Q6: $(0.88 + \underbrace{112}_{V_1}) \times (0.12 + \underbrace{96}_{V_2}) = 10,850.0256$. by V_1 和 V_2 衍生.

QKV 的關係和 Z 的關係，用單頭注意力解釋，並說明 Z 是怎麼 derived

1. input sequences: thinking, machine
2. embed each words → X_1 2×4, X_2
3. 計算單頭注意力 → W^Q, W^K, W^V 4×3
4. 得自注意力 → Q_0, K_0, V_0 2×3
5. $\text{Softmax}\left(\frac{QK}{\sqrt{d_K}}\right) \cdot V$ 得 Z → Z_0 2×3
6. 把多頭 Z 連起來 (相加) × W^O 得 Z

Z_0 Z_1 Z_2 Z_3 Z_4 … × (□×4) = Z 2×4

2×□

意旨計算詞在所有 thinking 單頭自注意力中的 Value 值，成多頭的 Z 值.

透過將 one-head attention 的結果彙總產生 multi-self attention，是為了得到多個 self attention 間不同詞間相關性和上下文意完整理解. 透過將詞嵌入向量得 X_1 X_2 並將詞向量 x_1 / x_2 分別乘上 $W_{q/k/v}$ (Matrixs) 得到 Head 0 的 Q_0 / K_0 / V_0 Self-attention

→ 再 $\text{Softmax}\left(\frac{Q \cdot K}{\sqrt{d_K}}\right) \times V_0$ 得 Z。

→ 把所有單頭 Self-attention 的 Z 結起來 × W^O 得 Z，就可以計算所有詞在不同 self-attention 的值由多頭

故 $Z_1 = V_1 \cdot \text{softmax}(0.88) + V_2 \cdot \text{softmax}(.12)$

= 詞 1 (thinking) 在 thinking 和 machine 中產生的相關性和上下文意都被考慮.

矩陣&向量的計算

1. 一維向量 × 一維向量　先唸直 再唸橫

$[1,0,0] \times [1,2,0]$

將上述轉行列相乘 → $\underset{1\times 3}{[1,0,0]} \times \underset{3\times 1}{\begin{bmatrix}1\\2\\0\end{bmatrix}} = \underset{(1\times 1)}{1}$

$1\times 1 + 0\times 2 + 0\times 0 = 1$

遇到矩陣相同要轉置

2. 不同大小矩陣相乘　☆算法2較簡單

算法一

$\underset{1\times 2}{\begin{bmatrix}1 & 2\end{bmatrix}} \times \underset{2\times 4}{\begin{bmatrix}1 & 3 & 2 & 1\\2 & 3 & 1 & 2\end{bmatrix}} = \underset{1\times 4}{\begin{bmatrix}5 & 9 & 4 & 5\end{bmatrix}}$

$1\times 1 + 2\times 2 = 5$，$1\times 3 + 2\times 3 = 9$，$1\times 2 + 2\times 1 = 4$，$1\times 1 + 2\times 2 = 5$

or $\begin{bmatrix}1\end{bmatrix} \times \begin{bmatrix}1 & 3 & 2 & 1\end{bmatrix} + \begin{bmatrix}2\end{bmatrix} \times \begin{bmatrix}2 & 3 & 1 & 2\end{bmatrix}$

$= \begin{bmatrix}1 & 3 & 2 & 1\end{bmatrix} + \begin{bmatrix}4 & 6 & 2 & 4\end{bmatrix} = \begin{bmatrix}5 & 9 & 4 & 5\end{bmatrix}$

更大的矩陣相乘也要記得 前面橫的一列一列處理.

