Open Elective Course [OE]

Course Code: CSO507

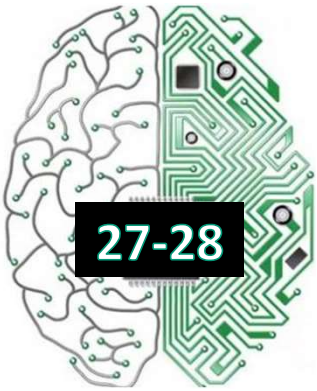
Winter 2023-24

Lecture#

Deep Learning

Unit-5: Transformer (Part-II)
Sequence Modeling for NLP

Unit-6: Representation Learning



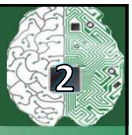
Course Instructor:

Dr. Monidipa Das

Assistant Professor

Department of Computer Science and Engineering

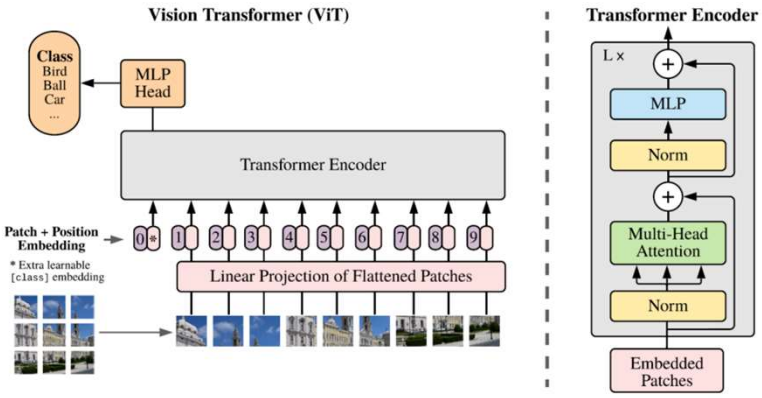
Indian Institute of Technology (Indian School of Mines) Dhanbad, Jharkhand 826004, India



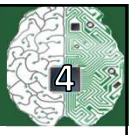
Transformers for Vision

Image Classification

- **Vision Transformer (’21) [ViT]**
  - Decompose an image to NxN patches and then apply transformer encoder



