Question 3)

$$\text{Attention}(Q,K,V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \times V.$$

Q = query matrix. K = key matrix. V = value matrix. d_k = dimension.

$$K^T = \begin{bmatrix} 1.22 & 5.47 & 0.88 & 2.81 & 0.06 \\ 4.95 & 1.85 & 1.96 & 5.43 & 8.15 \\ 0.34 & 9.70 & 0.45 & 1.41 & 7.07 \\ 9.09 & 7.75 & 3.25 & 8.02 & 7.29 \\ 2.59 & 9.39 & 3.89 & 0.75 & 7.71 \\ 6.63 & 8.95 & 2.71 & 9.87 & 6.74 \\ 3.12 & 5.98 & 8.29 & 7.72 & 3.58 \\ 5.20 & 9.22 & 3.57 & 1.99 & 1.16 \end{bmatrix}$$

QK^T.

Row 1:

$(1,1)$: $3.75 \times 1.22 + 9.51 \times 4.95 + 7.32 \times 0.34 + 5.99 \times 9.09 + 1.56 \times 2.59 +$
$1.56 \times 6.63 + 0.58 \times 3.12 + 8.66 \times 5.20$.

$= 161.72$.

Row $(1,2)$: $3.75 \times 5.47 + 9.51 \times 1.85 + 7.32 \times 9.70 + 5.99 \times 7.75 + 1.56 \times 9.39 +$
$1.56 \times 8.95 + 0.58 \times 5.98 + 8.66 \times 9.22$.

$= 251.88$

Row $(1,3)$: $3.75 \times 0.88 + 9.51 \times 1.96 + 7.32 \times 0.45 + 5.99 \times 3.25 + 1.56 \times 3.89 + 1.56 \times 2.71$
$+ 0.58 \times 8.29 + 8.66 \times 3.57$.

$= 88.76$

Row $(1,4)$: $3.75 \times 2.81 + 9.51 \times 5.43 + 7.32 \times 1.41 + 5.99 \times 8.02 + 1.56 \times 0.75 + 1.56 \times 9.87 +$
$0.58 \times 7.72 + 8.66 \times 1.99$.

$= 150.96$

Row $(1,5)$: $3.75 \times 0.06 + 9.51 \times 8.15 + 7.32 \times 7.07 + 5.99 \times 7.29 + 1.56 \times 7.71 + 1.56 \times 0.74 +$
$0.58 \times 3.58 + 8.66 \times 1.16$.

$= 173.50$

Row 2:-

(2,1) : 6.01×1.22 + 7.08×4.95 + 0.21×0.34 + 9.70×9.09 + 8.32×2.59 + 2.12×6.63 + 1.82×3.12 + 1.83×5.20.

= 175.03 .

(2,2) : 6.01×5.47 + 7.08×1.85 + 0.21×9.70 + 9.70×7.75 + 8.32×9.39 + 2.12×8.95 + 1.82×5.98 + 1.83×9.22.

= 245.96 .

(2,3) : 6.01×0.88 + 7.08×1.96 + 0.21×0.45 + 9.70×3.25 + 8.32×3.89 + 2.12×2.71 + 1.82×8.29 + 1.83×3.57.

= 102.71 .

(2,4) : 6.01×2.81 + 7.08×5.43 + 0.21×1.41 + 9.70×8.02 + 8.32×0.75 + 2.12×9.87 + 1.82×7.72 + 1.83×1.99.

= 175.05 .

2,5 : 6.01×0.06 + 7.08×8.15 + 0.21×7.07 + 9.70×7.29 + 8.32×7.71 + 2.12×0.74 + 1.82×3.58 + 1.83×1.16.

= 198.26

Row 3.

(3,1) : 3.04×1.22 + 5.25×4.95 + 4.32×0.34 + 2.91×9.09 + 6.12×2.59 + 1.39×6.63 + 2.92×3.12 + 3.66×5.20.

= 112.37 .

