1. Skip gram
2. CBOW
3. Custom BOW
4. Sentence embedding (Langchain, hugging, Sentence transform, transform)
5. RNN / LSTM / Transformer
6. LLM
7. API, huggingface

Word2vec (NN)

1. word → vector (Dense vector)
   (Non-zero)
   Less dimension.
2. Semantic info

   Vector

   (happy, Joy)

   (happy, sad)

3. Low-Dim (300, 700, 1000, 1500)

   ⇒ computation

4. Dense (info)

Word2vec

- **Pretrain**
  - google (google news)
  - 300
- **Custom**
  - data (domain cust) → 100, 200, 300 ...

Word ⇒ 300 dim ⟶ 2dim, 3dim ⟶ 

[0-1]

	F1 Gender	F2 Wealth	F3 Power	F4 Weight	F5 Speak
King	1	1	1	0.8	1
⇒ Queen	1	0.9	0.8	0.9	1
MAN	1	0.5	0.2	0.8	1
WOMAN	1	0.3	0.1	0.7	1
Monkey	1	0	0	0.3	0

Queen = KING − MAN + WOMAN

$$Queen = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0.5 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ 0.5 \\ 0.2 \\ 0.8 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0.3 \\ 0.1 \\ 0.7 \\ 1 \end{bmatrix}$$

man

KING − (MAN) + WOMAN

⑤ features

Type of Word2vec

- CBOW (Continuous Bag of words)
- Skip-gram.

CBOW (Neural Network knowledge is required) (Prequest)

Example ⇒ Follow | Sunny | for | gen AI → WS=1 ③

target

Window Size ⇒ 1, 2, 3, 4, 5 ..... n

Dimension of vector ⇒ 2, 3, 4, 5 ..... n

Window Size ⇒ 3 words continuously

Window Size = 1

⇒ 5 word cont.