Standard Transformer Architecture



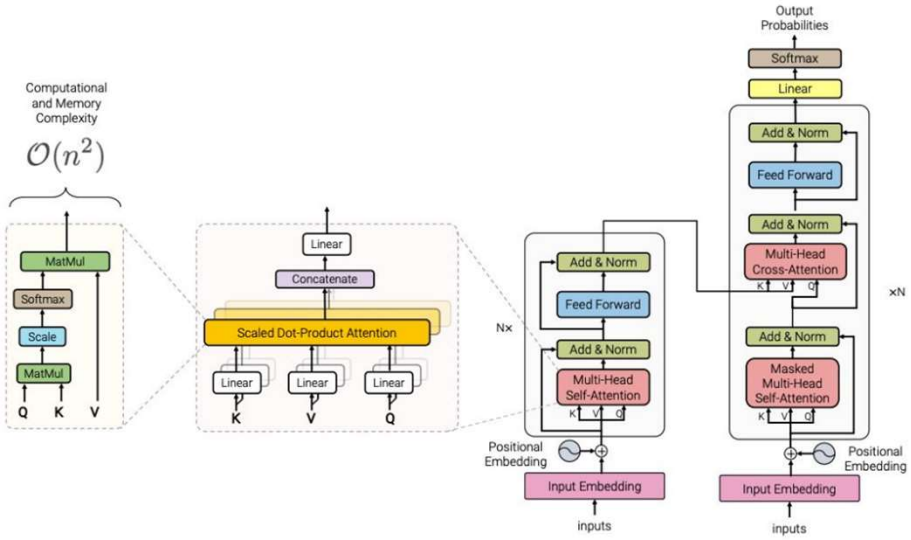


Image Classification

5

ResNet-ViT Hybrid

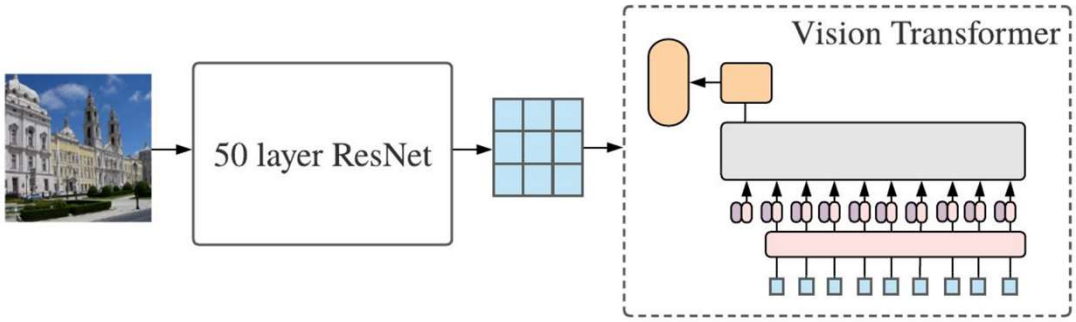
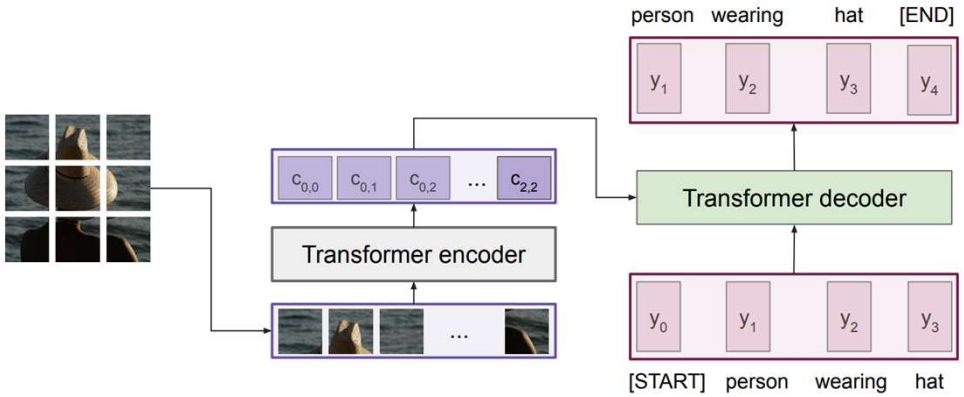


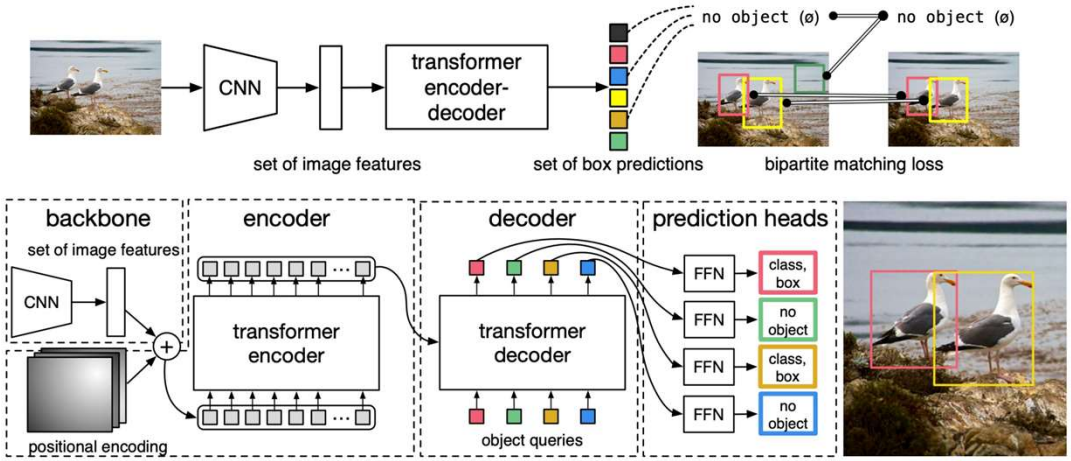
Image Captioning

6

- **Image captioning based on ONLY Transformer**



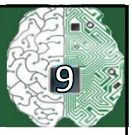
Object Detection with Transformers: DETR



Carion et al, "End-to-End Object Detection with Transformers", ECCV 2020

Sequence Modeling for NLP

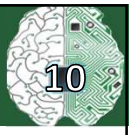
Natural Language Processing (NLP)



- **NLP:** Use of natural languages, such as Hindi, English, French, Bengali, etc. by a computer.

