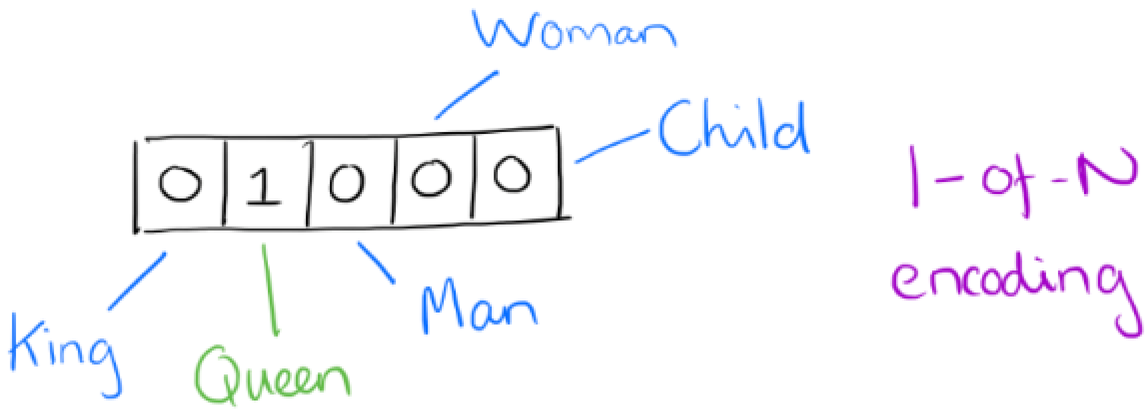
Word Vector Model

Word Vectors: One-Hot Encoding

- Suppose our vocabulary has only five words: King, Queen, Man, Woman and Child.
- We could encode the word ‘Queen’ as:



Limitations of One-hot encoding

- Word vectors are not comparable
- Using such an encoding, there is no meaningful comparison we can make between word vectors other than equality testing.

Word2Vec – A distributed representation

- Distributional representation – word embedding ?
  - Any word w_i in the corpus is given a distributional representation by an embedding
    - $w_i \in R^d$ i.e a d-dimensional vector that is learnt.
- For Example:

$$\textit{linguistics} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$

Distributional Representation

- Take a vector with several hundred dimensions (say 1000).
- Each word is represented by a distribution of weights across those elements.
- Instead of a one-to-one mapping between an element in the vector and a word
  - the representation of a word is spread across all the elements in the vector
  - each element in the vector contributes to the definition of many words.

Distributional Representation: Illustration

- If we label the dimensions in a hypothetical word vector (there are no such pre-assigned labels in the algorithm of course), it might look a bit like this:

Word Embeddings

- d typically in the range 50 to 1000
- Similar words should have similar embeddings