$\underset{\downarrow 3\times \vec{2}}{\begin{bmatrix}1 & 2\\2 & 1\\3 & 4\end{bmatrix}} \times \underset{\downarrow 2\times \vec{4}}{\begin{bmatrix}1 & 3 & 2 & 1\\2 & 3 & 1 & 2\end{bmatrix}} = \underset{3\times 4}{\begin{bmatrix}5 & 9 & 4 & 5\\4 & 9 & 5 & 4\\11 & 21 & 10 & 11\end{bmatrix}}$

算法一

→ $2\times 1 + 1\times 2 = 4$，$2\times 3 + 1\times 3 = 9$，$2\times 2 + 1\times 1 = 5$，$2\times 1 + 1\times 2 = 4$

↓ $3\times 1 + 4\times 2 = 11$，$3\times 3 + 4\times 3 = 21$，$3\times 2 + 4\times 1 = 10$，$3\times 1 + 4\times 2 = 11$

算法二

$\underset{\downarrow 3\times \vec{2}}{\begin{bmatrix}1 & 2\\2 & 1\\3 & 4\end{bmatrix}} \times \underset{\downarrow 2\times \vec{4}}{\begin{bmatrix}1 & 3 & 2 & 1\\2 & 3 & 1 & 2\end{bmatrix}} = \underset{\downarrow 3\times \vec{4}}{\begin{bmatrix}5 & 9 & 4 & 5\\4 & 9 & 5 & 4\\11 & 21 & 10 & 11\end{bmatrix}}$

1. 一個配一列

$\begin{bmatrix}1\end{bmatrix} \times \begin{bmatrix}1 & 3 & 2 & 1\end{bmatrix} + \begin{bmatrix}2\end{bmatrix} \times \begin{bmatrix}2 & 3 & 1 & 2\end{bmatrix}$

$= \begin{bmatrix}1 & 3 & 2 & 1\end{bmatrix} + \begin{bmatrix}4 & 6 & 2 & 4\end{bmatrix}$

$= \begin{bmatrix}5 & 9 & 4 & 5\end{bmatrix}$

2.

$\begin{bmatrix}2\end{bmatrix} \times \begin{bmatrix}1 & 3 & 2 & 1\end{bmatrix} + \begin{bmatrix}1\end{bmatrix} \times \begin{bmatrix}2 & 3 & 1 & 2\end{bmatrix}$

$= \begin{bmatrix}2 & 6 & 4 & 2\end{bmatrix} + \begin{bmatrix}2 & 3 & 1 & 2\end{bmatrix}$

$= \begin{bmatrix}4 & 9 & 5 & 4\end{bmatrix}$

3

$\begin{bmatrix}3\end{bmatrix} \times \begin{bmatrix}1 & 3 & 2 & 1\end{bmatrix} + \begin{bmatrix}4\end{bmatrix} \times \begin{bmatrix}2 & 3 & 1 & 2\end{bmatrix}$

$= \begin{bmatrix}3 & 9 & 6 & 3\end{bmatrix} + \begin{bmatrix}8 & 12 & 4 & 8\end{bmatrix}$

$= \begin{bmatrix}11 & 21 & 10 & 11\end{bmatrix}$

Transformer Single Embedding 嵌入.

前由 input → input embedding → encoding 編碼馬

1. Input Embedding 嵌入：將單詞轉換為一系列數列（向量），並將向量做為處理和理解語言的輸入（稱為編碼器）. Dimensional 維度是自設的.

input Embedding →

Cat	is	sleeping	on	the	mat
.54	.76	.23	.89	.1	.67
.31	.93	.68	.45	.82	.17
.40	.56	.72	.64	.20	.95
.72	.89	.18	.31	.52	.39
.62	.87	.04	.78	.34	.52

} 預設5維 / 5 dim

2. Positional Encoding 位置編碼：讓 Model 知道每個詞的順序，Transformer 是一整句一起理解，並會考慮「詞 & 詞互相影響」（似上下文）
   ↓ 相關程度.

★PE 有2公式 For even position / For old position，不過應該題目會給.

3. Adding "Word Embedding" & "Positional Embedding"

input embedding							positional embedding					
for ex. Cat	is	sleeping	on	the	mat.	+ Cat	is	sleeping	on	the	mat.
.54	.76	.23	.89	.1	.67		.0	.841	.90	.141	-.75	-.95
⋮							⋮					

= ★ Cat is sleeping on the mat ★

.54	1.601	1.13	1.031	-0.65	-0.28
⋮					

4. Multi-self attention：Matrix 相乘 QKV 考慮上下文、詞間互相影響.

計算/考慮句子中某個詞與其他部份的相關程度.

4. Multi-self attention

所有的 Word Embedding + Positional Embedding 都相同，不同的是 QKV 的 linear weights.

