Informática y comunicaciones

ies azarquiel

Curso especialización en inteligencia artificial y big data

Tema 2: Estructura y Herramientas de Sistemas Inteligentes

Sebastián Rubio Valero

Septiembre 2025

ARTIFICIAL INTELLIGENCE

1 Desarrollo de aplicaciones con IA

La IA a menudo se confunde con la ciencia de datos, los macrodatos y la minería de datos. La Figura 1.5 muestra las relaciones entre la IA, el aprendizaje automático, el aprendizaje profundo, la ciencia de datos y las matemáticas. Tanto las matemáticas como la ciencia de datos están

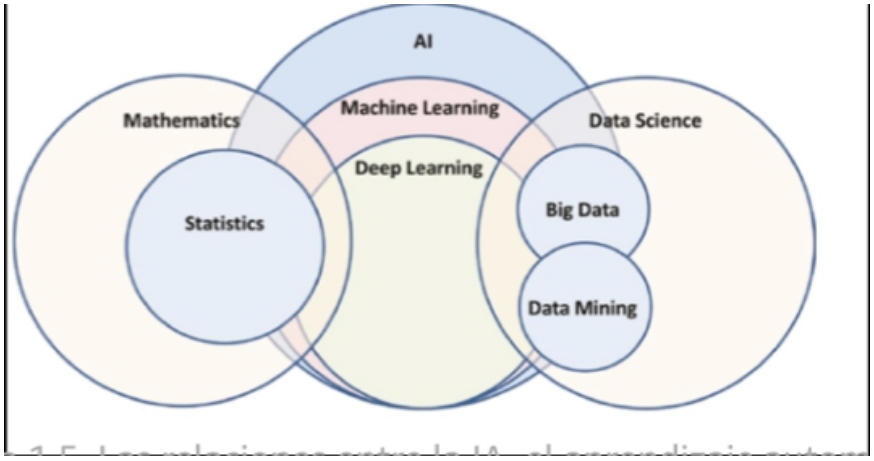

Figure 1: Enter Caption

Como se puede ver en la gráfica diferentes tecnologías tiene su base en la Inteligencia Artificial, pero no son las únicas. Existen otros sectores, donde se aplican otras tecnologías de I.A En el desarrollo de una aplicación empresarial convencional pasa por diferentes fases hasta que se pone en producción. Por regla general todos los proyectos se desarrollan pasando por las mismas fases. Aunque se pueden diferenciar algunas etapas en función de la metodología que usan y las escala del proyecto. De esta forma proyectos muy grandes o sensible utilizan muchas más etapas que los proyectos empresariales de menor envergadura. Sin embargo cualquiera de ellos se diferencia entre, otras cosas, las fases que son necesarias para la realización de un proyecto de Inteligencia Artificial La programación de inteligencia artificial puede considerarse como otro paradigma de programación. Muchos de los algoritmos más existosos de la IA no funcionan en problemas del mundo real En el siguiente esquema se aprecia entre las diferentes aplicaciones de la IA.

Lenguajes de programación en la I.A

La elección del lenguaje de programación para I.A es un factor decisivo para el éxito del proyecto. Para eso hay que conocer características del lenguaje. En la página web https://www.tiobe.com/tiobe-index/ podemos analizar un ranking de los lenguajes de programación. Hoy en día se utilizan muchos lenguajes de programación de inteligencia artificial. En el siguiente listado vemos los lenguajes más utilizados junto las librerias de mayor uso.