- **NLP Applications:** mostly based on ***language models***
  - ***Machine Translation***
  - ***Speech Recognition***
  - ***Document Summarization***
  - ***Question answering***
  - ***Caption generation etc.***

Language Model (LM)

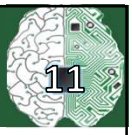


- An LM defines a *probability distribution* over tokens in natural language

- Earliest successful LMs were based on fixed length sequences of tokens called *n-grams*

- Neural Network based Language Models
  - Generic Methods
  - Domain-specific Methods
    - To build an efficient model, we must use techniques specialized to sequential data

NLP: n-Gram Models



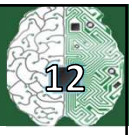
- *n*-gram models define the conditional probability of the *n*-th token given the previous $n-1$ tokens

$$P(x_1,..,x_\tau) = P(x_1,..,x_{n-1})\prod_{t=n}^{\tau} P(x_t \mid x_{t-n+1},..,x_{\tau-1})$$

Sequence of length n

- Comes from chain rule of probability
- For small values of n, we have
  - n=1 : unigram P(x1)
  - n=2 : bigram P(x1, x2)
  - n=3 : trigram P(x1, x2, x3)

Training n-Gram Models



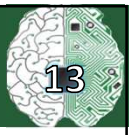
- Usually train both an n-gram model and an n-1 gram model making it easy to compute

$$P(x_t \mid x_{t-n+1},..,x_{\tau-1}) = \frac{P_n(x_{t-n+1},..,x_t)}{P_{n-1}(x_{t-n+1},..,x_{t-1})}$$

n-gram probability

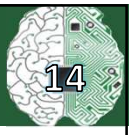
n-1 gram probability

Shortcomings of n-gram models

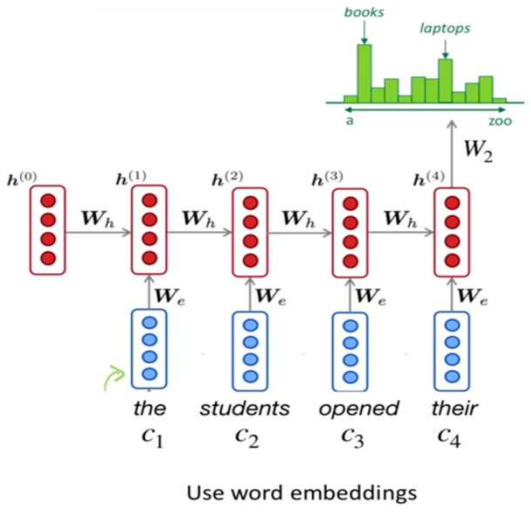


- Vulnerable to curse of dimensionality
- Even with a massive training set most n-grams will not occur
- Insufficient context
- Two different words at the same distance in one-hot vector space

Neural Language Models (NLMs)

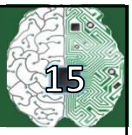


- How does it solve the problem?
  - Words as dense distributed vectors
  - there can be sharing of statistical weight between similar words



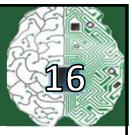
Use word embeddings

Strength of NLMs



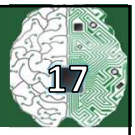
- Share statistical strength between one word (and its context) and other similar words and contexts
- Distributed representation allows model to treat words that have features in common similarly
- Overcomes curse of dimensionality for sequences

Improvements of Transformer

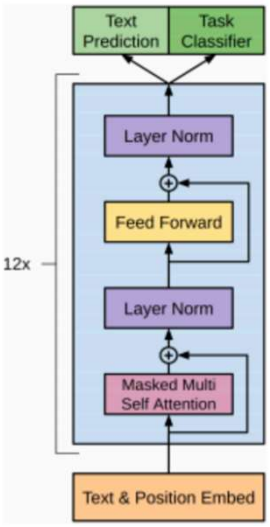


- Open AI's Generative Pre-Training (GPT)
  - Combines supervised and unsupervised learning to improve word vectors
    - Performs Generative Pre-Training
  - State-of-the art for following tasks:
    - Textual entailment, semantic similarity, reading comprehension, common sense reasoning, linguistic acceptability, multi-task benchmark
- Google's BERT
  - Bidirectional Encoder Representations from Transformers
  - BERT is an improvement of GPT
  - Key technical innovation is bidirectional training of Transformer

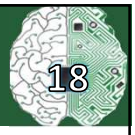
Architecture of GPT



- Text and position will be transformed to a vectors
- Pass to multi-head self-attention
- Combining result from step 1 and step 2 and performing a normalization
- Pass to a fully-connected feed-forward network
- Combining result from step 3 and 4 and performing a normalization
- Finally, combing multi-head (total 12 self-attention block) to together for computing vectors.



