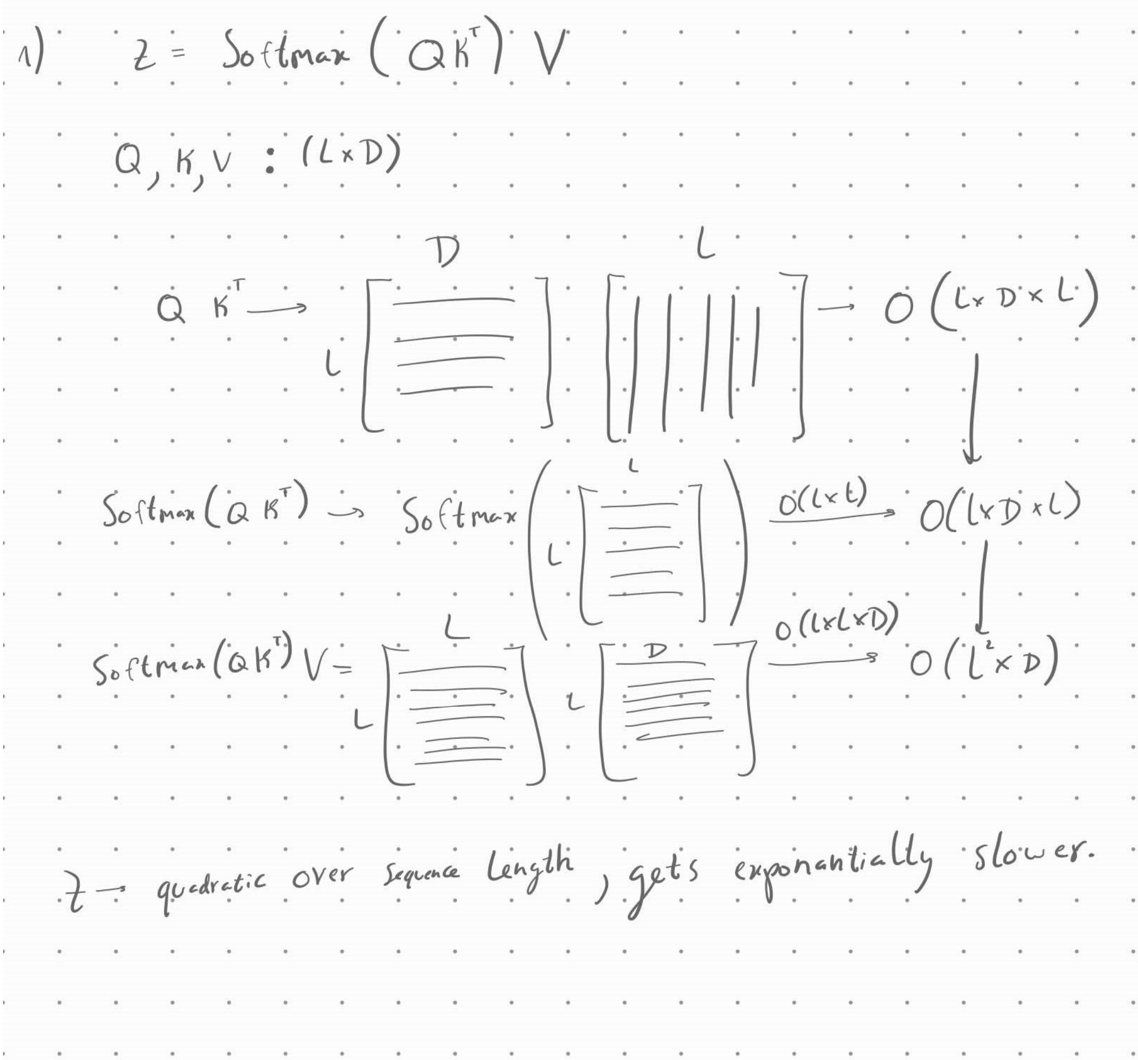
HomeWork 2

João André Roque Costa – 99088

Tomás Augusto Vilhena de Oliveira – 90781

Question 1

Q1.1



1) $Z = \text{Softmax}(QK^T)V$

$Q, K, V : (L \times D)$

$QK^T \rightarrow$ $\rightarrow O(L \times D \times L)$

$\text{Softmax}(QK^T) \rightarrow \text{Softmax}(\;) \xrightarrow{O(L \times L)} O(L \times D \times L)$

$\text{Softmax}(QK^T)V = \xrightarrow{O(L \times L \times D)} O(L^2 \times D)$

$Z \rightarrow$ quadratic over Sequence Length, gets exponantially slower.

