

INTRODUCTION TO
MACHINE LEARNING
COMPSCI 4ML3

LECTURE 30




HASSAN ASHTIANI

TEXT PROCESSING

- TEXT CLASSIFICATION
 - SENTIMENT ANALYSIS
 - SUBJECT CLASSIFICATION
 - ...
- TEXT-TO-TEXT
 - TRANSLATION
 - SUMMARIZATION
 -
- LANGUAGE MODELLING
 - LEARNING THE “DISTRIBUTION”
 - NEXT WORD/TOKEN PREDICTION
 - GENERATING NEW TEXT

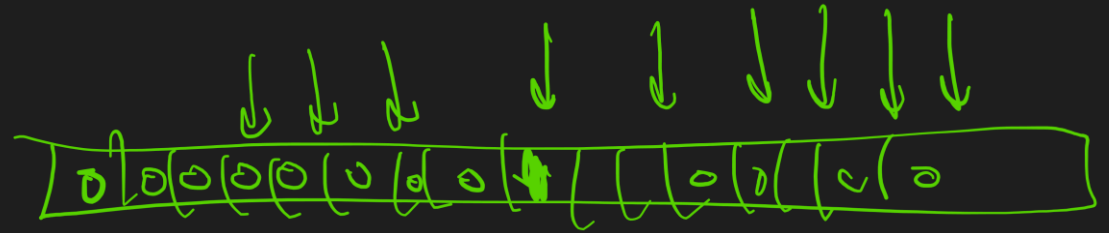
$p(x)$

VECTORIZING A TEXT

- TEXT CAN BE REGARDED AS A SEQUENCE OF
 - WORDS 
 - CHARACTERS 
 - TOKENS 
- WHICH ONE SHOULD WE USE?

REPRESENTATION OF A TOKEN

- HOW CAN WE REPRESENT A WORD/TOKEN?
 - INDEX IN A DICTIONARY?
- ONE-HOT ENCODING 



- LARGE DICTIONARY \Rightarrow LARGE ONE-HOT EMBEDDING
 - BUT VERY SPARSE
- DENSE BUT LOWER DIMENSIONAL REPRESENTATION OF A WORD? (LATER)

TEXT AS A SEQUENCE OF TOKENS

- ASSUME THE VOCABULARY SIZE IS $V=10000$
 - 10000 FEATURES PER TOKEN
- ASSUME TEXT CONSISTS OF $L=20000$ TOKENS
 - $10k \times 20k = 200M$ INPUT FEATURES
- STATISTICALLY AND COMPUTATIONALLY PROHIBITIVE
 - E.G., OVERFITTING
- ALSO, WHAT TO DO WITH THE VARIABLE LENGTH (L)?

$A \rightarrow 10, Book = 30$

BAG OF WORDS MODEL

- FORGET ABOUT THE ORDER OF WORDS
- USE JUST THE SET/COUNTS OF WORDS IN A DOCUMENT

- ONLY V FEATURES TO REPRESENT A TEXT
- GOOD FOR UNDERSTANDING THE GENERAL TOPIC
- BUT MAY NOT CAPTURE THE IMPORTANT CONTENT

$$V \times L \Rightarrow V$$

- {A, BOOK, BUT, DID, I, IT, IS, GOOD, LIKE, NOT, SOMEHOW, THINK, THIS} ✓

- {A=1, BOOK=1, BUT=1, DID=1, I=2, IT=1, IS=1, GOOD=1, LIKE=1, NOT=1, SOMEHOW=1, THINK=1, THIS=1}

- TWO DIFFERENT MEANINGS:

- {I THINK THIS IS A GOOD BOOK BUT SOMEHOW I DID NOT LIKE IT}
- {I THINK THIS IS NOT A GOOD BOOK BUT SOMEHOW I DID LIKE IT}

