Inteligencia Artificial

Apuntes de Clase — 21 de octubre de 2025

1st Kendall Rodríguez Camacho
Escuela de Ingenieria en Computación
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
Kenrodriguez@estudiantec.cr

***Abstract*—El presente documento contiene los apuntes de la clase del martes 21 de octubre de 2025, que cubren conceptos clave sobre los Modelos de Lenguaje Extensos (LLMs). Los apuntes explican cómo los LLMs representan el conocimiento mediante embeddings y espacios vectoriales, e introducen técnicas como Retrieval-Augmented Generation (RAG) y agentes inteligentes que amplían las capacidades de los LLMs con información externa y acciones autónomas.**

I. Introducción

Los Modelos de Lenguaje Extensos (LLMs) han transformado la interacción con la inteligencia artificial gracias a su capacidad para generar texto coherente, traducir idiomas, redactar código y analizar información compleja. Su funcionamiento se basa en la representación numérica de palabras y frases en espacios vectoriales, donde los embeddings capturan relaciones semánticas y contextuales.

Si bien los LLMs ofrecen capacidades sorprendentes, su conocimiento es limitado a los datos de entrenamiento y carecen de habilidades para actuar o buscar información activamente. Para superar estas limitaciones, se han desarrollado enfoques como Retrieval-Augmented Generation (RAG), que enriquece las respuestas con información externa relevante en tiempo real, y agentes inteligentes basados en LLMs, capaces de razonar, planificar y ejecutar tareas autónomas.

Este documento explora estos métodos y su evolución, mostrando cómo se puede pasar de modelos pasivos a sistemas que no solo comprenden el lenguaje, sino que también interactúan con el entorno, toman decisiones informadas y aplican conocimiento actualizado.

II. Aspectos del Curso

A. Calendario previsto para el resto del curso

La siguiente Tabla I muestra la planificación de las actividades restantes del curso, indicando las fechas y tareas correspondientes para cada semana.

Nota: En caso de que no se realice la visita a Microsoft, la clase *"Riesgos de IA"* se trasladaría al jueves de la semana 15, y el examen se aplicaría el jueves 20 de noviembre (semana 16).

TABLE I
Calendario previsto para el resto del curso

Semana	Martes	Jueves
12	Asignación de Tarea 04 (Agentes)	Clase de Quantization
13	Quiz 6, Terminar tema Quantization, Empezar Unsupervised Learning y PCA	Clase Unsupervised Learning y Entrega Proyecto 01
14	Revisión de Proyecto 01 de forma presencial (se sacará cita en un forms)	Entrega de Tarea 04 (Agentes) y continuar con revisión de Proyectos
15	Clase virtual, se asigna Proyecto 02 y Tarea 05 sobre Quantization	Revisión tarea 04 (Agentes) de forma Virtual
16	Clase Riesgos de IA	Visita a Microsoft
17	-	-
18	Examen presencial	Entrega de Proyecto 02

B. Asignación de Tarea 04

Se asigna la Tarea 04, la cual consiste en desarrollar un asistente conversacional que se desempeñe ante diferentes preguntas basadas en una base de documentos (Apuntes de clase realizados por los estudiantes hasta la fecha).

Se requiere implementar técnicas de recuperación y aumento de contexto (*RAG*) y comparar empíricamente los resultados con distintos esquemas de segmentación del texto.

La fecha de entrega está prevista para el jueves 6 de noviembre.

III. Fundamentos de los LLMs

A. Funcionamiento general

Los LLMs procesan los datos de entrada transformándolos en representaciones numéricas que describen características semánticas. Cada palabra, símbolo o carácter se convierte en una secuencia de valores numéricos mediante la tokenización, para luego ser procesados en redes neuronales profundas con millones o miles de millones de parámetros.

B. Del lenguaje al número

El texto debe convertirse en representaciones numéricas para ser interpretado por el modelo. El proceso de tokenización divide el texto en unidades mínimas llamadas *tokens* (palabras, subpalabras o caracteres), asignando a cada una un identificador único. Estos identificadores se transforman en vectores que los modelos utilizan como entrada.