Cat is sleeping on the mat

.54	1.601	1.13	1.031	-0.65	-0.28
1.31	1.917	1.63	1.33	1.625	0.87
.40	.566	0.721	0.691	0.202	0.953
1.72	1.889	1.17	1.309	1.509	1.329
.62	4.85	1.77	1.97	1.932	2.51

5×6 Matrixs

linear Weights 權重 weight 再相加

.67	.14	.71	.03
.85	.31	.14	.95
.24	.57	.76	.93
.39	.95	.59	.38
.64	.94	.77	.51
.94	.73	.17	.02

6×4 Matrix

=

Query 5×4 Matrixs

1.71	1.38	1.52	2.64
5.27	5.13	4.62	4.72
2.19	2.13	1.60	1.58
5.76	5.12	4.53	4.22
9.32	8.11	5.54	8.05

1 Query 計算

直向相加

.54 × (.67 .14 .71 .03) = (.362 .076 .383 .016)

+ 1.601 × (.85 .31 .14 .95) = (1.361 0.50 .224 1.521)

+ 1.13 × (.24 .57 .76 .93) = (0.271 .644 .860 1.051)

+ 1.031 × (.39 .95 .59 .38) = (.402 .980 .610 .392)

+ -0.65 × (.64 .94 .77 .51) = (-0.416 -0.611 -501 -.033)

+ -0.28 × (.94 .73 .17 .02) = (-.263 -.204 -.048 -0.0056)

Query 1

= 1.71 1.38 1.52 2.64

2 Key 計算

只算第一行

.54 × (.50 .76 .28 .12) = (.27 .41 .151 .065)

+ 1.601 × (.35 .33 .63 .25) = (.56 .53 1.01 .40)

+ 1.13 × (.79 .99 .04 .95) = (.90 1.12 0.05 1.07)

+ 1.031 × (.03 .09 .23 .09) = (.03 .09 .24 .09)

- 0.65 × (.85 .04 .09 .93) = (-0.55 -0.03 -.01 -.28)

- 0.28 × (.14 .63 .30 .21) = (.04 -.18 -.09 -0.06)

Key

1.17 1.94 1.35 1.285

≒ 1.16 ≒ 1.29

3. Value 計算

→ 老師給的公式錯誤.

.54 × (.50 .76 .28 .12) = (0.27 .410 .151 0.065)

+ 1.601 × (.35 .33 .63 .25) = (0.560 0.528 1.009 0.400)

+ 1.13 × (.79 .99 .04 .95) = (.892 1.119 0.045 1.074)

+ 1.031 × (.03 .09 .23 .09) = (0.031 0.093 0.237 0.093)

- 0.65 × (.85 .04 .09 .93) = (-.553 -0.026 -0.00585 -.280)

- 0.28 × (.14 .63 .30 .21) = (.039 .176 -0.006 .059)

.084

反正得出的 Query / Key / Value

5、Attention(Q, K, V) = softmax$\left(\frac{QK^T}{\sqrt{d_K}}\right)V$

Query 5×4 Key 5×4 ⟶ Key矩陣要轉置 Transpose 成4×5

1.71	1.38	~~5.12~~ 1.52	2.64
5.27	5.13	4.62	4.72
2.19	2.13	1.60	1.58
5.76	5.12	4.53	4.22
9.32	8.11	5.54	8.05

×

1.16	1.94	1.35	1.29
4.15	3.97	2.22	3.18
1.29	1.87	0.93	1.21
3.95	4.10	2.43	2.83
5.45	5.65	4.52	4.50

1.16	4.15	1.29	3.95	5.45
1.94	3.97	1.87	4.10	5.65
1.35	1.87	0.93	2.43	4.52
1.29	4.10	1.21	4.52	4.50

只求第一行

~~1.71×(1.16 1.94 1.35 1.29)~~
~~1.38×(4.15 3.97 2.22 3.18)~~
~~5.12×(1.29 1.87 0.93 1.21)~~ 1.52
+ ~~2.64×(3.~~

1.71×(1.16 4.15 1.29 3.95 5.45)
1.38×(1.94 3.97 1.87 4.10 5.65)
~~5.12~~ 1.52×(1.35 2.22 0.93 2.43 4.52)
+ 2.64×(1.29 3.18 1.21 2.83 4.50)
=