N-GRAM APPROACH

- CONSIDER N CONSECUTIVE WORDS/TOKENS AS ONE TOKEN
 - ADDS A BIT OF CONTEXT TO THE BAG OF WORDS
- N=2 EXAMPLE: {[A GOOD], [BOOK BUT], [BUT SOMEHOW], [DID NOT], [GOOD BOOK], [I DID], [I THINK], [IS A], [LIKE IT], [NOT LIKE], [THINK THIS], [THIS IS], [SOMEHOW I]}
- "I THINK THIS IS A GOOD BOOK BUT SOMEHOW I DID NOT LIKE IT"
- GENERALIZES THE TWO PREVIOUS IDEAS (N=1 OR N=L) $N \leq 10$
 - LARGE N IS COMPUTATIONALLY AND STATISTICALLY PROHIBITIVE

input size $\approx N \cdot V$

TEXT AS A SEQUENCE

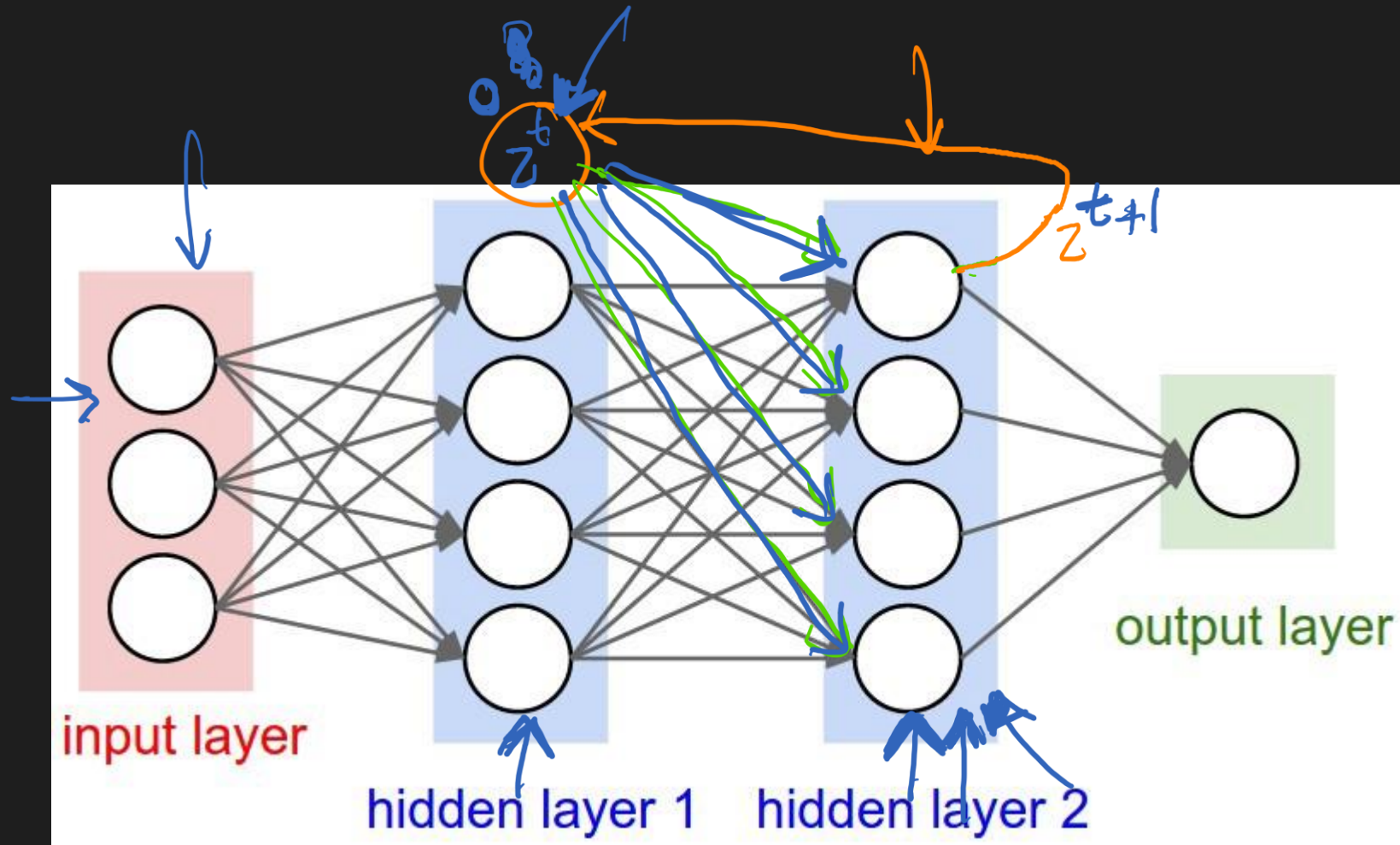
- REPRESENT DOCUMENTS AS A SEQUENCE...BUT KEEPING THE NUMBER OF FEATURES/PARAMETERS MANAGEABLE?
- ✗• CONVNET FOR TEXT CLASSIFICATION
- ✗• RECURRENT NEURAL NETWORKS..
- ✗• ATTENTION NETWORKS..
 - TRANSFORMERS