TABLE II
TIPOS DE TOKENIZACIÓN

Tipo	Ejemplo	Ventaja principal
Por palabra	”Los modelos”	Simplicidad
Por carácter	“H”, “o”, “l”, “a”	Sin palabras fuera del vocabulario (OOV)
Subpalabra (BPE, WordPiece)	“Compu”, “tadora”	Equilibrio entre vocabulario y contexto
Byte-level	Código ASCII o UTF-8	Soporta cualquier símbolo o idioma
Espacios en blanco	”Hola”, ”mundo”	Rápido y simple

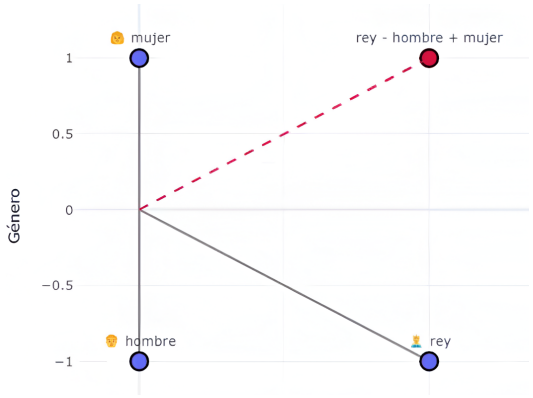

Fig. 1. Representación de tokens en un espacio bidimensional y ejemplo de operaciones semánticas.

IV. Representación del Conocimiento

A. Tokenización

En los LLMs se utilizan distintos enfoques de tokenización, cada uno con características particulares que afectan el rendimiento del modelo: La Tabla II resume estos enfoques, sus ejemplos y ventajas principales.

B. Representación en espacios vectoriales

Una vez tokenizado el texto, los identificadores se transforman en vectores dentro de un espacio continuo de alta dimensión. Las palabras con significados similares se ubican próximas entre sí, mientras que las palabras con significados distintos aparecen más alejadas.

Esto permite medir similitud semántica y realizar operaciones vectoriales, como analogías entre conceptos, suma o resta de vectores.

Tal como se muestra en la Figura 1, los vectores representan tokens proyectados en un espacio bidimensional para facilitar la comprensión, ilustrando relaciones semánticas entre ellos. Por ejemplo, la conocida analogía:

$$\text{Rey} - \text{Hombre} + \text{Mujer} \approx \text{Reina}.$$

En la práctica, estos vectores existen en un espacio de alta dimensión (n dimensiones), lo que permite capturar de manera más precisa la información semántica y contextual de las palabras.

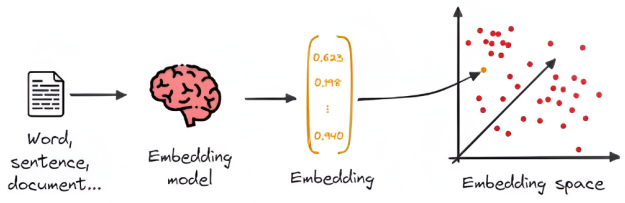

Fig. 2. Proceso de generación de embeddings desde palabra, oración o documento hasta el espacio vectorial.

C. Fórmulas de similitud entre vectores

Para comparar la similitud o distancia entre vectores, se utilizan diversas fórmulas matemáticas, entre las más comunes:

- Distancia euclidiana: mide la separación entre puntos en el espacio. Para dos vectores $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$:

$$d(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

- Similitud del coseno: mide el ángulo entre vectores y su orientación en el espacio, siendo la más usada en modelos de lenguaje:

$$\text{sim}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|}$$

D. Embeddings

Los *embeddings* son representaciones numéricas densas que capturan el significado semántico y las relaciones contextuales de palabras, oraciones o documentos completos. Estos vectores permiten comparar ideas, medir similitud y realizar operaciones semánticas en un espacio continuo de alta dimensión.

El proceso de generación de embeddings se puede resumir en los siguientes pasos:

- Entrada textual: La unidad de texto que se quiere representar, que puede ser una palabra, una oración o un documento completo.
- Modelo de embeddings: Un modelo que transforma la entrada en un vector numérico denso, capturando su significado y contexto.
- Embedding resultante: El vector que representa la entrada en el espacio continuo. Vectores cercanos indican conceptos semánticamente similares.
- Espacio de embeddings: El espacio vectorial donde cada embedding ocupa una posición. Este espacio permite medir similitudes y realizar búsquedas por proximidad.

Como se ilustra en la Figura 2, la figura representa el flujo de generación de embeddings: desde palabras, oraciones o documentos de entrada, pasando por el modelo de embeddings, hasta el vector resultante y su posición en el espacio de embeddings.

Además, la Figura 3 muestra un ejemplo simplificado de *sentence embeddings* proyectados en un plano bidimensional, donde frases con significado similar aparecen próximas entre sí.