得 5×5 Matrix

10.1	24.3	9.39	23.5	35.8
28.3	67.5	26.3	66.4	99.8
10.8	26.1	10.2	25.7	38.3
28.1	67.7	26.3	66.6	99.7
44.4	108.7	42.08	106.3	157.8

☆ Scaling縮放，將上述 Q×K轉置結果的5×5 ÷ $\sqrt{d_K}$ → dimension 5

10.1	24.3	9.39	23.5	35.8
28.3	67.5	26.3	66.4	99.8
10.8	26.1	10.2	25.7	38.3
28.1	67.7	26.3	66.6	99.7
44.4	108.7	42.08	106.3	157.8

÷ $\sqrt{5}$ ($\sqrt{d_K} \leftarrow \sqrt{5}$)

= ($\frac{10.1}{\sqrt{5}}$ =) 四捨五入

4.52	10.87	4.20	10.51	16.01
12.66	30.19	11.76	29.70	44.63
4.83	11.67	4.56	11.49	17.13
12.57	30.28	11.76	29.78	44.59
19.86	48.61	18.82	47.54	70.57

} $\sqrt{5}$ 可以想成一條一個頭

☆ SoftMax = $\frac{e^{4.52}}{e^{4.52}+e^{10.87}+e^{4.20}+e^{10.51}+e^{16.01}}$ = 0.00001

(4.52 ← 當前；分母：橫全相加)

每一個重複softMax得 5×5 Matrix

Z2 = ↗第一個詞如Thinking在第一個self-attention的值

$Z_1 = V_1 \cdot softmax_1 + V_2 \cdot softmax_2$

softmax$\left(\frac{QK^T}{\sqrt{d_K}}\right)$ 5×5 Maxtrix × Value 5×4 = Softmax值 × V = Z

0.00001	0.0054	0.0007	0.00404	0.9904
0	0.0005	0	0.00003	0.9999
0.00045	0.004	0.0033	0.0036	0.9922
0	0.0005	0	0.00037	0.9999
0	0.0003	0	0.0001	1

×

.20	1.80	0.32	1.90
2.95	4.52	3.45	4.99
0.84	1.58	1.14	1.83
3.17	4.64	3.85	5.17
3.79	7.19	5.03	7.70

只算一行

0.00001×(.20 1.80 0.32 1.90)
0.0054×(2.95 4.52 3.45 4.99)
0.0007×(0.84 1.58 1.14 1.83)
0.00404×(3.17 4.64 3.85 5.17)
+ 0.9904×(3.79 7.19 5.03 7.70)
=

得 5×4 Matrix

3.78	7.16	5.01	7.67
3.79	7.19	5.03	7.70
3.78	7.17	5.02	7.68
3.79	7.19	5.03	~~0.017~~ 7.70
3.79	7.19	5.03	7.70

一個單頭注意力得一 5×4 Matiixs

三注意力得三 5×4 Matiixs

⇒ 5×12 Matiix

Transformer: 序列輸入序列輸出 模型結構 → input → encoder → context → decoder
(時有 hidden state) (vector) output

based on RNN, 但透過「self-attention」自注意力機制考慮句子中所有單字相關性和上下文，可以同時理解句子的所有部分，對整理語意更精準分析，克服 RNN 中長期依賴 (long-term dependency) 和計算效率不佳的限制

☆ Transformer 由編碼器 encoder 和解碼器 decoder 組成

☆ Encoder 編碼器 → 將輸入序列轉為連續形式，以獲得上下文語意，輸出通常為一系列嵌入向量，稱「隱藏狀態」或「上下文」。

☆ Decoder 解碼器 → 將 Encoder 輸出的隱藏狀態反覆計算成 token (令牌)

