard*** replica set. Se debe tener el fichero ***keyfile*** accesible. Se configuran los 3 ficheros de configuración que son:

Shard1pri.cfg, shard1sec.cfg, shard1arb.cfg y los 3 puertos, 29017, 29117, 29217. Se muestra el fichero de configuración shard1pri.cfg.

```
security:
   keyFile: "C:/MongoDB/etc/keyfile.txt"
net:
   bindIp: 127.0.0.1
   port: 29217
sharding:
   clusterRole: shardsvr
replication:
   replSetName: "replicaShard1"
storage:
   dbPath: "C:/MongoDB/data/cluster/shard1arb/"
systemLog:
   destination: file
   path: "C:/MongoDB/log/shard1arb.log"
   logAppend: true
```

Se iniciarán ahora los 3 server *shard*:

```
>mongod --config "C:\MongoDB\etc\shard1pri.cfg" --install --serviceName Mongosh1pri
--serviceDisplayName "MongoDB Shard 1 Primary"

>mongod --config "C:\MongoDB\etc\shard1sec.cfg" --install --serviceName Mongosh1sec
--serviceDisplayName "MongoDB Shard 1 Secondary"

>mongod --config "C:\MongoDB\etc\shard1arb.cfg" --install --serviceName Mongosh1arb
--serviceDisplayName "MongoDB Shard 1 Arbitrary"
```

Ahora se realizará la conexión al interfaz localhost de uno de los servers *shard* para hacer el replicaset de este *shard*. Se configurará también cual es el primario, el secundario y el árbitro con *priority*.

```
>mongo localhost:29017
rs.initiate(
        {
        _id: "replicaShard1",
        members: [
        { _id : 0, host : "localhost:29017", priority : 20 },
        { _id : 1, host : "localhost:29117" , priority : 10 },
        { _id : 2, host : "localhost:29217", arbiterOnly : true }
        ]
        }
)
```

Por último, se deben **configurar los routers**. Tienen que tener el fichero *keyfile* accesible, se debe generar el fichero de configuración router1.cfg, y arrancar mongos. Se muestra a continuación el fichero de configuración del router:

```
security:
   keyFile: "C:/MongoDB/etc/keyfile.txt"
net:
   bindIp: 127.0.0.1
   port: 27017
sharding:
   configDB: replicaConfSvr/localhost:28017,localhost:28117,localhost:28217
systemLog:
   destination: file
   path: "C:/MongoDB/log/router1.log"
   logAppend: true

>mongos --config "C:\MongoDB\etc\router1.cfg" –install --serviceName Mongorouter --
serviceDisplayName "MongoDB Router"
```

Hasta aquí se tiene el router que se encarga de todas las consultas y se le ha dicho dónde está la configuración del cluster. Ahora se le indica donde están los datos y los shard. Para ello se le añade el *shard* al *cluster*. Primero se inicia sesión con admin.

```
>db.getSiblingDB("admin").auth("admin", "Password1")
>sh.addShard("replicaShard1/localhost:29017")
```

Finalmente, se debe habilitar la característica de *sharding* sobre la base de datos y sobre las collections a dividir:

```
sh.enableSharding("data")
sh.shardCollection("data.sensor",  {"id":1} )
```

Por último, se pueden consultar las bases de datos y las colecciones del *sharding* con:

```
>show dbs
>show collections
```