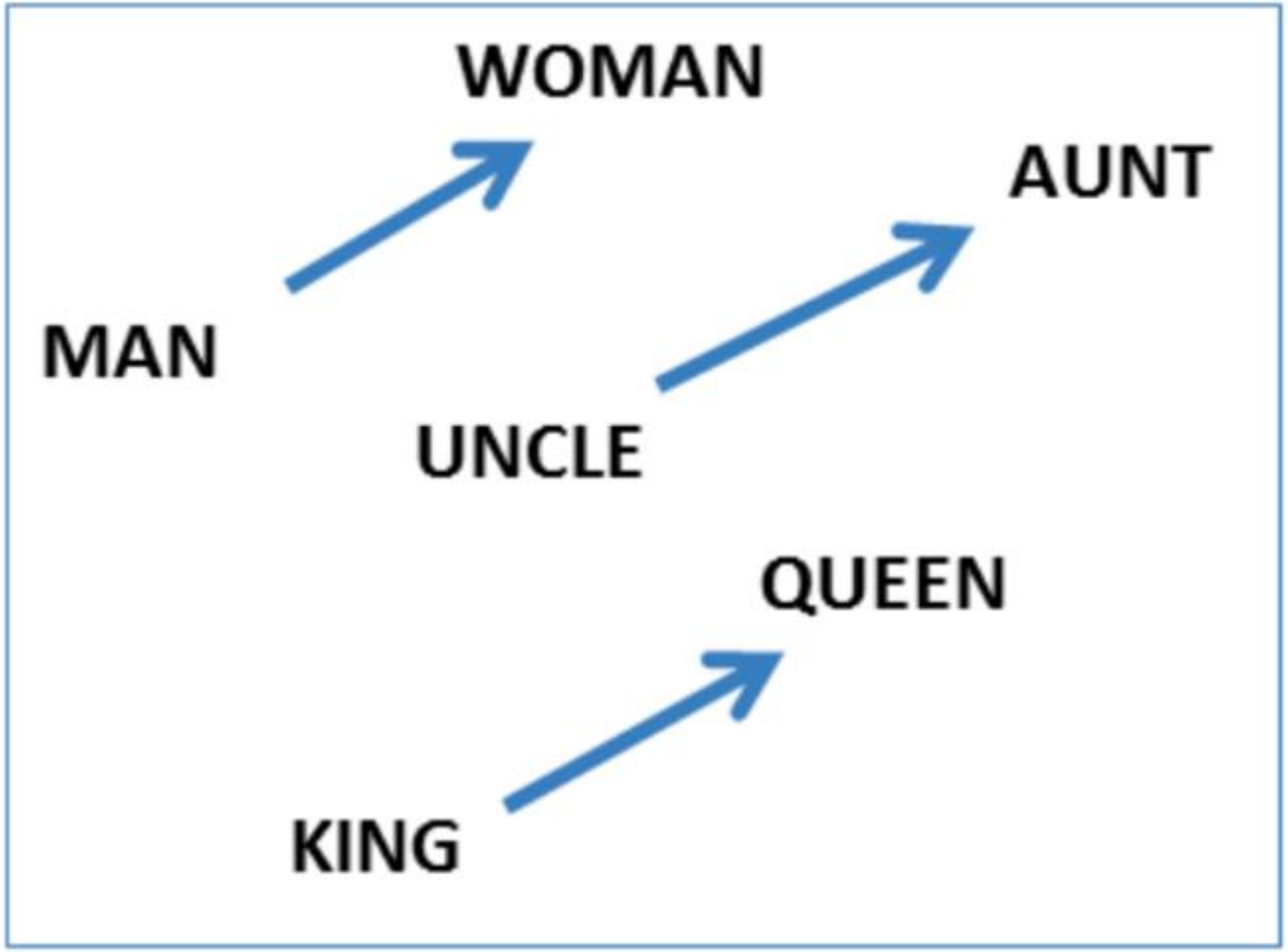
Reasoning with Word Vectors

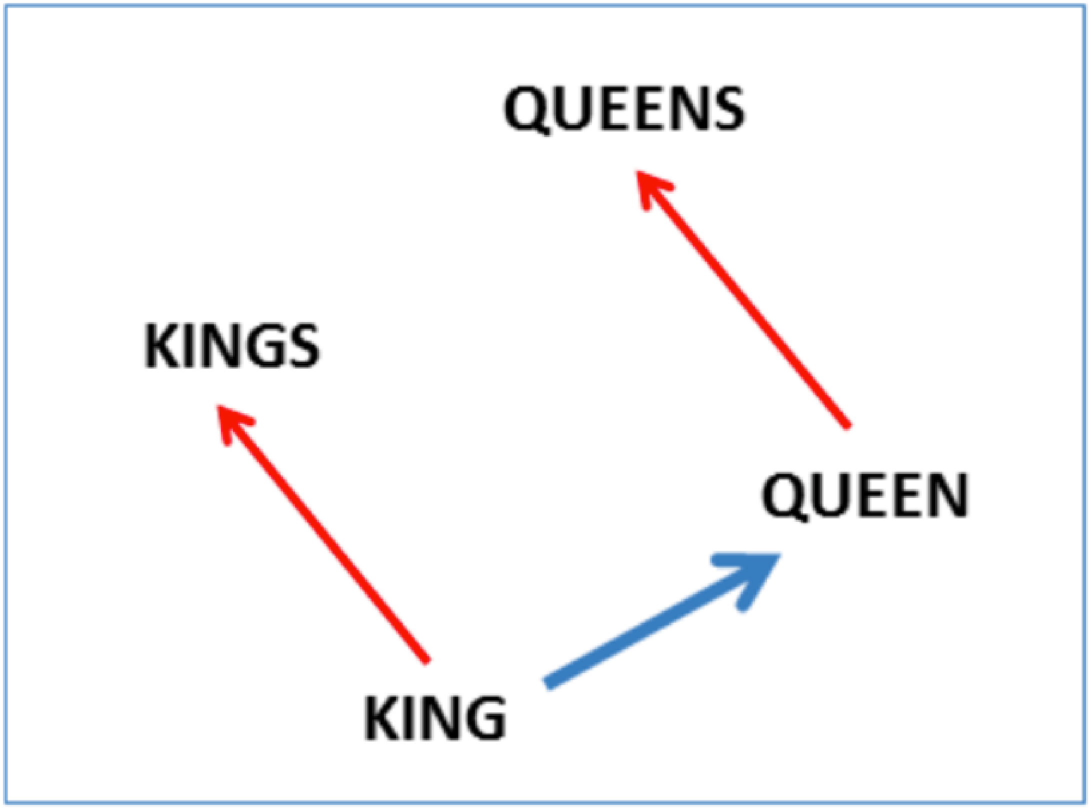
- It has been found that the learned word representations in fact capture meaningful syntactic and semantic regularities in a very simple way.
- Case of Singular-Plural Relations
  - If we denote the vector for word i as x_i, and focus on the singular/plural relation, we observe that
    - $x_{apple} - x_{apples} \approx x_{car} - x_{cars} \approx x_{family} - x_{families} \approx x_{car} - x_{cars}$ and so on.