LIMITATIONS OF CNN FOR TEXT PROCESSING

DIFFERENCES BETWEEN IMAGES AND TEXT

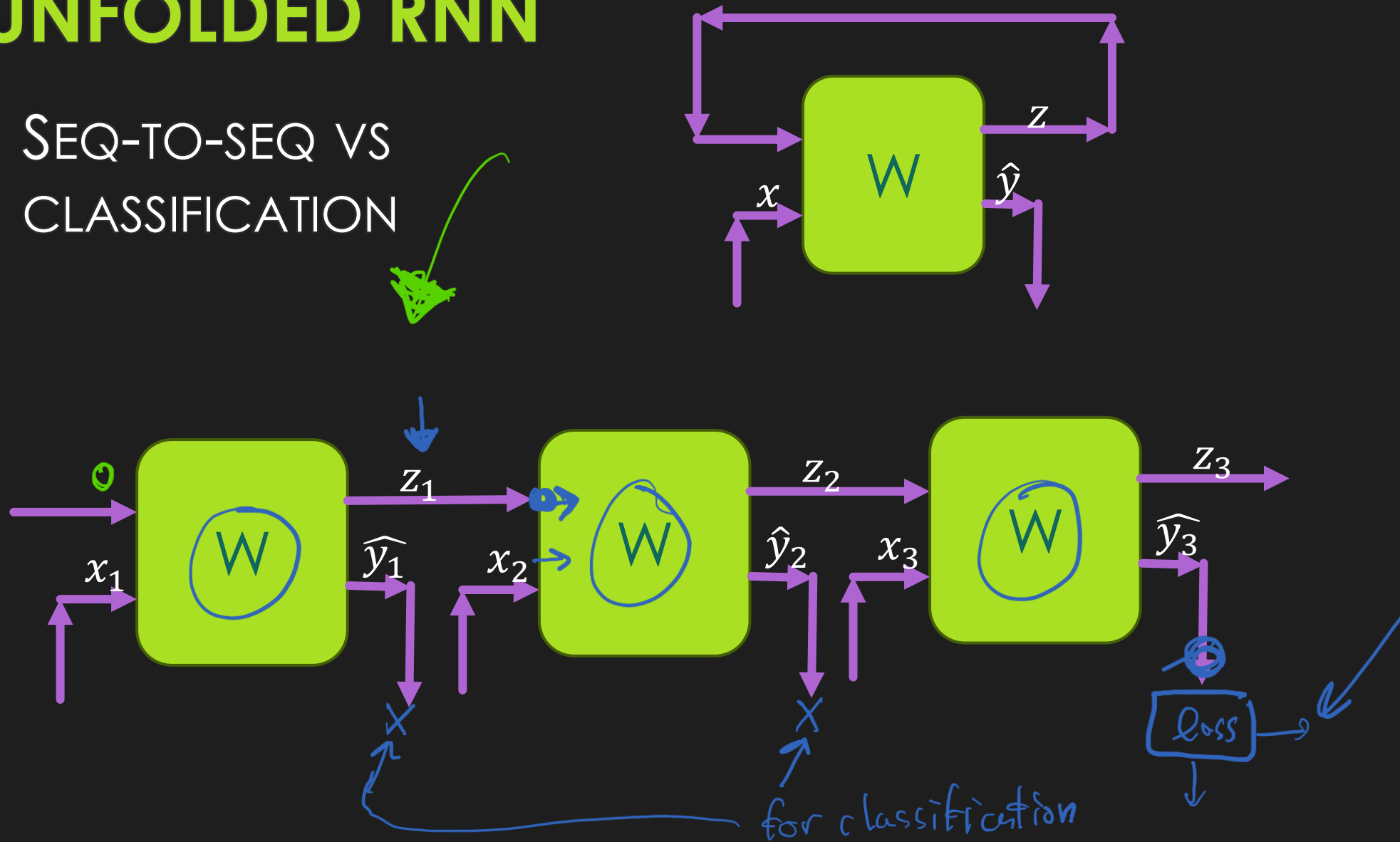
- WORDS ARE DISCRETE, VALUE IS NOT MEANINGFUL
 - UNLIKE PIXELS
- NEIGHBORING WORDS ARE NOT USUALLY SIMILAR
 - UNLIKE PIXELS IN AN IMAGE
- A SINGLE WORD CAN MAKE A HUGE DIFFERENCE
 - “NOT”
 - UNLIKE PIXELS
- DOCUMENTS HAVE VARIABLE LENGTHS
 - IMAGES CAN BE SCALED TO BE THE SAME SIZE