WS ⇒ 2

⇒ 7 words

WS ⇒ 3

Dimension of vector = 3

⇒ window size = 1

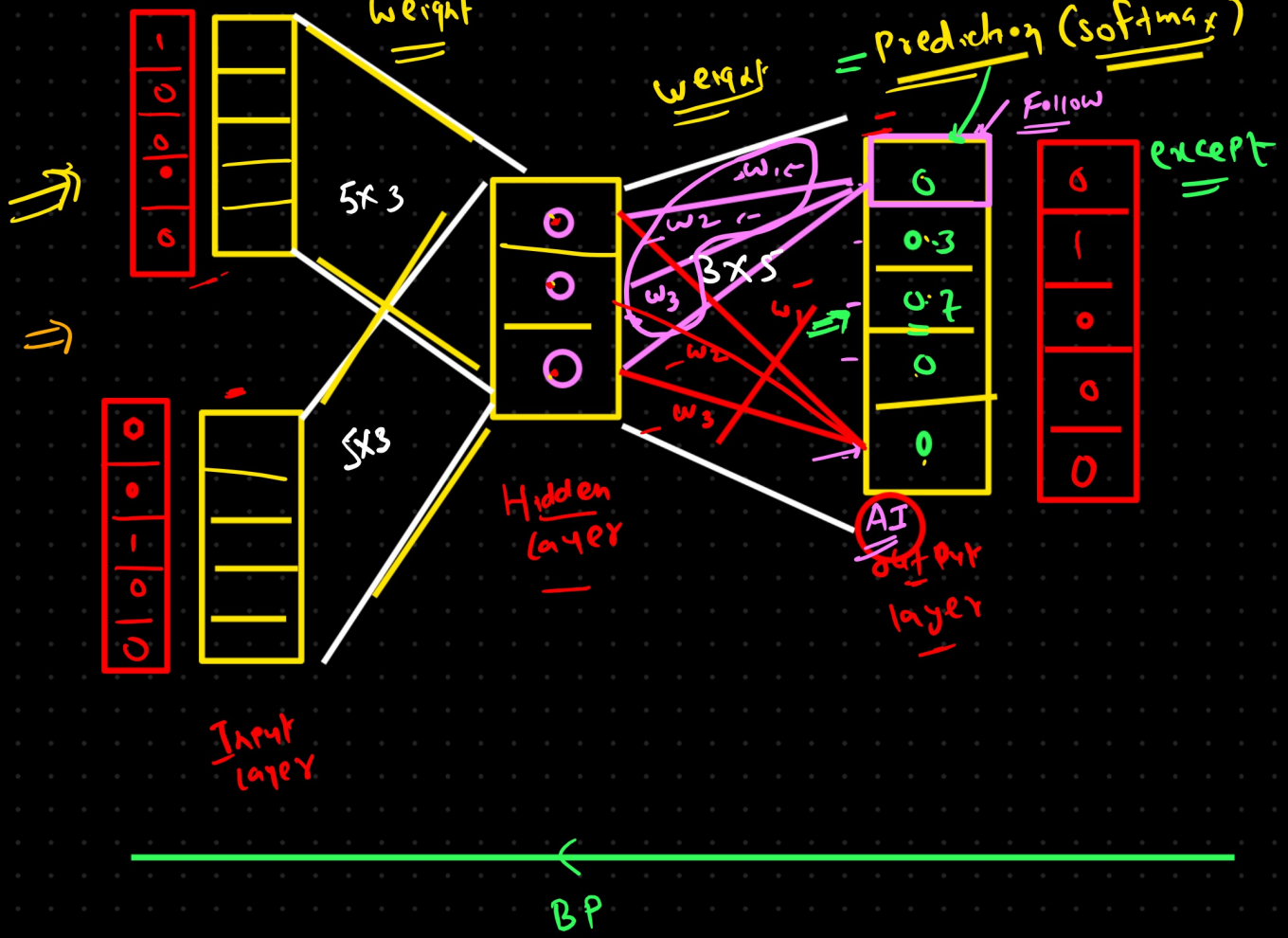
3 Continuous word

- [1 0 0 0 0]
  [0 1 0 0 0]
- [0 0 1 0 0]
  [0 0 0 1 0]
  [0 0 0 0 1]

X	Y
Follow, For	Sunny
Sunny, Gen	For
For, AI	Gen

Loss ↓

FP



BP

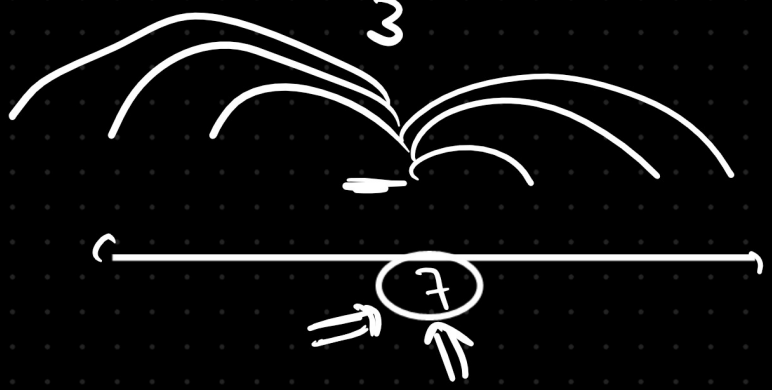
(Window Size)