Generative Pre-training (GPT)



- OpenAI multilayer decoder

W_e is the token embedding matrix
W_p is the position embedding matrix
$h_0 = UW_e + W_p$
h_i = transformer block $(h_{i-1})\ \forall i \in [1, n]$

- Unsupervised pretraining task
  - Language Modeling

Language Modeling Loss

$$P(u) = \text{softmax}(h_n W_e^T)$$
$$L_1(U) = \Sigma_i \log P(u_i | u_{i-k}, .., u_{i-1}; \theta)$$

- Supervised fine-tuning

Classification Loss

$$P(y|x^1, \ldots, x^m) = \text{softmax}(h_n^m W_y)$$
$$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y|x^1, \ldots, x^m)$$

Final Loss

$$L_3(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C})$$

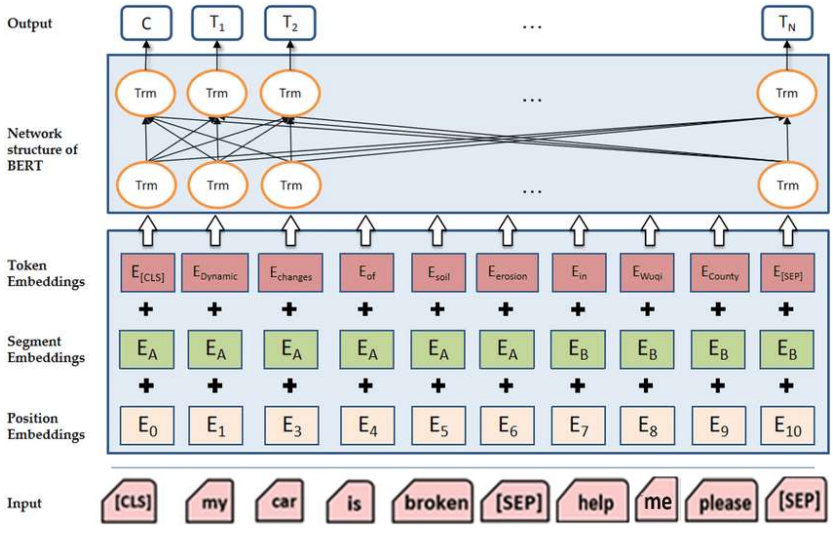
BERT

19

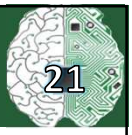
- Bidirectional Encoder Representations from Transformers
- Uses Transformer attention mechanism to learn contextual relations between words
  - Transformer includes two mechanisms
    - An Encoder that reads text input
    - A Decoder that produces a prediction for the task
  - Since BERT's goal is to generate a language model, only the encoder mechanism is necessary
- Uses Bidirectional training
- Deeper sense of context/flow than single-direction
  - Masked LM (MLM) allows bidirectional training

BERT Architecture

20

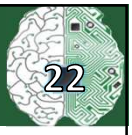


BERT Architecture



- BERT architecture builds on top of Transformer
- There are two variants:
  - BERT Base:
    - 12 layers (transformer blocks),
    - 12 attention heads, and
    - 110 million parameters
  - BERT Large:
    - 24 layers (transformer blocks),
    - 16 attention heads and,
    - 340 million parameters

Pre-training BERT



- BERT is pre-trained on two NLP tasks:
  - Masked Language Modeling (For Bidirectionality)
    - Before feeding word sequences 15% of words in each sequence replaced with a [MASK] token
      - Model attempts to predict original value of masked words, based on context provided by non-masked words in sequence
  - Next Sentence Prediction
    - A text dataset of 100,000 sentences has 50,000 training examples or pairs of sentences as training data.
      - For 50% of pairs, the second sentence would be the next sentence to the first sentence
      - For the remaining 50% of pairs, second sentence would be a random sentence from corpus
    - The labels for the first case would be ‘IsNext’ and ‘NotNext’ for the second case