(3,2) : 3.04×5.47 + 5.25×1.85 + 4.32×9.70 + 2.91×7.75 + 6.12×9.39 + 1.39×8.95 + 2.92×5.98 + 3.66×9.22.

= 187.05 .

(3,3) : 3.04×0.88 + 5.25×1.96 + 4.32×0.45 + 2.91×3.25 + 6.12×3.89 + 1.39×2.71 + 2.92×8.29 + 3.66×3.57.

= 81.20 .

(3,4) : 3.04×2.81 + 5.25×5.43 + 4.32×1.41 + 2.91×8.02 + 6.12×6.75 + 1.39×9.87 + 2.92×7.72 + 3.66×1.99 =

= 112.68 .

3,5 : 3.04×0.06 + 5.25×8.15 + 4.32×7.07 + 2.91×7.29 + 6.12×7.71 + 1.39×0.74 + 2.92×3.58 + 3.66×1.6 .

= 142.84

Row 4:-

(4,1) :- 4.56×1.22 + 7.85×4.95 + 2.00×0.34 + 5.14×9.09 + 5.92×2.59 + 0.46×6.63 +
6.08×3.12 + 1.71×5.20.
= 135.95.

(4,2) :- 4.56×5.47 + 7.85×1.85 + 2.00×9.70 + 5.14×7.75 + 5.92×9.39 + 0.46×8.95
+ 6.08×5.98 + 1.71×9.22.
= 199.68.

(4,3) : 4.56×0.88 + 7.85×1.96 + 2.00×0.45 + 5.14×3.25 + 5.92×3.89 + 0.46×2.71 +
6.08×8.29 + 1.71×3.57.
= 109.68.

(4,4) : 4.56×2.81 + 7.85×5.43 + 2.00×1.41 + 5.14×8.02 + 5.92×0.75 + 0.46×9.87 +
6.08×7.72 + 1.71×1.99.
= 143.96.

(4,5) : 4.56×0.06 + 7.85×8.15 + 2.00×7.07 + 5.14×7.29 + 5.92×7.71 + 6.46×0.74
+ 6.08×3.58 + 1.71×1.16
= 163.84.

Row 5:-

(5,1) : 0.65×1.22 + 9.49×4.95 + 9.66×0.34 + 8.08×9.09 + 3.05×2.59 + 0.98×6.63 +
6.84×3.12 + 4.40×5.20.
= 175.68

(5,2) : 0.65×5.47 + 9.49×1.85 + 9.66×9.70 + 8.08×7.75 + 3.05×9.39 + 0.98×8.95 +
6.84×5.98 + 4.40×9.22.
= 258.42

(5,3) : 0.65×0.88 + 9.49×1.96 + 9.66×0.45 + 8.08×3.25 + 3.05×3.89 + 0.98×2.71 +
6.84×7.72 + 4.40×3.57.
= 124.77

(5,4) : 0.65×2.81 + 9.49×5.43 + 9.66×1.41 + 8.08×8.02 + 3.05×0.75 + 0.98×9.87 +
6.84 + 7.72 + 4.40×1.99 = 193.72.

(5,5) : 0.65×0.06 + 9.49×8.15 + 9.66×7.07 + 8.08×7.29 + 3.05×7.71 + 0.98×0.74
+ 6.84×3.58 + 4.40×1.16.
= 215.96

$Q K^T$ =

161.72	251.88	88.76	150.96	173.50
175.03	245.86	102.71	175.05	198.26
112.37	187.05	81.20	112.60	142.84
135.95	199.68	109.68	143.96	163.84
175.68	258.42	124.77	193.72	215.90

$\frac{Q K^T}{\sqrt{8}}$ =

57.19	89.08	31.39	53.38	61.36
61.91	86.99	36.32	61.92	70.12
39.74	66.14	28.72	39.85	50.51
48.08	76.62	38.79	50.89	57.94
62.13	91.46	44.13	68.52	76.34

Attention weights (after softmax):

0.000	1.000	0.000	0.000	0.000
0.000	1.000	0.000	0.000	0.000
0.000	1.000	0.000	0.000	0.000
0.000	1.000	0.000	0.000	0.000
0.000	1.000	0.000	0.000	0.000

Attention output = Attention weights × V(5,8).