Reasoning with Word Vectors

- Perhaps more surprisingly, we find that this is also the case for a variety of semantic relations.
- Good at answering analogy questions:
  - a is to b, as c is to ?
  - man is to woman as uncle is to ? (aunt)
- A simple vector offset method based on cosine distance shows the relation.

- .


WOMAN
AUNT
MAN
UNCLE
QUEEN
KING


QUEENS
KINGS
QUEEN
KING

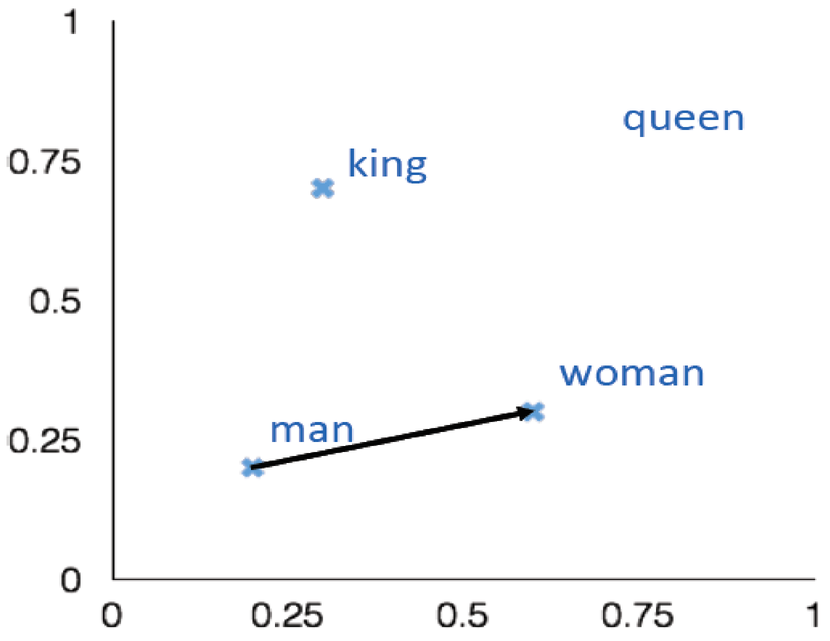
Analogy Test

a:b :: c:?

$$d = \arg\max_x \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c||}$$

man:woman :: king:?