Python

Es el lenguaje dominante hoy en día. Se caracteriza por:

- Sintaxis simple y legible
- Comunidad activa y enorme ecosistema de librerías
- Ideal para prototipado rápido
- Multiplataformaitemize

Existe muchas librerías especializadas en inteligencia artificial y aprendizaje automático, muchas de ellas escritas en el lenguaje C/C++ las más destacadas son:

- TensorFlow,DeepLearning, redes neuronales
- PyTorch Alternativa a TensorFlow
- scikit-learn Machine Learning clásico (regresión, árboles, clustering)
- Keras API de alto de nivel para Deep Learning
- NLTK y spaCy, Procesamiento de Lenguaje Natural (PLN)
- OpenCV Visión por computadora

Para Big Data las más utilizadas son:

- PySpark API para Apache Spark
- Dask, procesamiento paralelo
- Pandas, análisis de datos (no distribuido, pero muy usuado)
- Vaex Dataframes para datasets muy grandes

Java

Es muy usado en Big Data y en algunas aplicaciones en IA. Se caracteriza por:

- Lenguaje compilado, robusto y portable
- Buen rendimiento y manejo de concurrencia
- Gran uso de empresas y entornos legacy

Las librerías más utilizadas en Inteligencia Artificial son:

- Deeplearning4j, deep learning en Java
- Weka, Herramienta de ML clásica con GUI y API.
- MOA , análisis de flujos de datos

Para Big Data son librerías muy utilizadas

- Apache Hadoop, Marco para procesamiento distribuido de datos
- Apache Spark, procesamiento en memoria y distribuido
- Apache Flink, procesamiento en tiempo real
- Kafka, mensajería y flujos de datos (streaming)

R

Es un lenguaje muy enfocada al análisis estadístico y visualización. Destaca por:

- Diseñado para análisis estadístico y visualización
- Ampliamente usado por científicos de datos y académicos
- Interactivo y orientado a análisis exploratorio

Las librerías más destacadas para I.A son:

- caret, Machine Learning unificado
- randomForest, xgboost, en modelos predictivos
- nnet, keras (con backend en TensorFlow), en redes neuronales

Estas otras librerías son para Big Data

- data.table, para procesamiento eficiente en memoria
- sparklyr, conexión con Apache Spark
- dplyr, transformación de datos con gramática elegante

Scala

Es muy utilizado en Big Data , especialmente con Spark . Se caracteriza por:

- Funcional + Orientado a Objetos
- Corre en la JVM, pero es más conciso
- Base de Apache Spark

Las librerías más utilizadas para Big Data son:

- Apache Spark (nativo en Scala)
- Akka, computación concurrente y reactiva
- Kafka Streams, para procesamiento de flujos.

Para I.A es más limitado, utiliza las misma librerias que Java, descatar Breeze

Julia

Es un lenguaje de nueva creación, tiene ciertas características que los hacen apropiado para I.A y Big Data Se caracteriza por:

- Alto rendimiento
- Orientado a cálculos numéricos, álgebra lineal y simulaciones
- Sintaxis amigable similar a MATLAB o PHYTON

Las librebrería más populares para I.A son

- Flux.jl, para Deep Learning
- MLJ.jl, plataform unificada para ML

Para Big Data tenemos:

- JuliaDB, Manejo de datos grandes
- Dagger.jl, Computación distribuida

C/C++

Las librerias que podemos utilizar son:

- Google TensorFlow
- Caffe
- Herramientas congnitivas de Microsoft(CNTK)
- La librería MLPACK
- La libreria SHARK
- OpenNN

SQL/VARIANTES

Es especialmente para Big Data, para acceso y manipulación de datos. Se caracteriza por:

- Lenguajes declarativo para bases de datos relacionales
- Utilizado en casi todos los pipelines de datos

Las librerías más utilizadas son:

- Apache Hive, SQL sobre Hadoop
- Presto/Trino, SQL distribuiodo en Big Data
- BigQuery (Google), Athena (AWS)- SQL en la nube
- SQLAlchemy (Python), ORM para bases de datos

Lenguaje	IA	Big Data	Comentario
Python	★★★★★	★★★★	El más versátil y dominante en IA
Java	★★	★★★★★	Muy fuerte en entornos Big Data corporativos
R	★★★★	★★	Excelente para estadística; menos usado en Big Data real
Scala	★★	★★★★	Spark lo mantiene relevante
Julia	★★★	★★	Muy prometedor, aún emergente
SQL	–	★★★★★	Imprescindible para acceder a datos
C/C++	★★★	★	Para IA de alto rendimiento (backends)