Follow Sunny Savita For Gen AI and ML

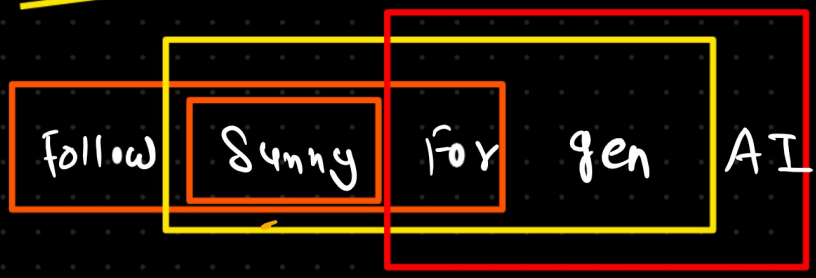
CBOW

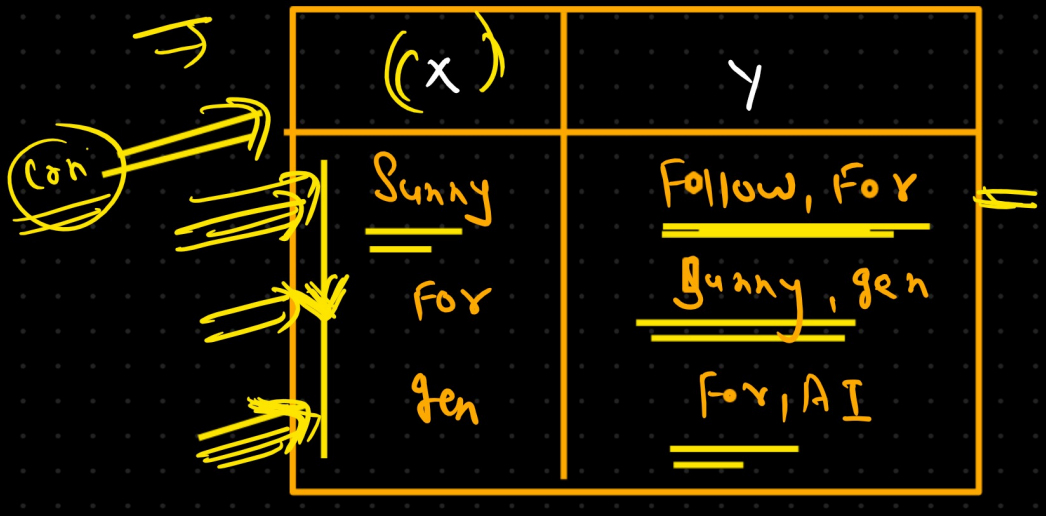
Window Size = 3

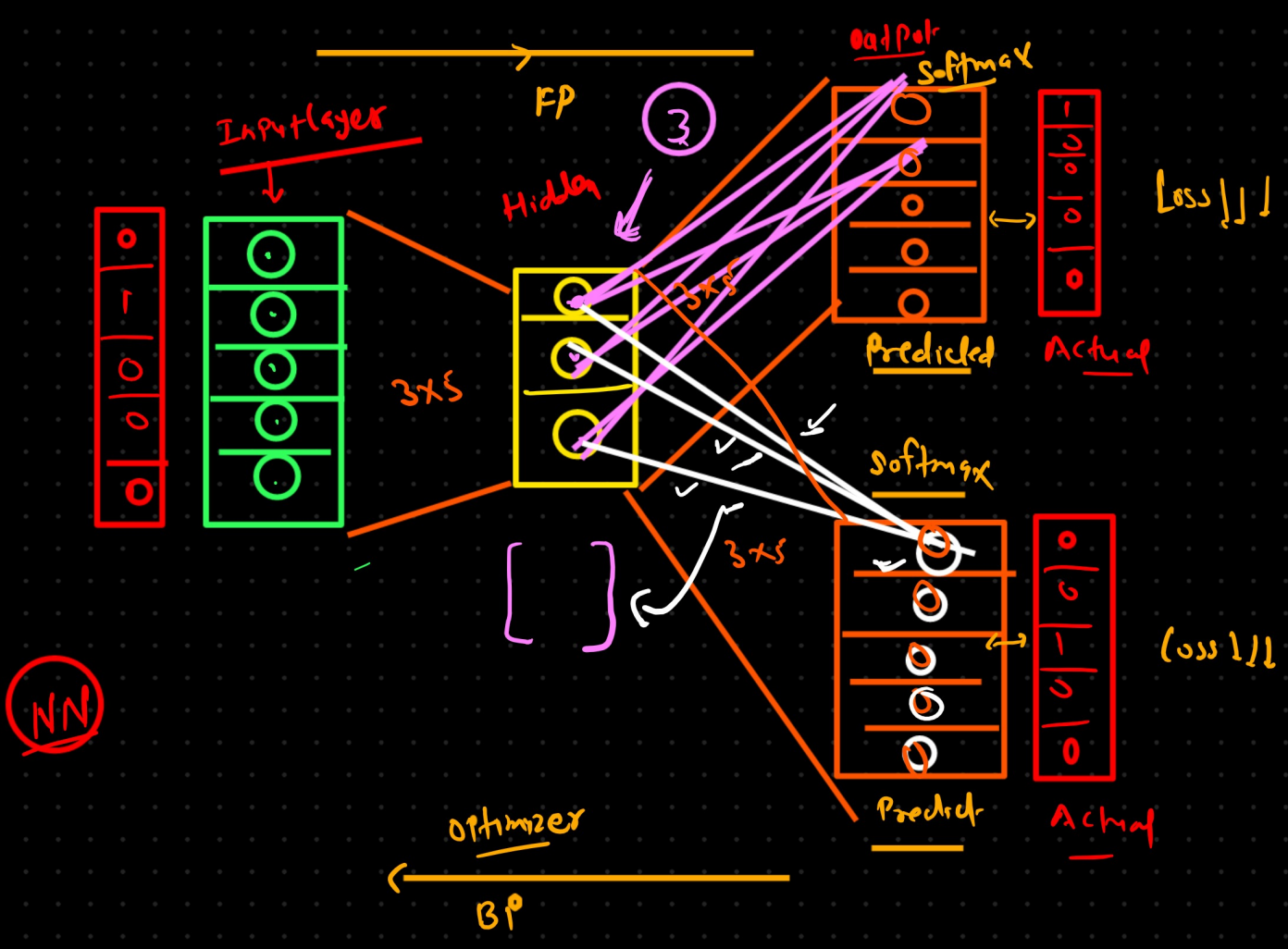
Context word = 7



Skipgram ⇒ Inverse of CBOW





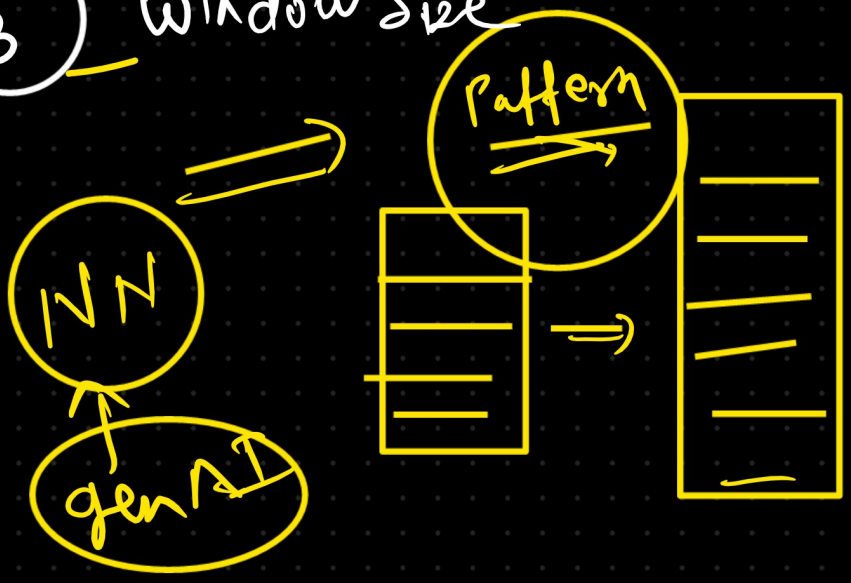


Word2vec

- (CBOW)
  - Small data
  - (10k-15k)
- Research paper
- Skip-gram.
  - (large data)
  - (millions)
- Subjective
- Experimental

word2vec

1. Increase the training dataset
2. Increase dimension of vector
   (100, 200, 300)
   ↓ good acc
3. Window Size



(Word → vector)