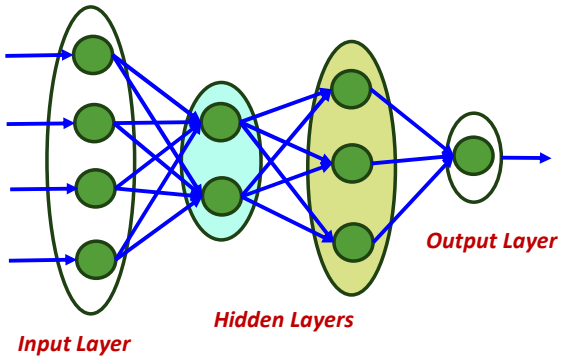
23

Unit-6: Representation Learning

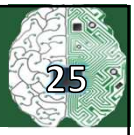
Representation Learning

24

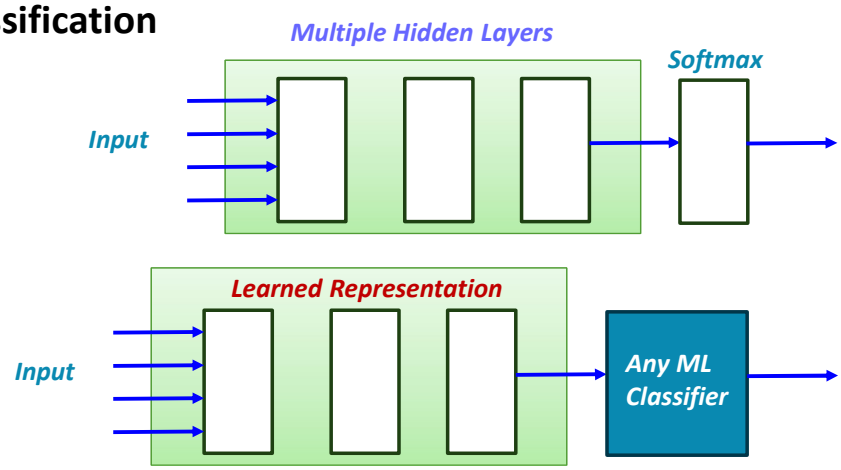
- **Learning new representations**
- **Intrinsic capability** of deep learning
- Representations give valuable insights into ***dimensionality reduction, transfer learning***, and ***learning embedding*** from data
- Simple task can become complicated due to inappropriate representations