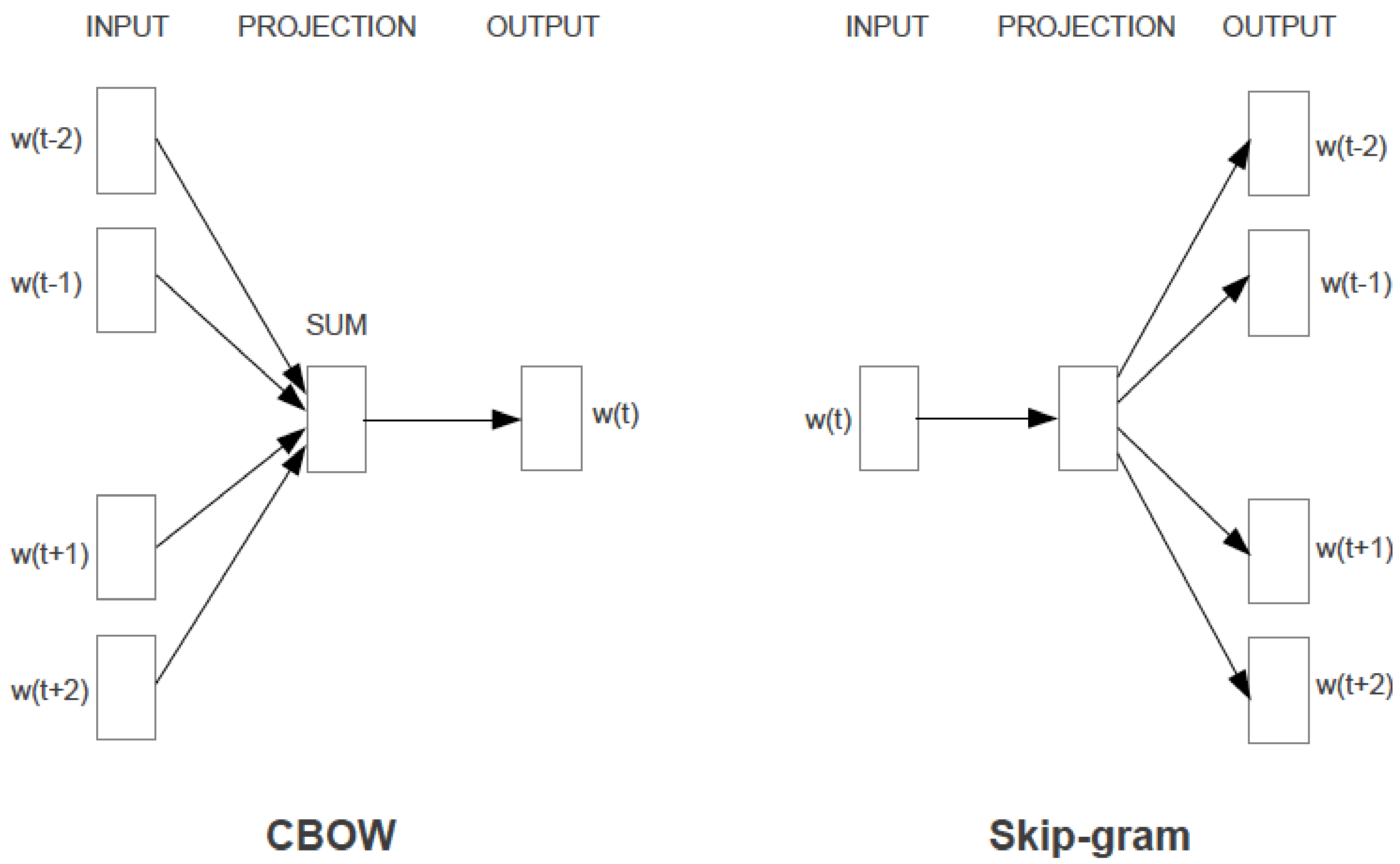
+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]



Learning Word Vectors

- Instead of capturing co-occurrence counts directly, predict (using) surrounding words of every word.
- Code as well as word-vectors: https://code.google.com/p/word2vec/