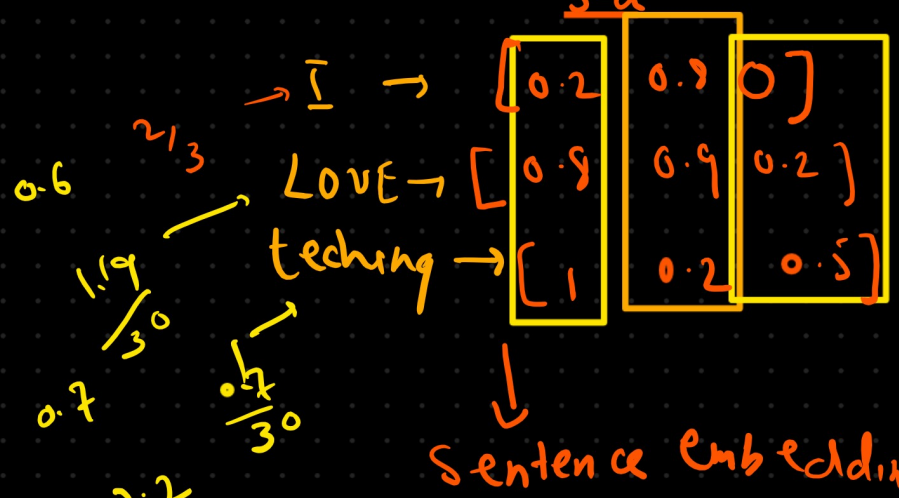
⇓

Sentences.

(Collection of words)

⇒ (Avg word2vec)

