Example

Window Size of 2.

The quick brown fox jumps over the lazy dog. ⇒ (the, quick)
(the, brown)

The quick brown fox jumps over the lazy dog ⇒ (jumps, brown)
(jumps, fox)
(jumps, over)
(jumps, the)

[(1, 2),
(1, 3),
(4, 3),
(4, 5),
(4, 6),
(4, 1)]

Codification to Number

Necessary for the input layers

XARXA NEURONAL

layer 1

layer 2

seq1 = [1, 1, 4, ..., 4]

seq2 = [2, 3, 3, ..., 1]

embedding_layer

embedded_1

embedded_2

Pooling-1D → Sum of column values and dividing by number of columns

↓

each feature is represented by a value.

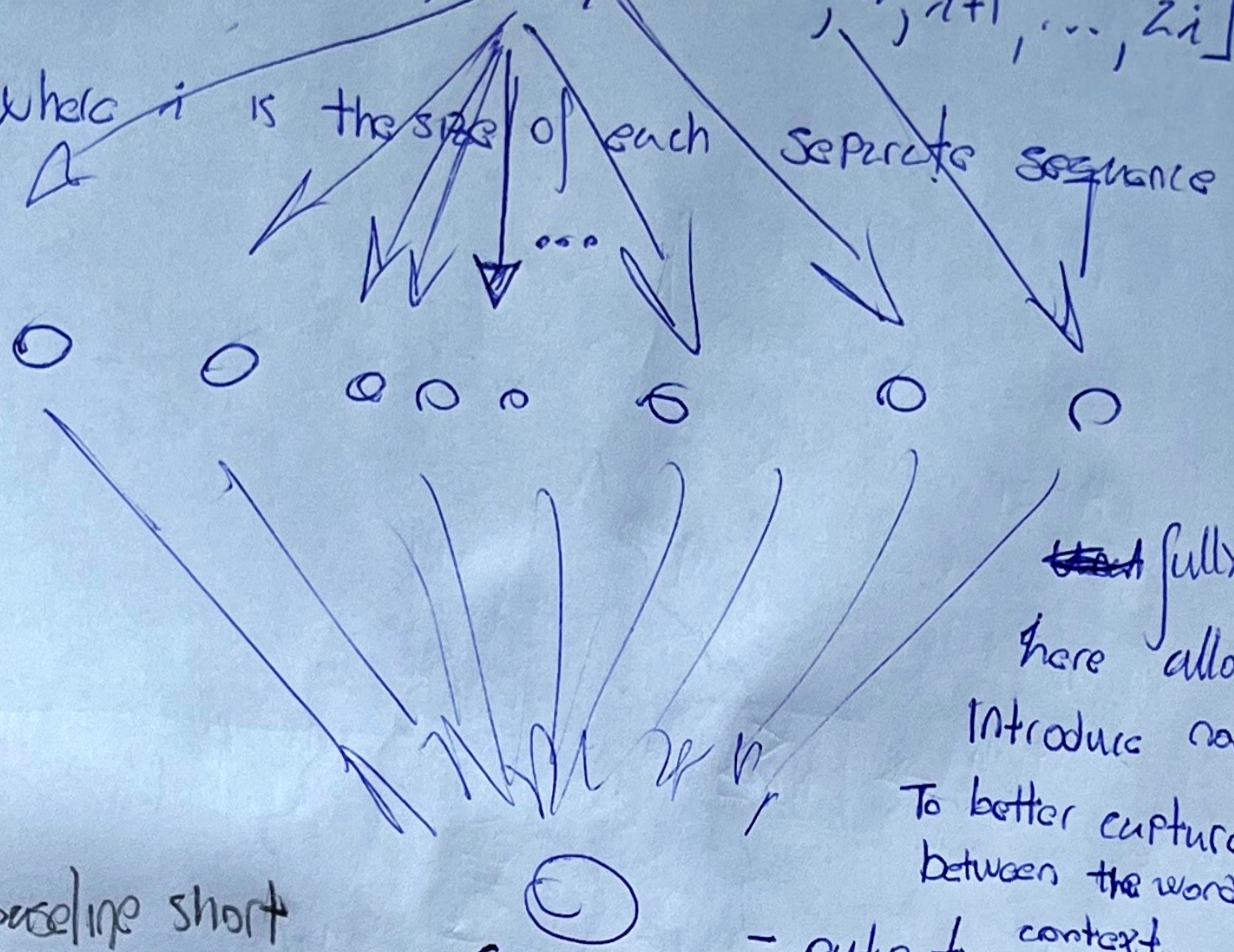
concatenate

- We combine them because we use two sources of data, the context and the word.

Pooled_1 Pooled_2

[0'1, 1'3, 0'7, ... i-1, i, i+1, ..., 2i]

where i is the size of each separate sequence



} Hidden layer amb ReLU

fully connected here allows us to introduce non linearity. To better capture relationships between the words and their context.

- output

baseline short

Single vector output with as many values as input data.

EACH VALUE of the output vector is the likelihood of each inputed pair of being a positive example for the skip-gram.

*

Explanation of an embedding layer:

~~Ex~~ One key component → Embedding Matrix

Embedding Matrix:

$$
l \updownarrow \begin{pmatrix} 0'1 & & & \\ 0'3 & \cdots & \cdots & 0'8 \\ 1'2 & & \cdots & \vdots \\ 1'7 & & & \vdots \\ 0'2 & \cdots & \cdots & 1'5 \end{pmatrix}
$$

$$\underleftrightarrow{\quad m \quad}$$

- Where l is the dictionary size.
- And m is the desired size of the embedded words.

  ↑ Based on resources.

$\boxed{l \geq m}$ ← If it is greater we are trying to represent ~~a non existent~~ Wo

Initiates with random weights, and during the learning process each weight is ADJUSTED so that each word ~~as~~ captures relationships between words, in a way that similar words have a similar representation.

$l = m$ → One-hot encoding, does not capture any relationship as words are considered independent.

~~A~~ A too ~~low m~~ high m can lead us to independency while a too low value can bring to information loss.

How is an embedded matrix used with an ~~tt~~ input vector?

Suppose $vec = (2,1,3,0)$ and $emb\_M = \begin{pmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \\ 0.7 & 0.8 & 0.9 \\ 1 & 1.1 & 1.2 \end{pmatrix}$

We do the following:

$$(2,1,3,0) \quad \begin{pmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \\ 0.7 & 0.8 & 0.9 \\ 1 & 1.1 & 1.2 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.7 & 0.8 & 0.9 \\ 0.4 & 0.5 & 0.6 \\ 1 & 1.1 & 1.2 \\ 0.1 & 0.2 & 0.3 \end{pmatrix}$$

We search the rows indicated by the index as they correspond to each word_id from the vocabulary.