Q1.2

2) $\begin{bmatrix} 1 & q^T K & \underbrace{\frac{q^T K^2}{2}} \end{bmatrix}$ $\quad D = 2, \quad q = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad K = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}$

$$\mathrm{trace}\,(q q^T K K^T) = \mathrm{trace}\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} \begin{bmatrix} Y_1 & Y_2 \end{bmatrix} \right)$$

$$= \mathrm{trace}\left(\begin{bmatrix} x_1^2 & x_1 x_2 \\ x_2 x_1 & x_2^2 \end{bmatrix} \begin{bmatrix} Y_1^2 & Y_1 Y_2 \\ Y_2 Y_1 & Y_2^2 \end{bmatrix} \right)$$

$\begin{bmatrix} 1 & , q^T K \end{bmatrix}$

$$= \mathrm{trace}\left(\begin{bmatrix} x_1^2 Y_1^2 + x_1 x_2 Y_1 Y_2 & \\ & x_1 x_2 Y_1 Y_2 + x_2^2 Y_2^2 \end{bmatrix} \right)$$

$$= \underbrace{x_1^2 Y_1^2 + x_2^2 Y_2^2 + 2 x_1 x_2 Y_1 Y_2}$$

$$\underbrace{\mathrm{Vec}(q q^T)^T}_{\phi(q)} \underbrace{\mathrm{Vec}(K K^T)}_{\phi(K)}$$

$$\phi(t) = \mathrm{Vec}\,(t t^T)$$

K	1	2	3	n
dim M	1	D	D^2	$\underline{\underline{D^{n-1}}}$

$\mathrm{Vec}\,(2[\,{}^1\,]\,1[\,{}^2\,])$

$\mathrm{vec}\,(2[\,{}^2\,])$

$1[\,{}^4\,]$

$$\mathrm{Dim}_M = D^{K-1}$$

Q1.3

3) $D^{-1}\, \Phi(Q)\, \Phi(K)^T V$

$$= \frac{\Phi(Q)\, \Phi(K)^T}{D} V$$

$$\approx \frac{\exp(QK^T)}{D} V$$

$$\approx \frac{\exp(QK^T)}{\mathrm{Diag}(\Phi Q\, \Phi K^T 1_L)} V$$

$$\approx \frac{\exp(QK^T)}{\mathrm{Diag}(\exp(QK^T)\, 1_L)} V$$

$$\approx \frac{\exp(QK^T)}{\sum_{i=1}^{L} \exp(q_i\, k_i^T)} V$$

$$\text{Softmax}(t_i) = \frac{\exp(t_i)}{\sum_{j=1}^{n} e^{t_i}}$$

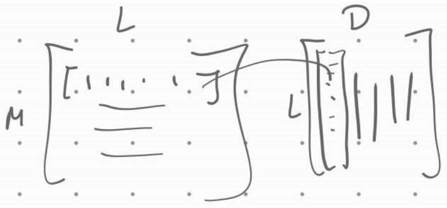
$$\approx \text{Softmax}(QK^T) V$$

$$Z = \text{Softmax}(QK^T) V$$

$$\approx Z$$

Q1.4

4) $$Z = \underbrace{\underbrace{D^{-1}}_{O(L\times M)} \underbrace{\underbrace{\phi(Q)}_{L\times M} \underbrace{\underbrace{\phi(K)^T}_{L\times M^T} \underbrace{V}_{L\times D}}_{O(L\times M\times D)}}_{O(L\times M\times D)}}_{O(L\times M\times D)}$$