2 Inteligencia Artificial EDGE/CLOUD

Una inteligencia artificial que se ejecuta en una máquina local se le conoce con el nombre de se les conoce con el nombre de IA Edge y las que se ejecutan en la nube se les conocen IA Cloud. Los tres principales proveedores de servicios IA Cloud son:

- Amazon AWS Machine Learning
- Microsoft Azure
- Google Cloud Platform
- IBM Cloud
- Alibaba Cloud

Las ventajas de Edge AI son la baja latencia, es en tiempo real y seguro. En estos sistemas es necesario un harware especial entre los más populares son:

- Microcontroller-based AI
- Raspberry Pi-bases AI:
- Google Edge TPU TensorFlow Processing Unit:
- NVidia Jetson GPU-based AI:
- Intel and Xilinx-based AI:

- BeagleBone AI
- 96Boards AI
- Baidu EdgeBoard

En definitiva los lenguajes deben tener librerías que den cobertura a los diferentes temas de la I.A:

- Redes neuronales
- Aprendizaje
- Procesamiento natural
- Procesamiento de textos
- Sistemas expertos
- Procesamiento matemático
- Visión por computador

Alpha Go fué uno de los hitos de la inteligencia artificial, su objetivo era ganar el máximo de partidas del juego de mesa GO. Se utilizaron 1920 CPU'S y 280 GPU'S y algunas unidades de procesamiento de tensor (TPU) de Google.

Lo último en la tecnología de IA es la IPU (Unidad de procesamiento inteligente) Mk2 tiene tiempos de entrenamiento 16 veces más rápido para la clasificación de imágenes que la GPU de NVIDIA, y 12 veces más económica.

Otra IA que irrumpió con fuerza GPT-3 (Generative Pre-trained Transformer 3), es un modelo de predicción de lenguaje, una red neuronal de aprendizaje profundo. Se entrenó con miles de millones de información al rastrear Internet para esto utilizarón Wrapper. Tiene 96 capas y la friolera de 175 mil millones de parámetros. Una estimación cuesta 1 dolar entrenar 1000 parámetros. A raiz de GPT-3 nacen DALL-E y CLIP que son dos redes neuronales muy interesantes.

AlphaFold AI hizo un gran avance en el plegamiento de proteínas. En el 2021, Google anunció el desarrollo de un nuevo modelo llamado Switch Transformer que contiene 1.6 billones de parámetros. Puede entender 101 idiomas. WuDao es un IA de procesamiento de lenguaje natural, se desarrolló con la ayuda de mas 100 científicos.

****** MIRAR ESTAS URL Informe del índice AI, Stanford: https://aiindex.stanford.edu/report/, otro enlace ******www.stateof.ai
En breve tendremos modelos con 10 billones de parámetros.

3 Desarrollo de Software

El desarrollo de un software convencional consta de las siguientes fases:

- Definición del problema
- Especificación de requisitos (funcionales y no funcionales)
- Recolección y preparación de datos
- Selección del enfoque y diseño del modelo
- Codificación
- Entrenamiento del modelo
- Evaluación y validación
- Despliegue
- Mantenimiento y mejora continua

3.1 Definición del problema

El objetivo es comprender el problema que se quiere resolver y definir claramente los requisitos del sistema.
Esta primera fase requiere las siguientes tareas

- Identificación de stakeholders
- Análisis del dominio del problema
- Determinación de objetivos específicos
- Establecimiento de métricas de éxito

3.2 Recolección y preparación de datos

Obtener datos relevante y convertirlos en una forma utilizable por el sistema. Esta fase es muy importante para el entrenamiento del modelo, y el éxito del módelo viene dado en gran medida por los datos con los que se entrena. Las principales tareas son:

- Recolección (fuentes internas, sensores, scraping, APIS, etc)
- Limpieza (eliminar valores nulos, duplicados, outliers)
- Etiquetar (en problemas supervisados)
- Dividir en conjuntos de entrenamiento, validación y prueba

3.3 Selección del enfoque y diseño del modelo

Hay que elegir el tipo IA adecuada (ML, DL, simbólica, híbrida) y diseñar la arquitectura. El diseño del modelo puede abarcar diferentes técnicas de IA. En muchos modelos avanzados se descomponen en diferentes etapas que se desarrollan con diferentes algoritmos o técnicas.
Las tareas que comprenden son:

- Decisión entre aprendizaje supervisado, no supervisado, por refuerzo, etc
- Selección de algoritmos o arquitecturas (rede neuronales, árboles de decisión, transformers,etc)
- Diseño del pipline de entrenamiento y predicción

3.4 Entrenamiento del modelo

El objetivo es ajustar los parámetros del en base a los datos de entrenamiento. Esta etapa puede requerir mucho tiempo. Por regla general requiere las siguientes tareas:

- Configuración de hiperparámetros
- Entrenamiento iterativo
- Monitorización de métricas de rendimiento
- Validación cruzada para evitar overfitting

3.5 Evaluación y validación

Asegurar que el modelo generaliza bien y cumple con los criterios definidos. Si el resultado de la evaluación no es el esperado o deseado hay revisar las etapas anteriores. En ocasiones hay revisar desde la primera fase, y en otras desde la preparación de datos o entrenamiento. Las diferentes tareas son:

- Evaluación con datos de prueba
- Comparación con benchmarks (rendimiento, performace)
- Análisis de errores
- Validación de sesgos y equidad

3.6 Despliegue

Ahora toca integrar el modelo en un entorno de producción que pueda ser usado por los usuarios o el sistema. En muchas ocasiones ya existe software empresarial, y se quiere mejorar el servicio con una IA, de modo que el despliegue requiere menos esfuerzo que si fuese una aplicación completamente nueva en la que la IA tiene un papel muy importante. Las diferentes tareas son:

- API o servicio en la nube
- Asegurar la escalabilidad y eficiencia
- Monitorizar en tiempo real
- Implementación de actualizaciones y rollback si es necesario

3.7 Mantenimiento y mejora continua

El objetivo es adaptar el sistema a cambios en los datos, entorno o necesidades. Los modelos , por regla general, se entrenan con datos de meses/años pasados. Por lo que en cambio brusco en los datos, normas, leyes, etc obliga a volver a entrenar el modelo. Consta de las siguientes tareas:

- Reentrenamiento con nuevos datos
- Detección de degradación del rendimiento (drift)
- Feedback loop con usuarios
- Incorporación de nuevas funcionalidades

3.8 Evaluación ética y cumplimiento

El objetivo es asegurar que el sistema cumple con principios éticos y regulaciones. Las siguientes tareas son las más comunes:

- Revisión de explicabilidad, privacidad y sesgos
- Conformidad con normativa (GDR,IA Act,etc)
- Auditorias independientes

Para que esta sección quede más clara veamos un ejemplo.

4 Importancia de lo Datos

En un proyecto de I.A la mayoría del tiempo está orientado a la mejora de los datos. El éxito de un proyecto de I.A, gran medida va venir determinado por la calidad de los datos y de su tamaño. Si los datos de partida no son los correctos el entrenamient0 de nuestros modelos serán fracaso o bien estarán muy sesgados. Debes saber que dentro de la programación de IA hay una rama conocida como cientifico de datos que abarca el problema del tratamiento de los datos.