可能有三種 Type 的 Transformer

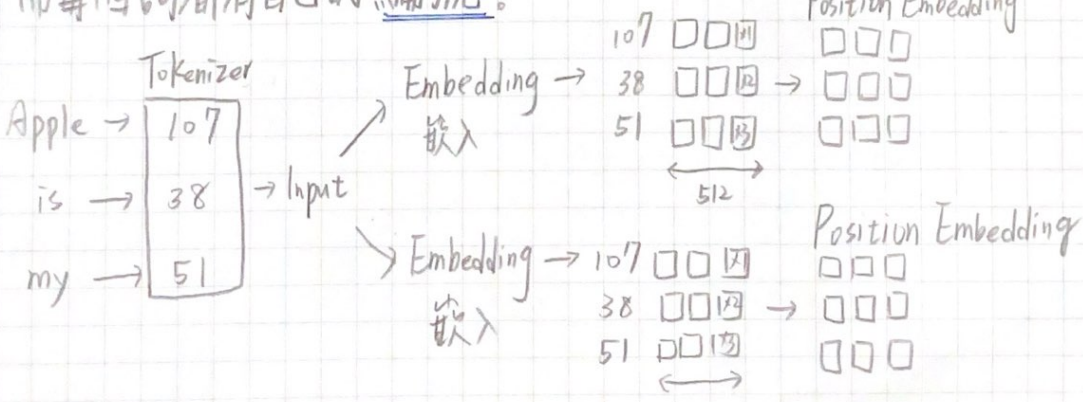
☆ 1. Encoder-only Transformers → 強在可深入理解和解釋文本，透過將輸入序列轉為各種數位形式，ex. BERT / RoBERTa / DistilBERT
使用雙向注意力 → 考慮前/後對內容。

☆ 2. Decoder-only Transformer → 此 Moder 強在強大的產出、表述能力，透過反覆計算預測下一個 P_{Max} 的單詞來完成輸出序列，or GPT。
而此 Moder 採 Autoregressive attention，僅考慮已輸出的上文。

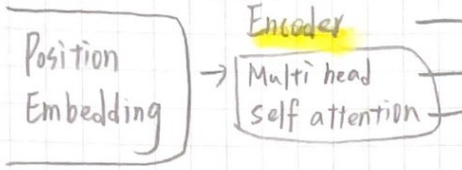
☆ 3. Encoder-decoder Transformer → 同時擁有輸入文本、理解語意、輸出回應的能力，如 T5 / BART，可用於翻譯和產生摘要。

☆ 標記化文本，

☆ Tokenizer 分詞器 → 使用 Tokenizer 將單詞轉為數位，因機器只懂數字，而每個詞都有自己的編號。



☆ Embedding 嵌入 → 負責將被標記成數字編號的詞轉成向量，一個標記為一個向量，而這些向量被存在一個「向量嵌入空間」，同時也會將單詞的位置紀錄下來（Position Embedding），方便文意理解。



☆ Encoder 編碼器 → 嵌入序列會 input 到編碼器中，這些 input 先經過 Multi head self-attention 多層注意力層，再到 Feed Forward，並輸入向量 (logits Vector)
↳ 就是把self-attention做多次，連接起來並把這向量×W^o得Z。

☆ Multi head self-attention → 句子中各個詞會互相影響，要用不同角度去分析語意，所以是「Milti head 多頭」

☆ Feed-Forward 前饋 → 由兩層全連接神經網路組成，單獨處理嵌入的序列，並由 Transformer 最終輸出層生成 logit，通常 FFD 使用 Gelu 做為激活函數（Gelu 讓數據可在線性和非線性中轉換）

☆ Decoder 解碼器 → 和編碼器相同，Model 通常有多個 Decoder 組成，Decoder 同樣將嵌入序列的 input 放到 Multi head self-attention 再將 input 傳到 FFD，Decoder 和 Encoder 主要差別在於，Decoder 解碼器有2個注意力子層。Masked multi-head self-attention 蒙面多頭注意力機制：避免偷看答案，確保生成的 token 僅基於「過去輸出(以產出的內容」和「當前預測」的 token。多頭同樣代表詞間互相影響，並讓 Model 考慮上下文意。

☆ Encoder-decoder attention → 讓 Model 專注在原文和譯文之間的轉換，此 decoder 會在輸出時考慮上下文以及目前為止輸入的內容，並輸出此時最相關的 Output (Token)
(P Max)

☆ Attention 機制：幫助機器提升翻譯質量

☆ Transformer：讓模型訓練並行，提升訓練效率