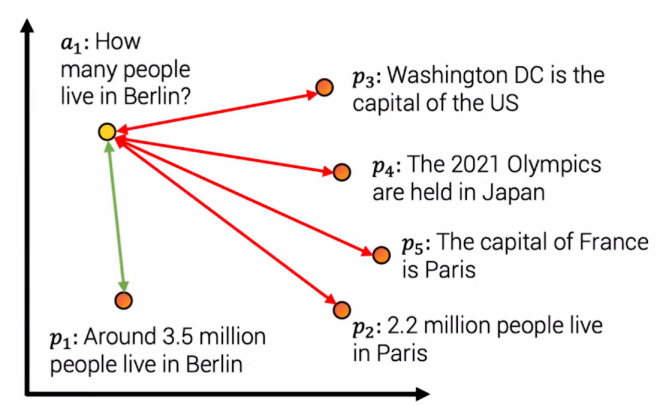

Fig. 3. Ejemplo de sentence embeddings en un espacio bidimensional

V. Capacidades y Limitaciones

A. Capacidades emergentes

Gracias a su entrenamiento masivo y al uso de arquitecturas basadas en *transformers*, los LLMs presentan capacidades como:

- Compresión textual: Interpretan el significado de palabras y frases según el entorno en el que aparecen.
- Generación coherente de texto: Pueden redactar, traducir o resumir información manteniendo estilo y consistencia.
- Razonamiento y planificación: Resuelven problemas, explican pasos y trazan estrategias simples.
- Aprendizaje en el prompt: Adaptan su comportamiento a partir de ejemplos dados en la misma conversación (*in-context learning*).
- Multitarea: Realizan traducción, clasificación, codificación, análisis o diálogo sin requerir entrenamiento adicional.

B. Limitaciones

A pesar de sus capacidades, los LLMs presentan limitaciones importantes:

- Alucinaciones: Pueden generar respuestas incorrectas o inventadas, especialmente cuando la información de entrada es ambigua o insuficiente.
- Memoria limitada: Olvidan información que se encuentra fuera del contexto actual, no recordando interacciones previas a menos que se almacenen externamente.
- Conocimiento estático: No tienen acceso a información posterior a su fecha de corte de entrenamiento, por lo que no están actualizados en tiempo real.
- Altos costos computacionales: Requieren hardware especializado y significativos recursos para entrenamiento e inferencia eficiente, lo que puede limitar su uso práctico.

VI. Retrieval-Augmented Generation (RAG)

Dadas las limitaciones de los LLMs tradicionales, los enfoques de *Retrieval-Augmented Generation (RAG)* entran en escena. El enfoque RAG potencia los LLMs conectándolos con un módulo de recuperación de información (*retriever*) que inyecta conocimiento externo relevante en tiempo real. Esto permite generar respuestas más precisas y coherentes, accediendo a información actualizada y fundamentada.

A. Preparación de los documentos

Los documentos que se desean utilizar para la recuperación se dividen en fragmentos llamados *chunks*, normalmente de entre 200 y 500 tokens. Para evitar pérdida de información, los chunks suelen tener un *overlap* entre sí.

1) Chunking de tamaño fijo: Se segmentan los documentos en trozos de longitud fija, respetando en la medida de lo posible los límites de las frases, y se mantiene un *overlap* para preservar contexto.

2) Chunking recursivo: En este enfoque, los chunks no se cortan estrictamente según el tamaño máximo, sino que se ajustan para mantener la semántica de las oraciones. Se comparan oraciones con la similitud del coseno y, si son suficientemente similares según un umbral, se combinan en un chunk más grande, logrando un almacenamiento más eficiente y contextual.

B. Transformación en embeddings

Cada *chunk* se convierte en un vector mediante un modelo de embeddings. Estos vectores se utilizan para medir similitud semántica y permitir la recuperación eficiente de fragmentos relevantes durante la consulta del usuario.

C. Indexación

Los vectores resultantes se almacenan en bases vectoriales especializadas, que pueden residir en memoria RAM o disco:

- FAISS: principalmente en RAM, rápido para búsquedas.
- Qdrant: almacenamiento en disco con soporte de búsqueda vectorial.
- Pinecone: almacenamiento en disco y nube, escalable.

Se almacena además la metadata asociada, como el texto original del chunk, para permitir una recuperación eficiente.

D. Consulta o recuperación

Cuando llega una pregunta del usuario, el proceso consiste en:

1) Transformar la consulta en un *embedding*.
2) Calcular la similitud con todos los embeddings indexados.
3) Seleccionar los *top-k* chunks más cercanos semánticamente.

E. Augmentación y generación

Para enriquecer el prompt del LLM, los chunks recuperados se organizan en una plantilla estructurada, que combina el *context* extraído de los documentos con la *question* del usuario. Esta plantilla asegura que el modelo reciba toda la información relevante de manera coherente, permitiéndole generar respuestas precisas y fundamentadas.