RECURRENT NEURAL NETWORKS

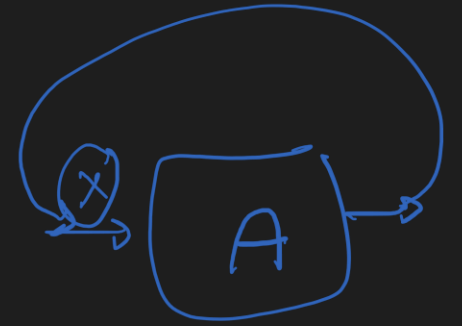


UNFOLDED RNN

- SEQ-TO-SEQ VS CLASSIFICATION



RECURRENT NEURAL NETWORKS



AAAAAAx

- THEY CAN “REMEMBER” THE PAST
- WE CAN GIVE A DOCUMENT WORD-BY-WORD
 - CAN HANDLE VARIABLE LENGTH INPUT
- THEY CAN SUFFER FROM THE PROBLEM OF VANISHING (OR EXPLODING!) GRADIENTS MORE THAN FEED-FORWARDS...
 - FEEDBACK LOOP!
 - TOP EIGEN VALUE OF A LINEAR CONTROL SYSTEMS
 - $\gg 1$: DIVERGENCE (EXPLOSION)
 - $\ll 1$: QUICK CONVERGENCE (SHORT MEMORY)
 - THEY ARE SLOW TO TRAIN (WHY?)
 - ALSO HARD TO PARALLELIZE