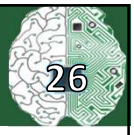
Scenarios of Representation Learning



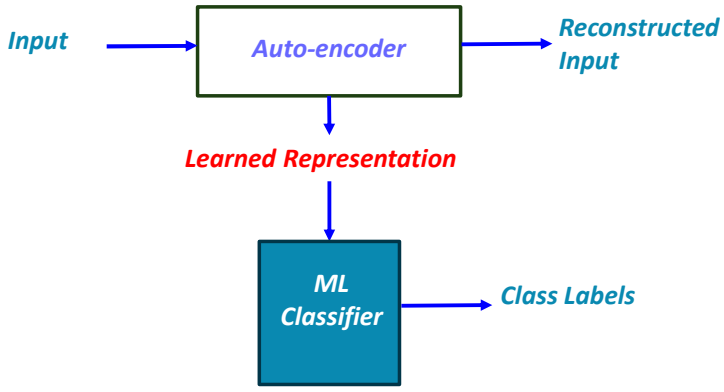
- **Classification**



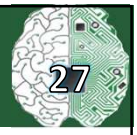
Scenarios of Representation Learning



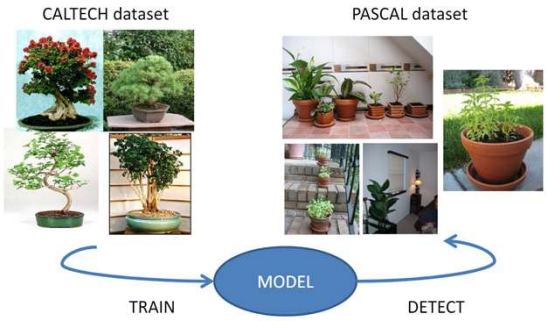
- **Semi-Supervised Learning**

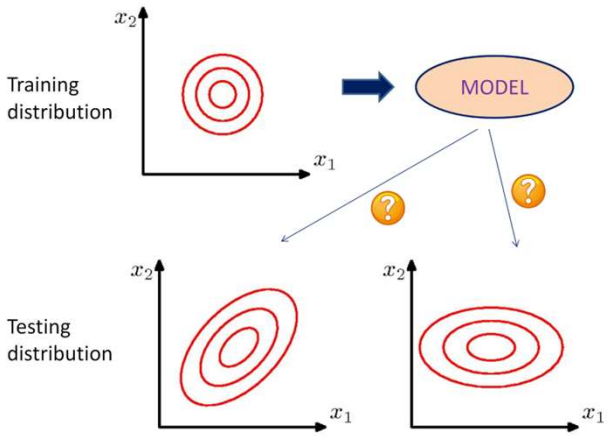


Scenarios of Representation Learning

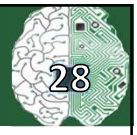


- **Transfer Learning**





Representation Learning in Text



- **Traditional Method**
  - Bag of Words
  - Topic Model
- **Word Embedding**
  - One-hot Encoding
  - Language Model
  - Continuous Bag of Word
  - Skipgram
  - Global Vector Representation
  - Pre-trained Model

Bag of Words

29

- **Corpus:** A collection of document
- **Document:** container of text. Documents are broken into words
- **Words:** often termed as terms
- **Count of words:** Term Frequency

Sample corpus:

Document 1 (D1): The game of life is a game of everlasting learning
Document 2 (D2): The unexamined life is not worth living
Document 3 (D3): Never stop learning

Term Document Matrix (TDM):

	The	game	of	life	is	a	everlasting	learning	unexamined	not	worth	living	never	stop
D1	1	2	2	1	1	1	1	1						
D2	1			1	1				1	1	1	1		
D3								1					1	1

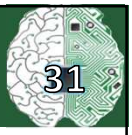
Bag of Words

30

- Basic Problem: Sense of order among words completely lost
- Partial solution: Bag of n-grams

- Other issues:
  - Sparse representation
  - Fails to capture word semantics
  - Traditional way of handling semantics: Topic Modelling

Topic Model



- Representing a document by topic

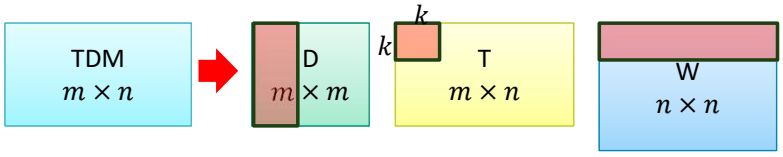
  Document vector as Topic-wise distribution:

Crime	Entertainment	Sports	Politics	Business
0.15	0.25	0.6	0	0

- Topic Modelling: Finding the relevant topics from the collection of documents

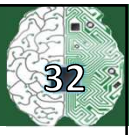
  -using SVD:

  Latent Semantic Analysis (LSA)



Select $k < m$ of the singular values

Topic Model



using LDA (Latent Dirichlet Allocation):

Document No.	Document Content
0	It was an exciting match of football. Ended with 4-4 goal each
1	The chicken and fish items were awesome, I liked the rice too.
2	The football match ended with a last-minute goal
3	I liked the fish tikka and chicken biriyani.
4	Fish and chips. Then brown rice with herbs.
5	We qualified for a semi-final inter-college football match.

