Autoencoder
encode to compressed meaningful rep.
learn non-linear dependency
bottle neck / latent layer
latent var ↑
↳ not learn
$x \rightarrow f(x) \rightarrow z\ f(x) \rightarrow x'$
epoch ↑ x' improve
reconstruction Loss $|x - x'|$ ↓
back propagation
I/p shape = O/p shape
⇒ auto.fit(
X train, X train,
valdat (X test, X test))

Sparse AE
compress + sparsity
$|x - x'| + \lambda$ Regularization
↳ cost function

Convolutional AE
E → conv2D + Maxpool
D → conv2D + upsampling

Upsampling 2D Vs Conv2D Transpose
up nearest neighbor no trainable para
conv2D − CNN type so has weights

Denoising AE
when input data is corrupted
denoising image, audio signal, text denoising
- lower risk of learn identity function
- to remove noise
- (x_train_noisy, x_train)
- (x_test_noisy, x_test)

Variational AE
for randomness, → fake image ~
deterministic → probabilistic app.
$z = \mu + \sigma * \varepsilon$, $N(0,1)$ normal dis
↓ pass to decoder
garbage in → garbage out
reparameterization
↳ allow back propagation
↳ to overlapping
$(\mu + \sigma)$ far away high gap so use KL divergence
loss fun − RCL + KL
shape of μ, σ, z shape
learnable para (μ, σ)

Kullback-Leibler div
1 p(x) is diff from 2 q(x)
not symmetric $D(P||Q) \neq D(Q||P)$
info lost when ...

AE for anomaly detection
1) make AE
2) compute reconstruction error
3) select threshold
anomalies ↑ RL ↑

GAN
boundary b/w class
conditional probability
- zero sum game
Nash equilibrium
fake image shape = original image shape
G & D trained separately
G → fake image (0) → true label (1)
we can't control the o/p type of output image
- deterministic loss fun.
d_{L1} = dis (real image, 1)
d_{L2} = dis (fake image, 0)
$L = \frac{d_{L1} + d_{L2}}{2}$
generator's bais is always false →
batch normalization
discriminator − vanilla (3-4 layer) so generator can over take it.

CGAN
- apply condition on class labels
- G & D are conditional
- loss function is probabilistic
- class info pass by embedding layer

G − U-net based arch (ReLU)
D − convolution patch GAN (Leaky Relu)

pix2pix − paired images
cycle GAN − unpaired images

Evaluation metric
1) capture both intra & inter class variability
2) align with human perception
3) less sensitive to pre trained model
4) not dependent on distribution
5) detect mode collapse

Mode collapse
G found a way to fool dis. so generate same images
→ data Reg/aug/...
→ multiple generator
→ WGAN (earth movers dis.)

Inception score
pre-trained V3
obj should have a conditional class
only on generated images
50K → IncV3 → [ ] 50K×1000

FID score
real & generated images
frechet dis.
RI → V3 − feature emb
GI → V3 − feature emb } frechet dis.
FID ↓ better

BOW
Binary BOW − document term incidence
frequency BOW − freq of word
Tokenization − split of words
Vocabulary − unique word
Encoding − assign a count to word
shape of BOW = (vocab × # of docs)

document similarity
cosine similarity $= \frac{a.b}{|a||b|}$
→ best bcz dot prod is cheapest matrix/vector operation
NLP → sparse matrix, high dim (so cosine gives best result.

Limitation of BOW
1) high sparse matrix curse of dimensionality
2) vocabulary size ↑ if new data add
3) no grammar/sequential info/semantic
4) polysemy & synonymy
5) issue with negation

N-grams
capture context, solve negation issue
Limitations −
1) limited semantic under
2) sparse representation
3) limited context

TFIDF
term frequency − inverse doc. frequency
global algorithm
$TF(t,d) = \frac{\# \text{ of } t \text{ in } d}{\# \text{ of word in } d}$
$IDF(t,D) = \log\left(\frac{\# \text{ of doc in } D}{\# \text{ of } D \text{ that has } t}\right)$
tfIDF = Tf * IDF
global algo
↓ impact of common stop word.
extracting relevant pattern

Limitation
lack context & order
high dim → slow down
no semantic similarities
struggle if short doc.

Text pre processing
lowercase → punctu → tokenization → token → stop word → stemming / lemmatization.

Tokenization
word based, char, sub word, sentence, chunking

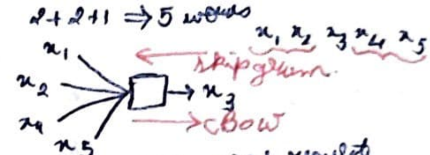
Out of Vocabulary (OOV)
word that are not encountered in the training dataset
sol^n →
Break OOV →
Unknown [UNK] →
character level
Byte pair (BPE)
rare words + OOV
breaking OOV into words sub words that makes likely seen during
BPE is language agnostic.