Long Short-Term Memory

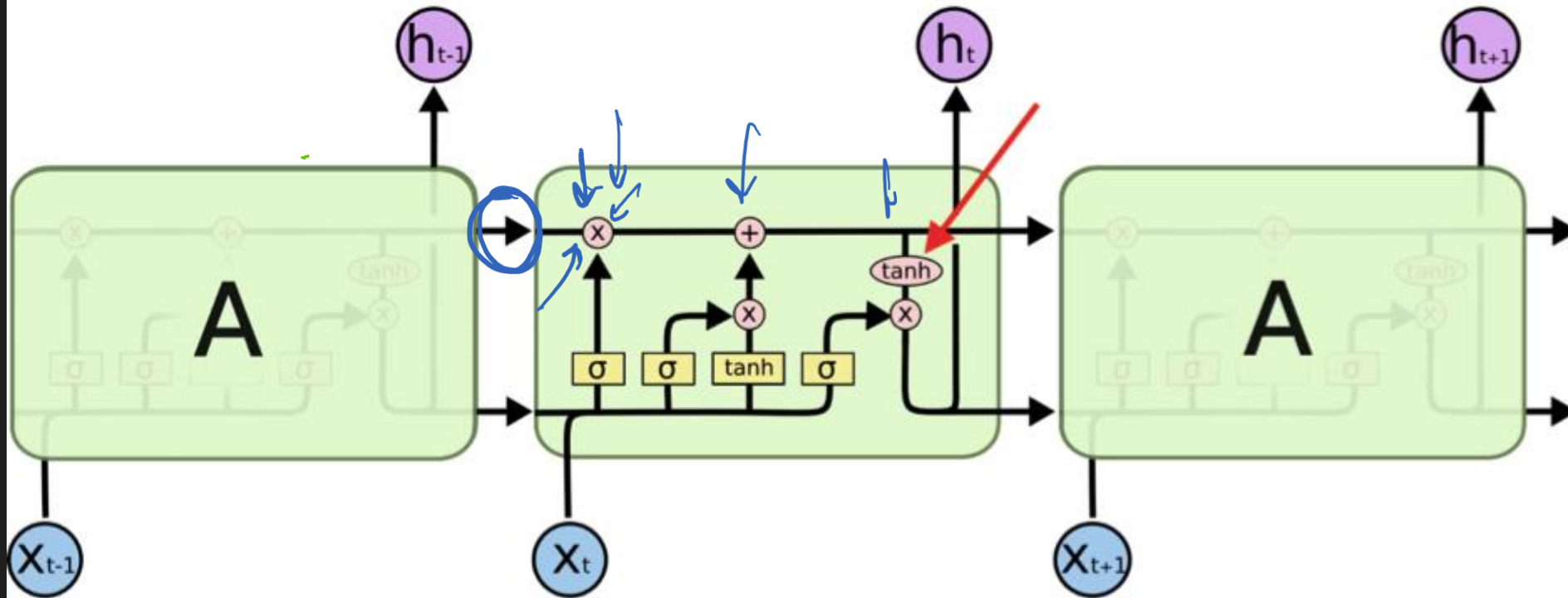


Image Source: colah.github.io

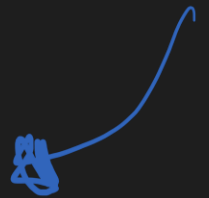
- LSTMS CONTROL THE FEEDBACK, AND ARE MORE ROBUST TO VANISHING/EXPLODING GRADIENT ISSUES
- BETTER “MEMORY” AND LONGER CONTEXT LENGTH

DENSE REPRESENTATIONS OF A WORD?

- CAN WE EMBED WORDS IN A LOW DIMENSIONAL SPACE (E.G., 100 DIMENSIONAL) SUCH THAT
 - CLOSE WORDS HAVE CLOSE MEANING?
- CAN WE USE THE VAST UNLABELED TEXT THAT IS AVAILABLE, E.G., ON THE INTERNET?
- TRAIN A LANGUAGE MODEL?

WORD2VEC

- GIVEN A LARGE TEXT CORPUS, TRAIN A NETWORK THAT
 - GIVEN 3 WORDS AFTER AND 3 WORDS BEFORE A WORD, PREDICTS THAT WORD (OR VICE VERSA)
- AFTER TRAINING, USE THE INTERMEDIATE FEATURES TO REPRESENT EACH WORD!