4.1 Tamaño y calidad de los datos

Para la construcción de un DataSet de calidad debe pasar por los siguientes pasos:

- Recolección de datos
- Identificación de propiedades y etiquetado
- Estrategia de muestreo de datos
- División del dataset en: Entrenamiento,validación y prueba

La aplicación de estos pasos va a venir dado por el problema de IA que vamos a resolver. Por ejemplo, para la identificación de las propiedades, seleccionamos una o dos propiedades que tengan una fuerte correlación con las variables objetivo, luego posteriormente podemos añadir más propiedades mas propiedades. El tamaño del dataset va a depender del problema que se está intentando resolver, no hay una regla que nos diga cual debe ser el tamaño del dataset, sin embargo existen algunas recomendaciones, por ejemplo, para las redes neuronales se recomienda que el número de ejemplos para el entramiento sea superior en orden de magnitud al número de parámetros entrenables. Otro factor importante es la calidad del dataset, para ello existen tres aspectos de los datos que hay que valorar:

- Alta fiabilidad. Hace referencia a la frecuencia de errores en el etiquetado, al ruido de la propiedades y si las propiedades son relevantes a la hora de resolver el problema
- Representación de propiedades adecuado. Normalización si es necesario, tipo de datos,
- Minimizar el sesgo. Hay que asegurar que el dataset es representativo para el entrenamiento

4.2 Fuentes de datos y etiquetado

Los datos de entrenamiento pueden venir de diferente fuentes datos u orígenes,también puede derivarse de evento (dirección IP desde la que se realizó una consulta) o de diferentes atributos (datos demográficos). Los datos pueden encontrarse en diferentes ubicaciones. Las fuentes de datos pueden ser Onlin u Offline. Los modelos dinámicos son entrenados online, es decir el modelo está continuamente entrenando conforme el sistema va recibiendo datos. Por el contrario los modelos estáticos, son entrenados de modo offline, es decir el modelo se entrena con un conjunto de datos fijo, el modelo se esta utilizando durente un periodo tiempo y no se vuelve a entrenar hasta la obtención de nuevos datos. El correcto etiquetado de los datos dará como resultado un buen y rápido entrenamiento del modelo. Existe dos formas de etiquetar:

- Etiquetas directas, representa los resultados exactos que se quieren predecir. Pueden hacer referencia a eventos y son fáciles de identificar. Etiquetar atributos es más complejo ya que necesitamos información anterior.
- Etiquetas derivadas, representan resultados indirectos.

En los siguientes ejemplos se muestran ejemplos de etiquetas directas:

- Clasificación de imágenes, una persona etiqueta una imagen como "gato" o "perro". Por ejemplo, una foto en un dataset viene etiquetada con un "automovil"
- Un analista etiqueta manualmente reseñas como "positiva", "negativa" o "neutral". Ejemplo, "Me encantó este producto" la etiqueta es "positiva".
- Un médico etiqueta una imagen de rayos X como "neumonía".
- Un analista clasifica manualmente una transacción como "gasto personal" o "gasto empresarial"

- Un producto es etiquetado como "ropa", "electrónica", etc,
- un cliente califica un producto con 1 a 5 estrellas

En estos otros ejemplos tenemos etiquetas derivadas:

- Unas transacción es etiquetada como "fraudulenta" si fue revertida o marcada como sospechosa por el sistema bancario.
- Si un usuario hace clic repetidamente en ciertos productos, el sistema infiere que eso productos son "relevantes" para él.
- Un sistema detecta "riesgo de diabetes" al analizar niveles de glucosa y antecedentes familiares.
- El sistema infiere que un cliente tiene "perfil conservador" al observar su bajo riesgo en inversiones.
- Un sistema etiqueta a un cliente como "comprador recurrente" tras más de 5 compras en 30 días.

5 Muestreo y división de datos

Un problema que nos podemos encontrar en un dataset es el de un dataset no balanceado. Podemos encontrarnos con una mayor proporción de datos con una etiqueta o clase que otra, cuando nos enfrentamos a un problema de clasificación. De esta forma, tenemos dos tipos clases: la clase mayoritaria y la clase minoritaria. Si cogemos el ejemplo de los perros y gatos, será mayoritaria si tenemos muchos más perros que de gatos. Existe tres niveles de desequilibrio dependiendo de la proporción de clases minoritarias frente a las clases mayoritarias.