I LOVE teaching



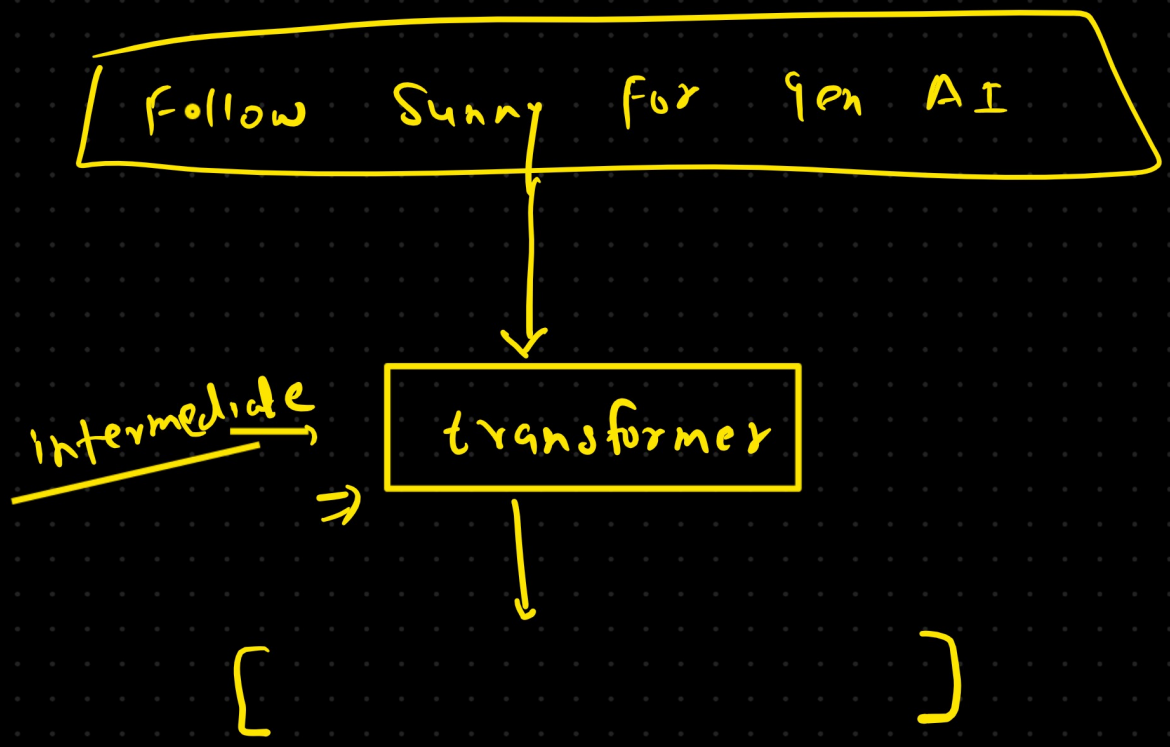
$\frac{0.2+0.9+1}{3} = \frac{2}{3}$

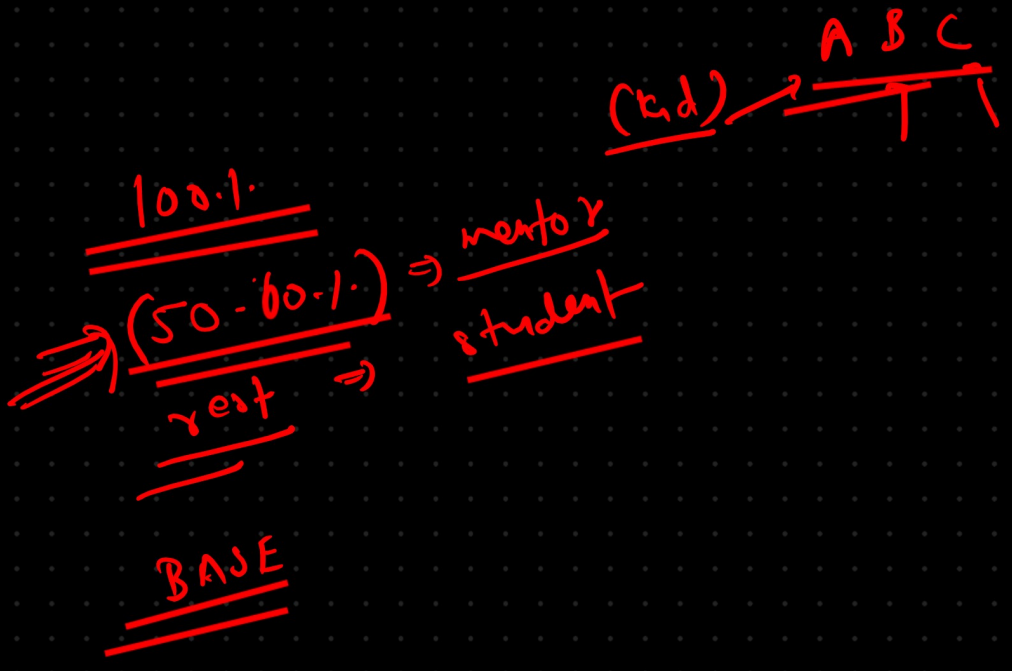
$\frac{0.8+0.9+0.2}{3} = \frac{1.9}{3}$

$\frac{0+0.2+0.5}{3} = \frac{0.7}{3}$

⇒ (I LOVE teaching) = [0.6, 0.7, 0.2]

avg word2vec


Follow Sunny for gen AI
intermediate
transformer
[ ]


A B C
(std)
100.1.
(50-60.1.)
rest
mentor
student
BASE

1. Data Ingestion
2. Transformation (embedding)
3. Model (Bert)
4. Evaluation

(Bert)

hugging transformer

(sentence transformer)

(Bert)

- OpenAI → gpt
  OpenAI embedding

1024

(BAAI) mpnet

Dataset size: Larger datasets generally benefit from more powerful models like MPNet.

Computational resources: If you have limited resources, BGE Small En or MiniLM might be better options.

Task complexity: For complex tasks like question answering or text summarization, MPNet is often preferred.

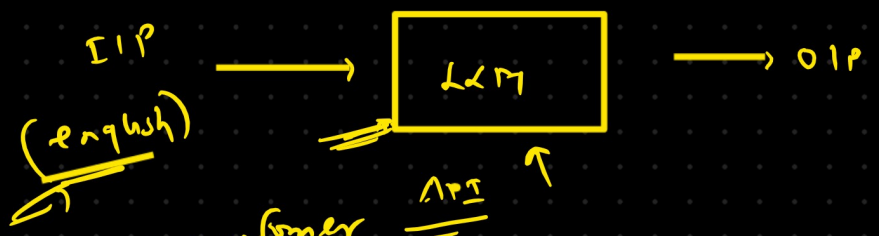
Embedding dimensionality: Different models produce embeddings of varying dimensions.Choose based on downstream task requirements.

Performance vs. efficiency trade-off: Decide if you prioritize high accuracy or faster processing

Experimentation is key. Try different models and evaluate their performance on your specific task and dataset to find the best fit.

Preprocessing (embedding) model

→ gpt-4 ← transformer (NN) (embedding)



Transformer API

→ Preprocessing (Lang 3.)

= (RAG)