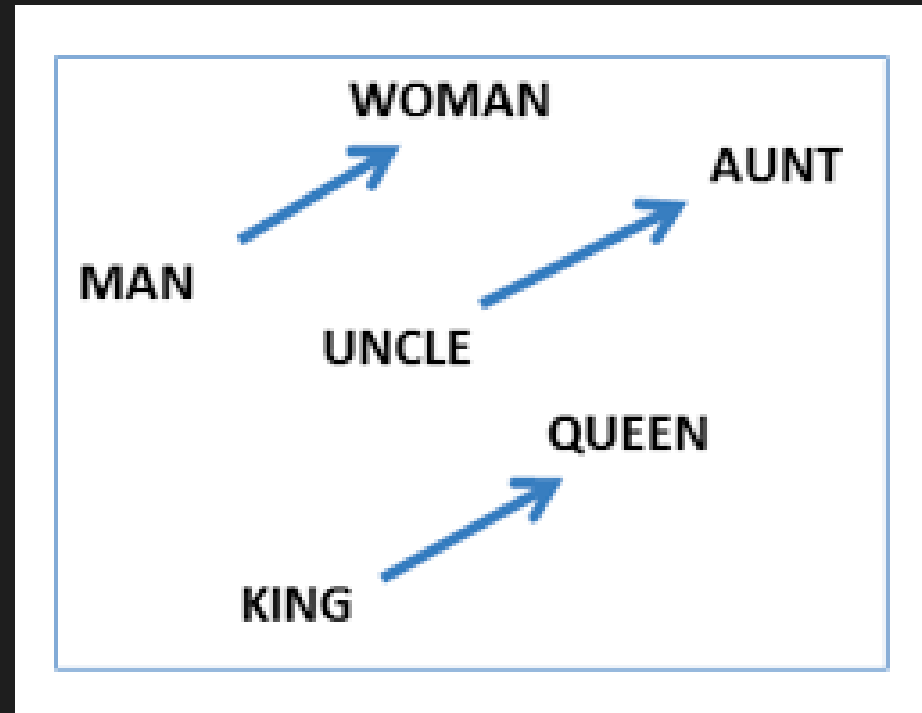


the raw data). However, natural processing systems traditionally treat words as discrete atomic symbols, and therefore 'cat' may be represented as `Id537` and 'dog' as `Id143`. These encodings are arbitrary, and provide no information to the system regarding the relationships that may exist between the individual symbols. This means that model can leverage very little of