- Desequilibrio leve, la clase minoritaria = 20% - 40%
- Desequilibrio moderado, la clase minoritaria = 1%-20%
- Desequilibrio extremo, la clase minoritaria $< 1\%$

Existen técnicas para corregir el desequilibrio.

En la división de datos para entrenamiento y pruebas se suele hacer de forma aleatoria, eso suele funcionar en la mayoría de las ocasiones, pero si los datos se encuentran agrupados estaremos formando los grupos de forma incorrecta provocando de esta forma un sesgo en el aprendizaje. La solución pasa por dividir los grupos.

6 Convecciones y buenas prácticas

Dependiendo del lenguaje escogido para la etapa de codificación hay una serie de buenas prácticas, que en muchas ocasiones son comunes en muchos lenguajes.

La guía de referencia PEP8, es una guía de python de buenas práctica, esta guía la puedes encontrar en el material del curso. Es importante hacer una lectura de esta guía antes de empezar los ejemplos que estudiaremos en el curso.

Hay muchos enfoques en la tarea de aprender a programa I.A, pero en este curso nos vamos a utilizar el enfoque que consta de dos fases, que se ajusta al currículum :

- 1.- Proceso de aprendizaje de fundamentos de la I.A
- 2.- Proceso de aprendizaje de la programación

7 HERRAMIENTAS

Para trabajar de forma local es suficiente con un ordenador básico que se puede comprar a precio reducido. Para crear modelos más grandes necesitamos tener un GPU, lo que nos facilitará el tiempo de computo. Los ordenadores gaming son una buena opción porque disponen de GPU normalmente NVIDIA GeForce RTX . Otro modelo de ordeandores son aquellos que disponen de FPGA, son más caros. Los IPU de Graphore es otra opción que son más rápidos que las GPU. Tenemos a nuestra disposición herramietas de desarrollo que se pueden utilizar para la programación de I.A. La herramienta debe permitirnos crear entornos virtuales, con el fin de evitar conflictos entre los proyectos que estemos desarrollando localmente. La herramienta de Google Colab, nos permite elegir entre TPU y GPU para probar nuestros modelos.

8 Plataformas de Inteligencia Artificial

https://www.gartner.com/reviews/market/public-cloud-storage-services-worldwide Hoy en día las grandes tecnológicas son las que dominan el desarrollo de software en la nube (computación en la nube): Amazon, Microsoft, Google. son las más conocidas, por otro lado tenemos Wasabi,Alibaba Cloud, IBM Watson, Tencent Cloud. Los beneficios de la computación en la nube bajo demanda, pago por uso (sólo pagamos por los recursos que usamos), podemos destacar los siguientes beneficios:

1. Agilidad
2. Ahorro de costes
3. Elasticidad

4. Innovación rápida
5. Desplieguen en minutos
6. Transición rápida a producción

En este tema vamos a estudiar el lenguaje python, y las librerías más utilizadas para el desarrollo de modelos IA Vamos a utilizar dos entornos de programación, uno online (colab), y otro offline pyCharm junto con los cuadernos jupyter. En los temas de deepLearning va a ser necesario utilizar los equipos de clase por su mayor capacidad de cálculo. Instalaremos en los equipos el siguiente software:

- Anaconda, creación entornos virtuales
- Crear un entonrno virtual
- Desde Anaconda instalar Jupyter NoteBook

En un cuaderno de Jupyter NoteBook que tienes en el aula virtual se explica con ejercicios prácticas tres librerías que vas utilizar mucho durante el curso de especialización, numpy, matplotlib y pandas. Posteriormente durante el desarrollo del curso se verán otras librerías. Veremos librerías para la programación de redes neuronales como tensorflow, pytorch y keras.