Document No.	Topic 1	Topic 2
0	0.919	0.081
1	0.085	0.915
2	0.915	0.085
3	0.085	0.915
4	0.085	0.915
5	0.942	0.058

Word Embedding

33

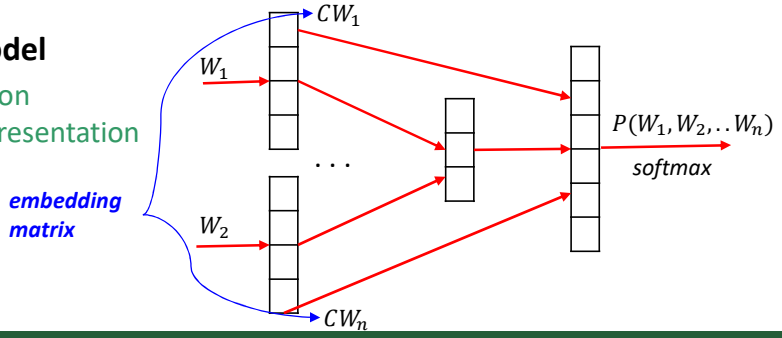
- **One hot embedding**
  - Lengthy vector
  - Sparse vector
  - Semantics lost

Corpus
Apple
Bus
Orange
Truck

Assuming 10 words in the vocabulary

0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	1

- **Neural Language Model**
  - Denser representation
  - Better semantic representation



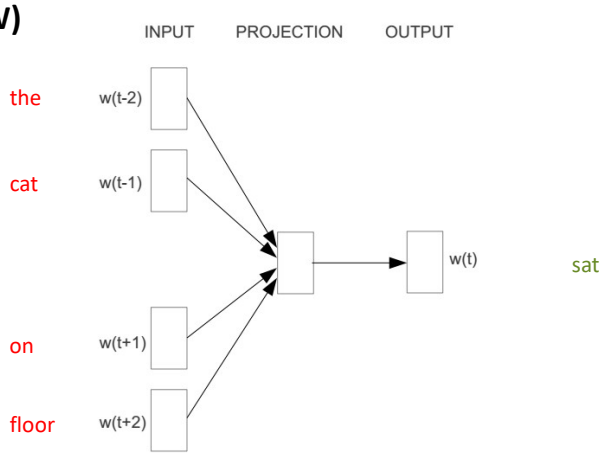
Word Embedding

34

- **Continuous Bag of Word (CBoW)**

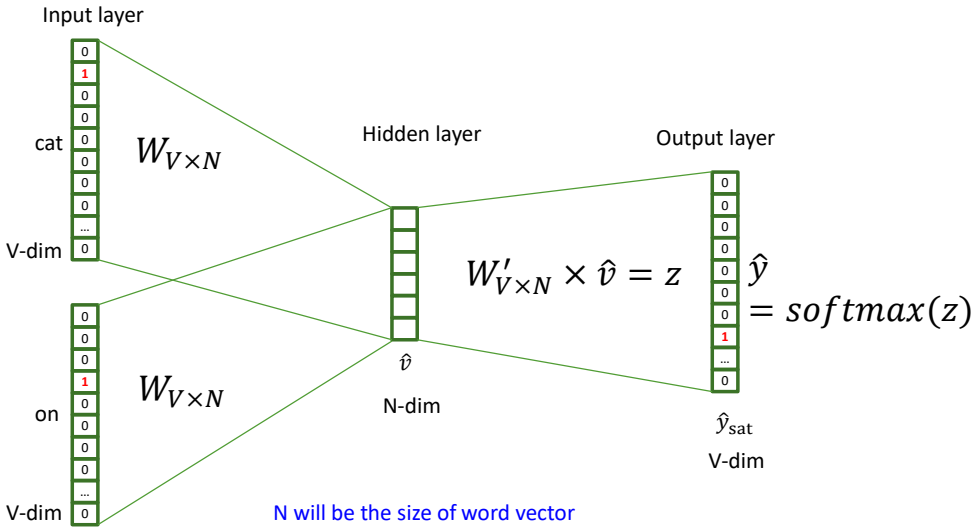
"The cat sat on floor"

Window size = 2



CBoW

35



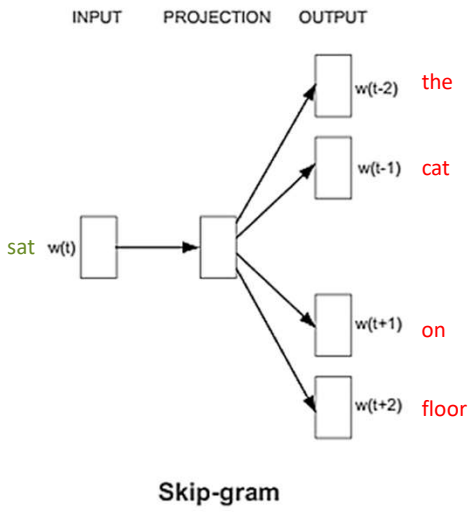
Word Embedding

36

- **Skip-gram**
  - alternative to CBOW

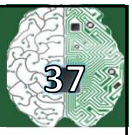
"The cat sat on floor"

