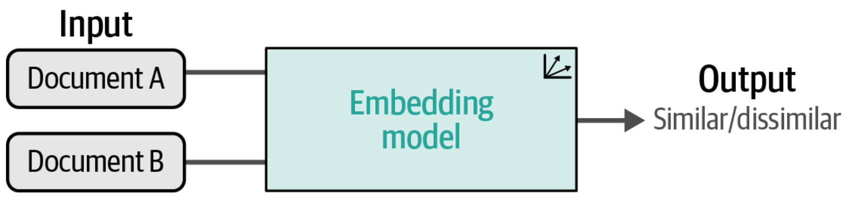
Chap-10
Creating Text Embedding Models

Objectives:

- Understand how text embedding models represent semantic meaning of text as numerical vectors.
- Learn the role of contrastive learning and SBERT in creating effective sentence embeddings.
- Apply embedding models for similarity-based tasks such as semantic search and retrieval.

Contrastive Learning (for Embedding Models)

- Contrastive Learning is a training technique used to learn embeddings by bringing semantically similar text representations closer and pushing dissimilar ones farther apart in the embedding space.



- It is one of the most important technique for both training and fine-tuning text embedding models.
- Through Contrastive Learning, the model learns relative relationships, not explicit class labels.

Pre-training (Teaches the model how language works in general)	**Fine-Tuning** (Teaches the model how to perform a specific task well)
• Pretraining is the initial stage where a model is trained on a very large corpus of general text.	• Fine-tuning is performed after pretraining to adapt the model to a specific task or domain.
• It helps the model learn basic language structure, grammar, word meanings, and contextual relationships.	• It uses task-specific data, which may be labeled or structured as similarity pairs.
• The training is usually self-supervised, meaning labels are automatically derived from the text itself.	• Fine-tuning adjusts the pretrained representations to better suit target applications such as semantic search or classification.
• Pretraining produces a general-purpose language model that can be reused for many tasks.	• This stage requires less data and computation compared to pretraining
• This stage requires high computational resources and significant training time.	• Fine-tuning improves task accuracy and relevance without retraining the entire model from scratch.
• The knowledge learned during pretraining is broad and domain-independent.	• The knowledge gained during fine-tuning is narrower and application-focused.

Why contrastive learning?

- Labeled data is expensive, time-consuming, and limited. However, most real-world data is unlabelled, so ontrastive learning enables self-supervised learning without manual annotations.
- Contrastive learning learns meaningful and rich feature representations, and it improves generalization across different tasks to reduce dependency on large labeled datasets.
- It requires fewer labels during fine-tuning and produces task-agnostic embeddings reusable for many applications.
- It works effectively for large-scale datasets and supports multimodal learning (image–text, audio–text, video–text).
- It improves accuracy in tasks like semantic search and retrieval.

Contrastive Loss:

- Contrastive loss is a loss function used in contrastive learning that trains a model to measure similarity between data pairs.
- It:

  Pulls similar samples closer in the embedding space
  Pushes dissimilar samples farther apart
- It penalizes the model when similar data points are far apart and when dissimilar data points are too close in the embedding space.

SBERT (Sentence-BERT)

- SBERT is a modified version of BERT designed to generate meaningful sentence embeddings that can be efficiently compared using similarity measures such as cosine similarity.

- It is a specialized architecture designed to efficiently generate sentence embeddings.
- Due to issues in standard BERT

  i.e., BERT compares two sentences using a **cross-encoder**, which:

  - Processes both sentences together
  - Is slow and expensive for large-scale similarity search

- Solution to above issues resolved by using a **Siamese** Transformer architecture. Which includes:

  - Two identical Transformer encoders (BERT/RoBERTa/etc.)
  - Shared weights between the encoders
  - Each input sentence is processed separately