Final Attention:

8.87	4.72	1.20	7.13	7.61	5.61	7.71	4.94
8.87	4.72	1.20	7.13	7.61	5.61	7.71	4.94
8.87	4.72	1.20	7.13	7.61	5.61	7.71	4.94
8.87	4.72	1.20	7.13	7.61	5.61	7.71	4.94
8.87	4.72	1.20	7.13	7.61	5.61	7.71	4.94

Question 4)

i) Without attention, the decoder only receives one final hidden state from the encoder, a single compressed summary of the entire input sequence.

With attention the decoder can directly access all encoder hidden states, any specific time step of the input and fine grained contextual details relevant to the current output token. This matters because the decoder no longer relies on only one vector; it can look back at the entire input sequence as needed.

ii) In RNNs, information must travel through many timesteps, gradients weaken and long term memory fades. Attention solves this because it creates direct connections between decoder steps and all encoder steps. The model does not need to store long range info inside hidden states; the attention layer retrieves it instantly. So even architecture remains same, the ~~attention layer~~ pathway for information becomes much shorter.

iii) Uniform weights mean each encoder time step contributes equally and the decoder fails to focus on any specific part of input. This failure mode is called attention collapse.

iv) Soft attention uses softmax as weights are continous where as hard attention selects one encoder position. Soft attention is most commonly used and hard attention is more computationally difficult.
The training implication is soft attention is easy to optimize, hard attention is unstable and slow but sometimes more interpretable.

v) Global attention, Decoder can attend to any encoder position.
Monotonic attention, Attention moves forward only, never backward. It is useful for speech, ASR and alignment task.

This is helpful in speech because speech signals are naturally sequential; monotonicity reduces confusion and stabilizes training.

vi) RNN works well without attention when input sequences are short (e.g ≤ 10 tokens) and tasks require global understanding, not precise alignment. Example: sentence classification.
Attention becomes essential when inputs are long (machine translation, summarization). Model must align input and output. Example: image captioning.

vii) LSTMs solve forgetting using gates, but they still compress the entire sequence into one vector. For long sequences, this bottleneck vector becomes overloaded. Attention removes the bottleneck by allowing direct lookup into all encoder states. Reducing the burden on LSTM memory.

viii) Before attention decoder state must store all context from the entire input. With attention decoder state only stores local decoding info. Global context comes from attention distribution.

ix) Attention improves gradient flow because it creates shortcut paths directly from decoder to encoder states. Gradients don't need to backprop through long chains of timesteps. This leads to training becomes more stable and long range dependencies are easier to learn.

x) Attention score - $\text{score} = QK^T$.

Scaled dot product attention $a = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$.

Context vector $C = aV$.

Softmax.

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_i e^{x_i}}$$

xi) Two main conceptual failure cases.

1) Encoder representation are poor.

Even a deep RNN can: over compress, loss information and produce similar hidden states at every timestep.

2) Decoder ignores the attention.

Sometimes the decoder learns to rely on its own hidden state. Treats attention as noise. Sets attention weights almost uniform.

xii) Multi-head attention is better when input has multiple types of features. Model to focus on different positions simultaneously. Task requires analyzing relationships at different scales.

Examples: Translation (syntax head + semantic head), speech recognition.

The intuition is that each head becomes a specialist, seeing the sequence from a different perspective.

xiii) Two internal causes.

1) Encoder states are too similar.

If all encoder hidden states look alike then attention has no reason to pick one. Best solution is to focus on one position everytime.

2) Decoder is too strong.

If decoder LSTM learns to predict well from previous outputs, its hidden state, final encoder state. Then attention becomes unnecessary.

The loss can still be low as training data may be predictable and decoder may memorize patterns and even wrong attention can still produce correct outputs. But generalization becomes poor as model fails on unseen patterns.