$M \times L \times D$

$$D = \underbrace{\mathrm{Dig}\,(\underbrace{\phi(Q)\ \underbrace{\phi(K)^T 1_L}_{O(L\times M)}}_{O(L\times M)})}_{O(L\times M)}$$

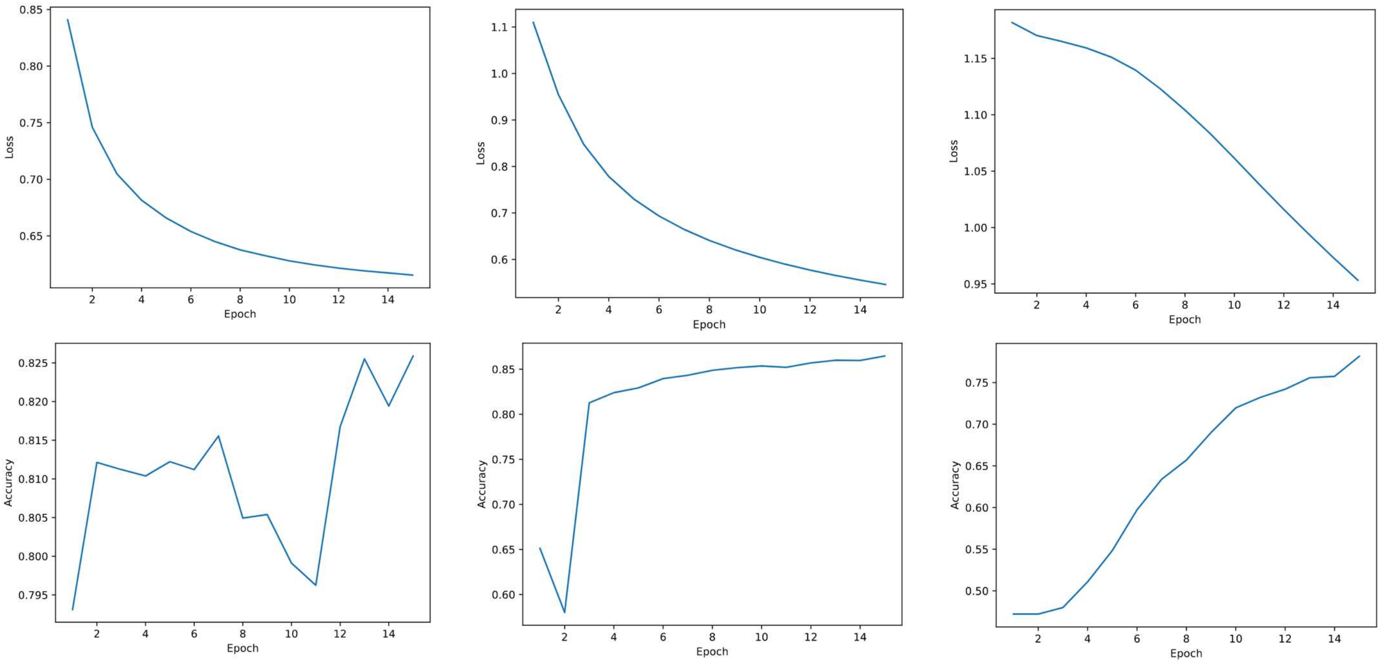
$Z \to (L, M, D)$

Linearly dependent of L, M and D

Question 2

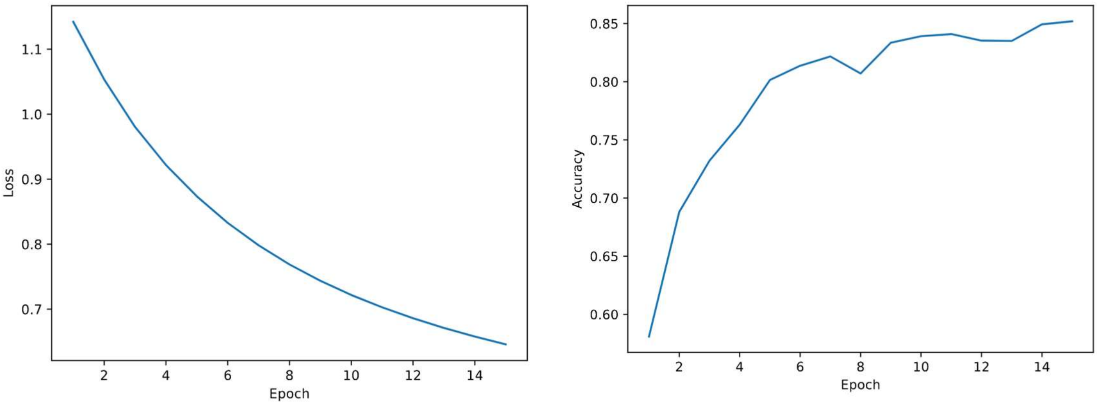
Q2.1

0.1 0.01 0.001



The best configuration is with a learning rate equal to 0.01 with a final loss of around 0.55 and 0.86 accuracy.