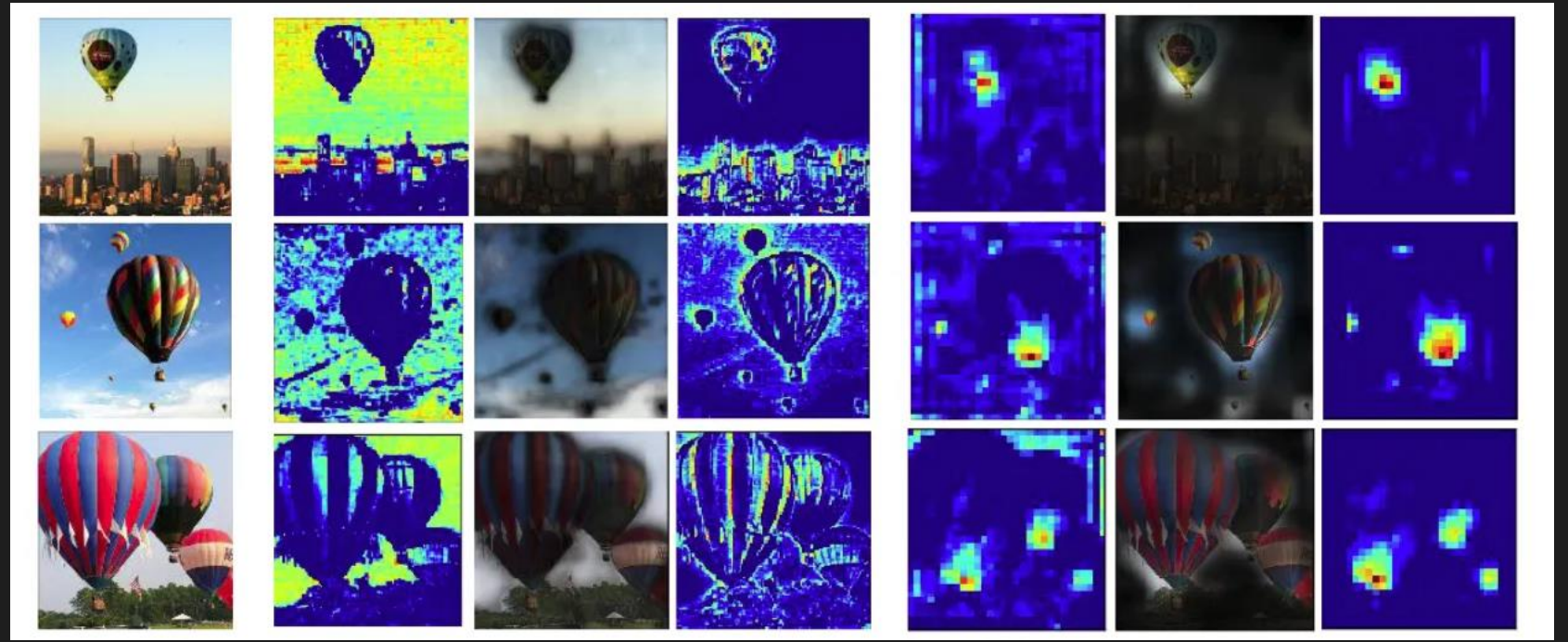
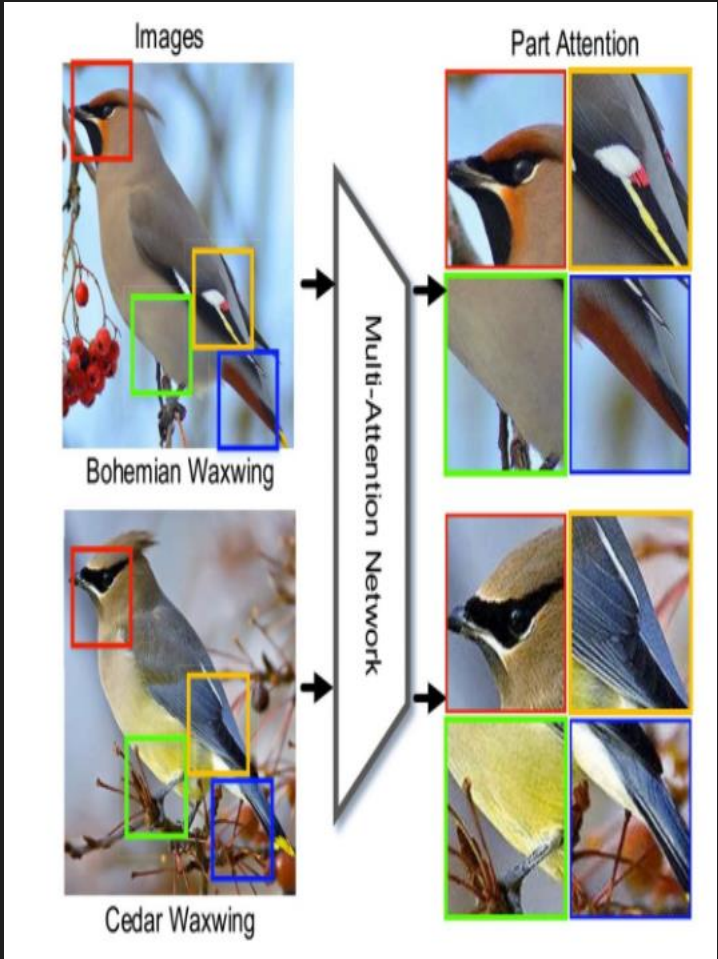
WORD2VEC

- KING - MAN + WOMAN \approx QUEEN!