Window size = 2



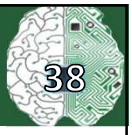
Skip-gram

Word Embedding



- Global Vector Representation (GLoVe)
  - Suppose the embeddings of the target word and context word are E_t and E_c, respectively
  - Also let the number of times they are co-occuring in the document is w_{tc}
  - GLoVe assumes that the inner product of these embeddings should be closed to the count
  - Loss is computed on $E_c^T * E_t + b_t + b_c - \log(w_{tc})$ and considering sum over all pairs of target and context (here b indicate bias terms)
- Pre-trained Model
  - Majority of the words do not change much from context to context.
  - Embeddings learn on general corpus like Wikipedia, can be used for other text classification tasks.
  - Using such an embedding is called pre-trained embedding

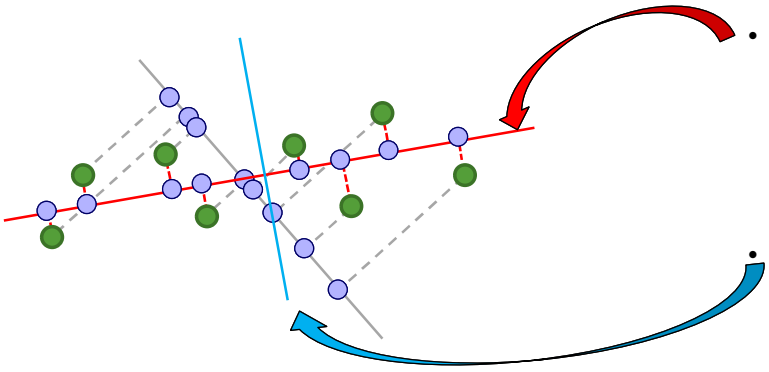
Traditional Representation Learning



- **Principal Component Analysis (PCA)**
- Unsupervised, linear dimensionality reduction
- Objectives
  - transforming a number of (possibly) correlated features into a (smaller) number of *uncorrelated* features called **principal components**
  - preserving as much of the *variance* in the high-dimensional space as possible (using linear projection)

Geometric picture of principal components (PCs)

39



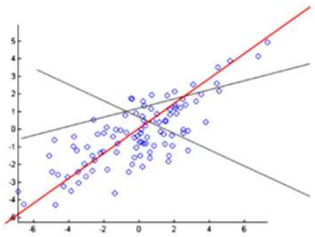
- First PC is the projection direction that maximizes the variance of the projected data
- Second PC is the projection direction that is orthogonal to the first PC and that captures greatest variability of what is left

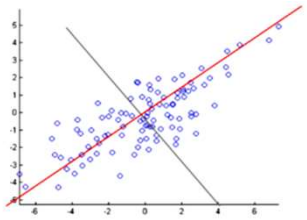
Each succeeding component accounts for as much of the remaining variability as possible

PCA: Conceptual View

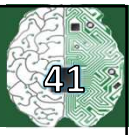
40

- Find a line, such that when the data is projected onto that line, it has the maximum variance.
- Find a second line, orthogonal to the first, that has maximum projected variance.
- Repeat until have *K*<*D* orthogonal lines.
- The projected position of a point on these lines gives the coordinates in the *K*-dimensional reduced space.





PCA: Algorithm



1. Create $N \times D$ data matrix X, with one row vector x_i per data point
2. $\Sigma \leftarrow$ covariance matrix of X

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\Sigma = \frac{1}{N}\sum_{i=1}^{N} (x_i - \bar{x})(x_i - \bar{x})^T$$

3. Find eigenvectors W and eigenvalues Λ of Σ

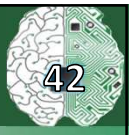
$$\Sigma = W\Lambda W^T$$

4. Principal Components W_K are the K eigenvectors with largest eigenvalues

$$W_K = [w_1 | w_2 | \dots | w_K]$$

5. Transformed data $Y = XW_K$

$N \times K$ $N \times D$ $D \times K$



Questions?