📚 **Resumen de la Clase 2 – Word Embeddings y Representaciones Semánticas**

### **🧩 Contenido Técnico**

#### **1. Segmentación y Tokenización**

* Se repasó el proceso de **segmentación de textos**: dividir documentos en unidades más pequeñas (capítulos, párrafos, oraciones, palabras, caracteres).
* Se definió el concepto de **token** como la unidad mínima que será vectorizada (puede ser palabra, subpalabra, etc.).
* Cada token se representa numéricamente mediante un índice → necesario para alimentar modelos de aprendizaje automático.

#### **2. Limitaciones de representaciones clásicas**

* Se revisaron representaciones como **one-hot** y **bag-of-words**, que son **sparse** (poco eficientes) y **no capturan semántica**.
* Ejemplo: "viajo en colectivo" vs "voy en bus" → significan lo mismo, pero no comparten palabras → su representación queda ortogonal (producto escalar = 0).

#### **3. Word Embeddings**

* Se introdujeron los **word embeddings** como representaciones **densas** y **semánticamente informadas**.
* Objetivos:  
  + Reducir la dimensionalidad.
  + Capturar relaciones semánticas (banco como institución ≠ banco para sentarse).
  + Ubicar palabras similares cerca en el espacio vectorial.

#### **4. Modelos preentrenados: GloVe y FastText**

* **GloVe**: basado en **matrices de coocurrencia** (conteo de apariciones de palabras en contextos).
* **FastText**: basado en **n-gramas de caracteres** (permite generalizar a palabras no vistas).
* Ambos modelos se entrenan sobre grandes corpus (ej. Wikipedia).

#### **5. Evaluación: Analogías y Similaridades**

* Se presentó el **test de analogías**: relaciones como "Rey - Hombre + Mujer ≈ Reina".
* Se midió la **similaridad coseno** entre vectores.
* Ejemplos visuales con proyecciones de embeddings en 2D/3D usando **PCA** o **t-SNE**.

#### **6. Arquitecturas de Word2Vec**

* Se explicaron los dos modelos clásicos de **Word2Vec**:  
  + **CBOW (Continuous Bag of Words)**: dado un contexto, predecir la palabra central.
  + **Skip-Gram**: dada una palabra, predecir palabras de su contexto.
* Ambos entrenan una red simple donde la **capa de embeddings** es el subproducto útil.

#### **7. Entrenamiento y Negative Sampling**

* Entrenar embeddings requiere construir pares palabra-contexto.
* El **negative sampling** genera ejemplos negativos (pares aleatorios) para evitar un set de entrenamiento desbalanceado.

#### **8. Visualización**

* Se usaron técnicas de visualización (PCA, t-SNE) para observar cómo las palabras similares se agrupan espacialmente.

#### **9. Experimentos con letras de los Beatles**

* Se entrenó un modelo Skip-Gram sobre un corpus de letras de canciones.
* Se analizaron similaridades: por ejemplo, “darling” se asocia con “pretty”, “sleep”, etc.
* Ejemplo de aplicación práctica de embeddings entrenados desde cero.

### **📌 Organización del curso**

* Se explicó que el desafío siguiente será **entrenar embeddings personalizados** con un corpus propio.
* Se sugirió experimentar con analogías dentro del propio corpus (e.g., letras, artículos, etc.).
* Se introdujo la librería **Gensim** para trabajar fácilmente con embeddings.