Final
Until
Here

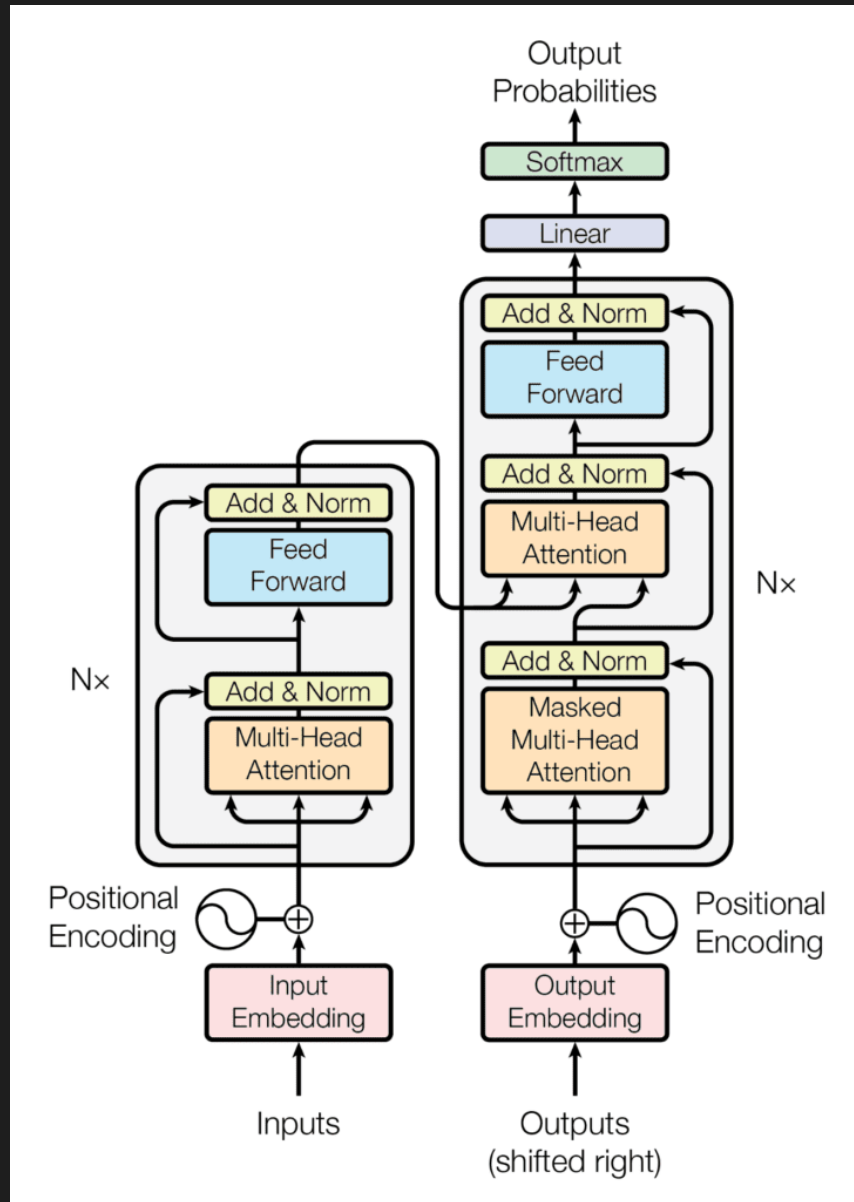


ATTENTION



TRANSFORMERS

- GET RID OF THE RECURRENCE



- x^t : INPUT WORD $t \in [T]$
- $z^t = emb(x^t)$: WORD EMBEDDING
- $q^t = W_q z^t + b_q$: QUERY , $q \in R^{d_1}$
 - $Q = [q^1 | q^2 | \dots]^T$: QUERY MATRIX, $T \times d_1$
- $k^t = W_k z^t + b_k$: KEY , $k \in R^{d_1}$
 - $K = [k^1 | k^2 | \dots]^T$: KEY MATRIX, $T \times d_1$
- $v^t = W_v z^t + b_v$: VALUE , $v \in R^{v_1}$

- $\alpha^{i,j} = \frac{e^{\langle q^i, k^j \rangle / \sqrt{d_1}}}{\sum_j e^{\langle q^i, k^j \rangle / \sqrt{d_1}}}$

- $v^i = \sum_j \alpha^{i,j} v^j$

- COMPUTE THE ATTENTION MATRIX IN ONE SHOT

- $\text{SOFT-MAX}\left(\frac{QK^T}{\sqrt{d_1}}\right)$

OTHER THINGS

- MULTI-HEAD ATTENTION
- POSITIONAL EMBEDDING
- MASKING FOR SEQ-TO-SEQ TASKS