Q2.2



Q2.3

no_maxpool: False

Number of trainable parameters: 168572

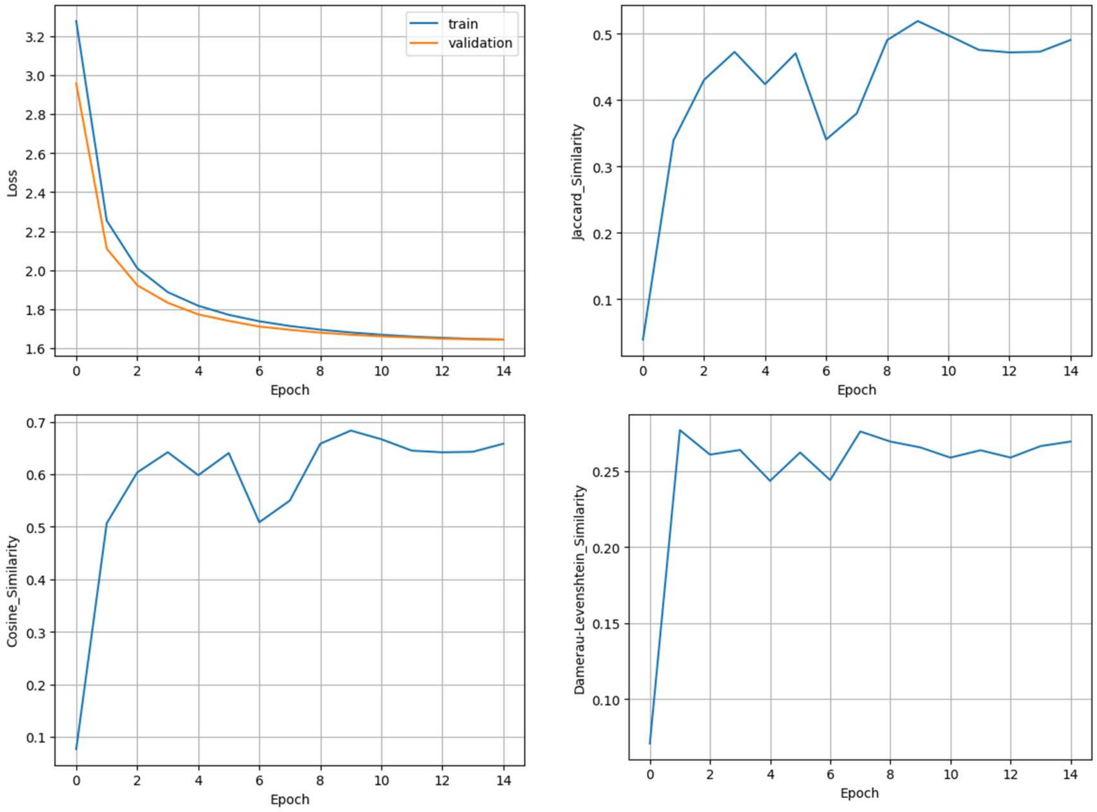
no_maxpool: True

Number of trainable parameters: 224892

Having more parameters can enhance the model's capacity, enabling it to capture intricate patterns in the data. Nevertheless, an elevated parameter count also escalates the likelihood of overfitting, particularly when the dataset lacks sufficient size.