A modo de ejemplo, la Figura 4 muestra la estructura de la plantilla, donde se pueden observar sus componentes principales: *prompt*, *context* y *question*.

```
prompt_in_chat_format = [
    {
        "role": "system",
        "content": """Using the information contained in the context,
give a comprehensive answer to the question.
Respond only to the question asked, response should be concise and relevant to the question.
Provide the number of the source document when relevant.
If the answer cannot be deduced from the context, do not give an answer.""",
    },
    {
        "role": "user",
        "content": """Context:
{context}
---
Now here is the question you need to answer.

Question: {question}""",
    },
]
```

Fig. 4. Estructura de un documento RAG

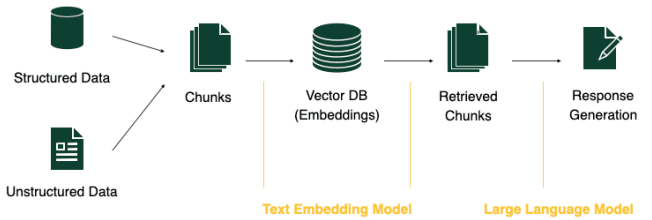

Fig. 5. Diagrama del flujo de *RAG* mostrando la preparación de documentos, generación de embeddings, indexación, recuperación y generación de respuestas.

F. Beneficios de RAG

- Reducción de alucinaciones.
- Actualización continua con información reciente.
- Eficiencia en costos y tiempo de respuesta.
- Aplicabilidad en dominios especializados con información privada.

G. Aplicaciones de RAG

- Asistentes empresariales enriquecidos: pueden consultar documentación interna y responder de forma precisa.
- Investigación: lectura automática de papers, resúmenes y citas.
- Soporte al cliente: análisis de tickets previos para generar respuestas coherentes y rápidas.

La Figura 5 ilustra el flujo general de un sistema RAG, donde se integran la creación de embeddings, la indexación y la búsqueda vectorial para enriquecer las respuestas generadas por el LLM.

Si bien los RAG mejoran significativamente el rendimiento de un LLM tradicional al proporcionarle información externa y actualizada, estos sistemas siguen siendo pasivos: no pueden buscar activamente información en la web ni tomar decisiones autónomas. Su función se limita a complementar la respuesta del LLM con los datos recuperados.

VII. De LLMs a agentes inteligentes

Los *agentes basados en LLMs* representan un paso más allá de los RAGs. Mientras que los RAGs solo enriquecen respuestas con información recuperada, los agentes pueden razonar, planificar y actuar de manera autónoma, ejecutando tareas en el mundo real, como consultar APIs, buscar información o tomar decisiones basadas en conocimiento externo.

Esta capacidad se estructura en tres componentes principales:

A. Memoria

La memoria permite al agente mantener coherencia y contexto a lo largo de la interacción:

- Corto plazo: ventana de contexto del modelo.
- Largo plazo: bases de datos externas, incluyendo sistemas RAG donde la información se divide en *chunks* y se representan como *embeddings* para su recuperación.

B. Planificación

La planificación dota al agente de la habilidad de descomponer problemas complejos y razonar sobre múltiples pasos:

- *Chains of Thought (CoT):* razonamiento secuencial paso a paso.
- *Trees of Thought (ToT):* exploración de múltiples posibles caminos de razonamiento antes de tomar decisiones.

C. Acción

Finalmente, la acción permite al agente ejecutar tareas concretas y aplicar su razonamiento en el mundo real:

- Utilización de herramientas externas, como buscadores, APIs o sistemas RAG.
- Enriquecimiento de respuestas con información recuperada en tiempo real, basada en *chunks* de documentos relevantes.

VIII. Conclusión

Los Modelos de Lenguaje Extensos (LLMs) han revolucionado el procesamiento del lenguaje natural, permitiendo tareas complejas como la generación de texto coherente, el razonamiento contextual y la ejecución multitarea sin necesidad de reentrenamiento.

El uso de *embeddings* y espacios vectoriales permite que los LLMs comprendan relaciones semánticas profundas. Adicionalmente, técnicas como Retrieval-Augmented Generation (RAG) mejoran su precisión y acceso a información actualizada, mientras que los agentes inteligentes basados en LLMs les permiten actuar de manera autónoma, planificar y utilizar herramientas externas, superando la pasividad de los modelos tradicionales.

A pesar de estas mejoras, los LLMs y sus extensiones enfrentan limitaciones importantes, como memoria finita, costos computacionales elevados y riesgo de alucinaciones. Por ello, su implementación requiere un diseño cuidadoso y un uso responsable, asegurando que sus capacidades se aprovechen de manera eficiente y confiable.