Text preprocessing
padding tokens
all input size matching.
zero padding [0...]
EOS padding [EOS]
post padding
pre padding

Word embedding
dense, meaningful representation
dim: vocab × embedding size
semantic prope
Input 2 = 7 | 192 para
dim: 8
I/p dim = 7 × 8 | ↑ 24×8
Vocab size 24 ↑

Word 2 Vec
1) CBOW
predict target from context word

2) skip gram
predict context from target

softmax → slow (10K+ class high no. of node last layer
window size = 2 {−2 to +2}
2+2+1 ⇒ 5 words


non-zero vector, good results for syntactic & semantic word similarity.
⇒ solution to skip-gram
↳ hierarchical softmax
↳ skip-gram −ve sampling
3) skip gram with −ve sampling
multi class classification (softmax)
↳ to binary (sigmoid
1 if pair found in dataset
0 if word not found in dataset

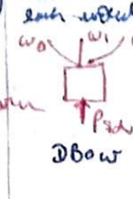
Adv of Word 2 Vec
semantic relationship b/w words analogies.

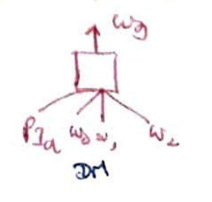
Limitation
polysemy, being sensitive to training data.

GloVe − co-occurrence matrix
meaning based on their usage pattern across the entire corpus.
$P(I|like) = \frac{P(I \cap like)}{P(like)}$
ratios of co-occurrence with prob k
$\frac{P(k|ice)}{P(k|steam)}$ >>1 more related to ice; <<1 more related to steam; close to 1 ⇒ equal

Sentence Embedding
weighted average of all words in doc
$[\ ] + [\ ] + \dots = \left[\frac{w_1 + w_2 \dots}{n}\right]$
context awareness, retention of word order. on diff length.
→ loss word order
→ use parking NL sentence ...

Limitation
represent word isolation, diff for longer text/doc, losses word order as BoF.

Doc to vec
1) distributed (DM) memory
fixed length vectors for each text unique doc ID is added
2) DBOW
doc to learn & predict prob of each word.




Vector database

ANN, RAG
maps sentence & para to 384 dim vector.

N-gram language model

prob over words
Bi-gram, tri-gram
N-gram
(context vector ⇒ context size* vocab size) by concatenation of 60 vector

N↑ – large content ↑
complex ↑
exponentially ↑
sparse data
sliding window.
activation → softmax
loss → sparse_categ_ce
padding → pre padding

Retrieval aug generation (RAG)

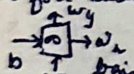
hallucination, outdated knowledge
Non-transparent & untraceable reasoning
Indexing, retrieval & generation

Drawback

retrieval challenges
generation issue
augmentation hurdles

RNN

- time series analysis, sequential data (NLP), autoreg (ARIMA)
  ↳ fore-casting &



bias are of 1

$\omega_x \omega_h \omega_y$ same in all hidden layer.

$\omega_h = \tanh(\omega_h h_{t-1} + \omega_x x_t)$

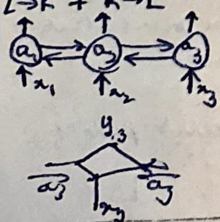
Sigmoid – vanishing gradient
ReLU – exploding gradient
bad for long term due to vanishing gradient

Limitation

1) not from future
2) ✗ long term
3) only L→R

Bidirectional model

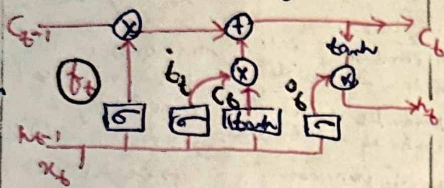
L→R + R→L



One to one – Image classification
One to many – text generation, image caption
many to one – sentimental analysis
many to many – text summarization & $(x_t \neq y_t)$ machine lang/trans.
many to many – entity recognition / part of speech tagging $(x_t = y_t)$

LSTM (Long term short memory)

4 gate & cell memory.



1) forget gate how much to forget
activation = sigmoid [0 1]
[0, 1] — 0 → forget, 1 → remember
weights, current info

$$f_t = \sigma(\omega_f \cdot [h_{t-1}, x_t] + b_f)$$

h_{t-1} → previous info, b_f → bias

2) Input Gate

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

3) Candidate layer

$$\tilde{C}_t = \tanh(W_i \times [h_{t-1}, x_t] + b_i)$$

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t$$

[Cell state]

3) Output layer

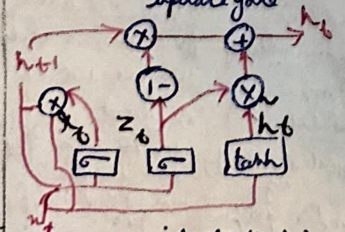
$$O_t = \sigma(\omega_i \cdot [h_{t-1}, x_t] + b_i)$$

$$h_t = O_t * \tanh(C_t)$$

(filtered version)
th [-1, 1] for C_t
sigmoid decides which part of all goes to output.