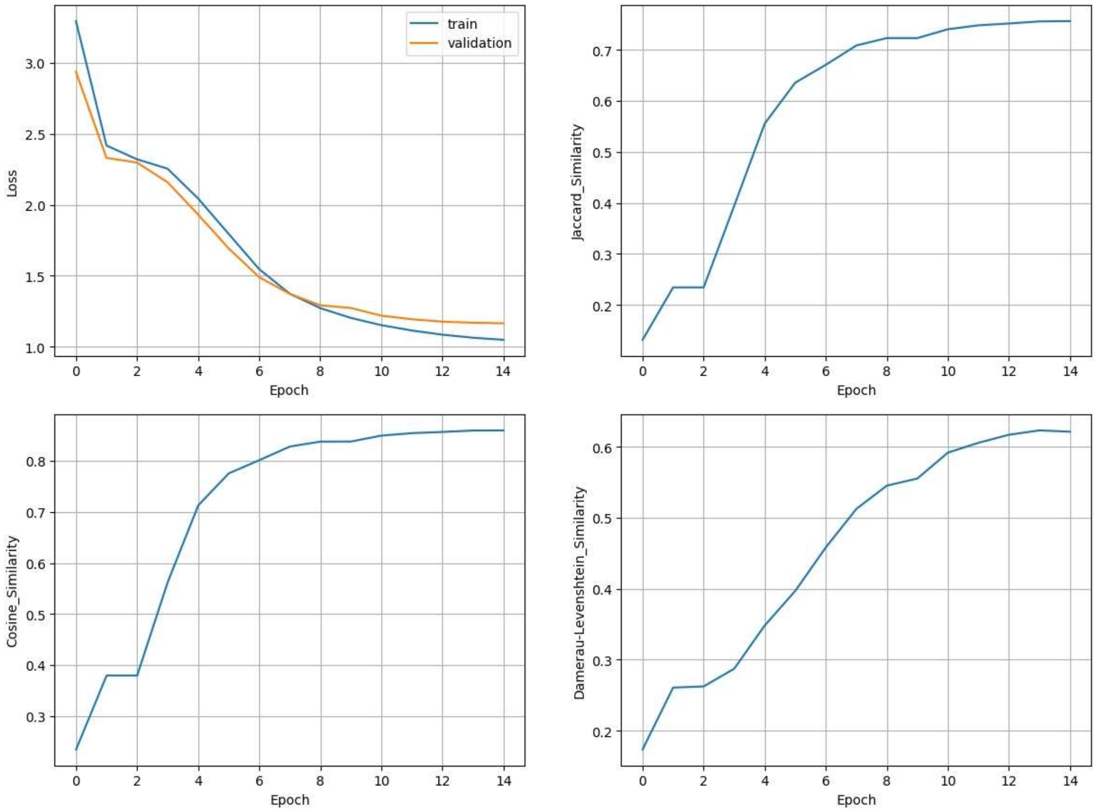
Question 3

Q3.1



{'jaccard_similarity': 0.503583607220501, 'cosine_similarity': 0.669388120203678, 'damerau-levenshtein_similarity': 0.2736332793584701, 'loss': 1.6690384796479854}

Q3.2



{'jaccard_similarity': 0.7636617752753647, 'cosine_similarity': 0.8647155264952522, 'damerau-levenshtein_similarity': 0.6314035820584282, 'loss': 1.160844937330339}

Q3.3

Unlike the sequential input processing of the LSTM, the attention mechanism allows the model to focus on different parts of the input sequence simultaneously. It assigns different weights to different positions in the input sequence, emphasizing the relevant parts for predicting the current token.

Q3.4

There is not a very big difference between the three measures in use. When maximizing one, the other will come to a near maximum state as well. In particular, Jaccard Similarity, when calculating the percentage of common elements in a sequence, can be considered alike Demerau-Levenstein which counts the number of edits for two sequences to reach equality.

Finally, the cosine turns the text sequences into vectors of words and its frequencies and calculates the cosine between two given vectors.

While Tomás Oliveira came up with the coding resolutions to the second and third questions, João Costa solved the hole question 1, ran the necessary code and created the report.