GRU (gated recurrent unit)

3 gates & no have a cell state
forget + input gate = update gate



1) update gate (how much info to pass)

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$

2) reset gate

can control the info flow
how much to forget

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$

3) Candidate hidden state

current memory content

$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

control both previous step & current step info using update gate

⇒ update gate → 0 (forget everything) (→ & pass)
1 → remember pass all.

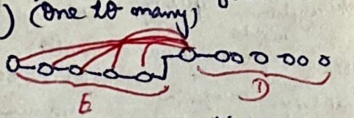
No. of parameter

RNN = $u(u+i+1)$ vain ✓
LSTM = $4 \times u(u+i+1)$ "
u = hidden units
i = input units
GRU = $3u(u+i+1)$

BLEU score is used for NLP and other related task

Sequence 2 Sequence model

CNN + LSTM – Image captioning & vision transformer) (One to many)



skip connection, to preserve previous info & to learn early word knowledge.

Transformer

- NN arch + attention based
- self attention to compute input sequence representation, capturing long-term dependency & enabling effective parallelization
- multiple machine translation.
- Encoder + decoder

[Multi head self attention
Add & normalized
feed forward]

also skip connection / residual connection & layer normalization
↳ to prevent over fitting
→ positional encoding (PE)

Multi head – allows model to self attention focus on input simultaneously

feed forward – helps to learn complex relationship coming from MHSA.

info. is processed parallely so to loss order

"Transformers are not sequential"

$x_1 = [\ ]_{512} + [0 \dots 0]_{512}$
$x_2 = [\ ]_{512} + [1 \dots 1]_{512}$
embedding, position vector
} wrong way bcz addition with the value insignificant as

embedding will be insignificant as PV↑ so E↓ ⇒ so will not be good.

Positional encoding

due to parallel processing → loss order & sequential info ⇒ so it uses PE.
- uses sinusoidal function – consist of series of sine & cosine wave with varying frequency. pos → position

$PE = \sin\left(\frac{pos}{10000^{2i/d}}\right)$
i → dim index, $PE = \cos$ – – –
d → total dim.

- no. of scope = no. of embedding
- multiple capture to get multiple position of a word.

⇒ word embedding + positional encoding
dim. of PE = dim of E
→ don't loose meaning of word after applying word.

# of encoder decoder ⇒ hyper parameter.

word → queries (Q), keys (K), values (V)
vector $[\ ]_{512}$

5K / 8 (head count) ↳ 64

H_1 ... $\frac{4}{8}$
Q_1 □ 64
K_1 □ 64
V_1 □ 64

$$\text{attention}(Q * V) = \sigma\left(\frac{QK}{\sqrt{d}}\right) \times V$$

QK → correlation along word
× V → converts scaler to vector
$\sqrt{d}$ → normalized & reduce the value to avoid exploding value

V → semantic embedding for each word.
$(H_1 | H_2 \dots H_n)$ → concat → skip connect – FF

BERT (bidirectional encoder rep from transformer)

- drop decoder
- multi-layer encoder
} context aware bidirectional } BERT

→ pre-trained with 2 task
1) Masked language modeling (MLM) – 15% masked
→ ↑ strength of bidirectional & content awares
2) next sentence prediction (NSP)
→ build relationship with sentences.

MLM → cross entropy loss
NSP → binary entropy (IsNext / NotNext)

Addition of 2 new tokens

1) CLS – classification
2) SEP – separation of sentence

I/P → token emb + segment emb + position embedding
- token emb: emb for each word
- segment emb: indicate token belong to which each sentence
- position embedding: position of each token in the sequence

(Transformer)

Total loss – Mask LM loss + NSP loss

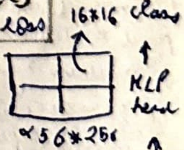
Vision Transformer (ViT)

divide the image into patches

$\frac{256 \times 256}{16 \times 16}$ ⇒ 16×16 ⇒ 256 patches

patches + PE → transformer input encoder

$OH, OW = \left\{\frac{\text{Input Height} - \text{kernel size}}{\text{stride}} + 1\right\}$, filter



Generative pre-training (GPT)

remove (encoder & cross attention to decoder)
training ⇒ self-supervised (next word pred)
GPT-2 does not require any fine-tuning for downstream – zero shot prediction.
masked multi head attention