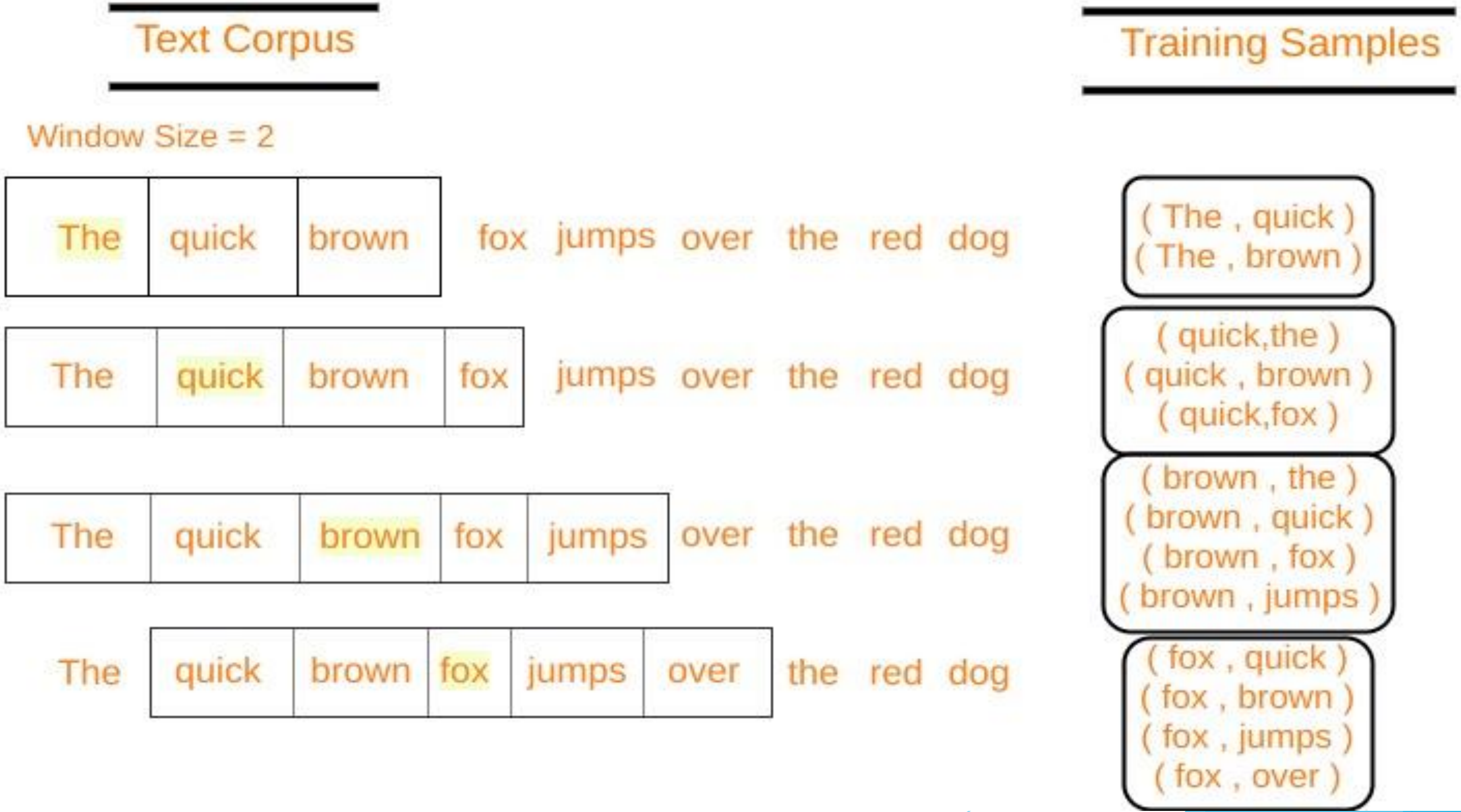
Two Variations: CBOW and Skip-grams



CBOW

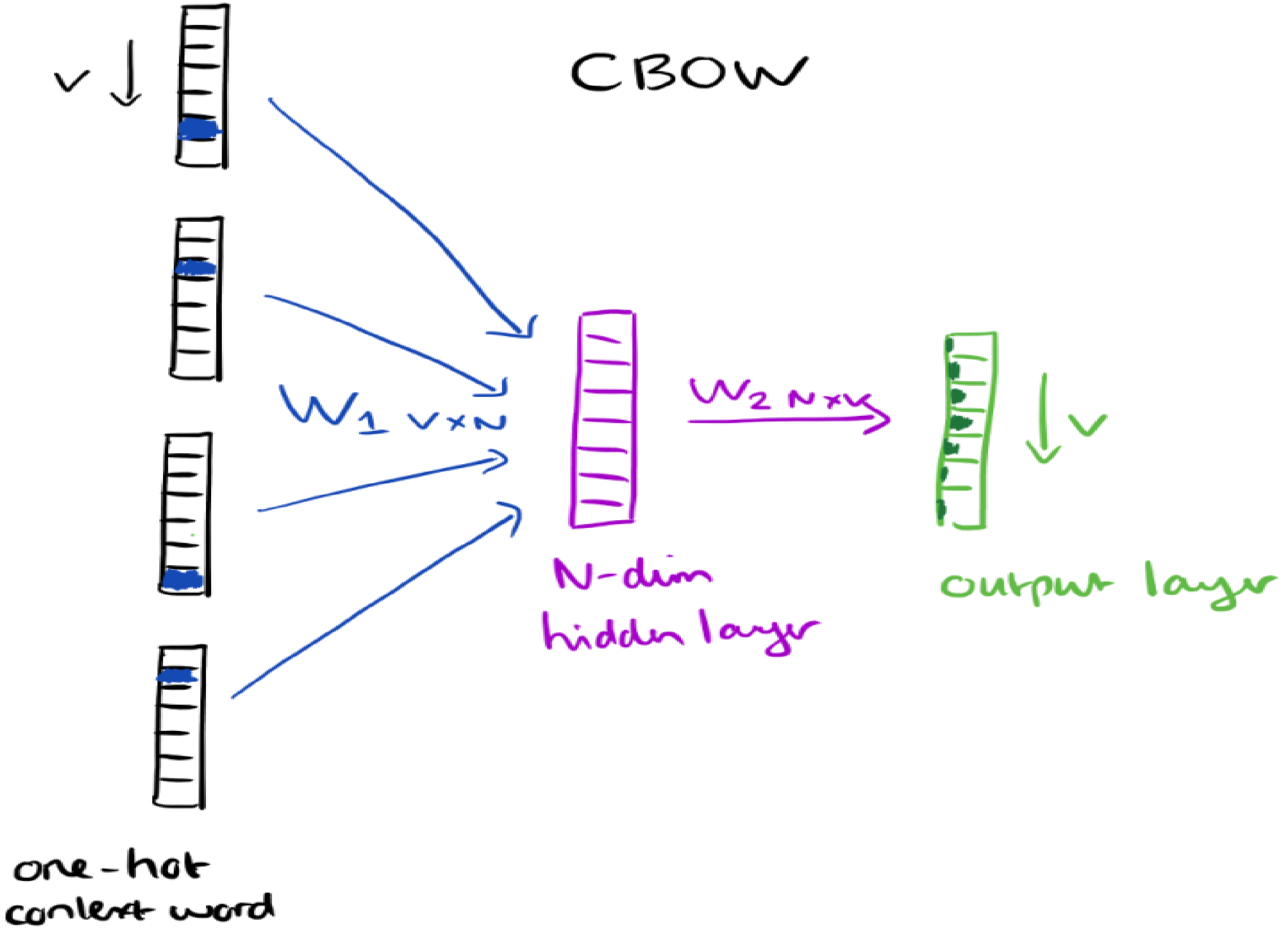
- Consider a piece of prose such as:
- “The recently introduced continuous Skip-gram model is an efficient method for learning high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships.”
- Imagine a sliding window over the text, that includes the central word currently in focus, together with the four words that precede it, and the four words that follow it:

An example

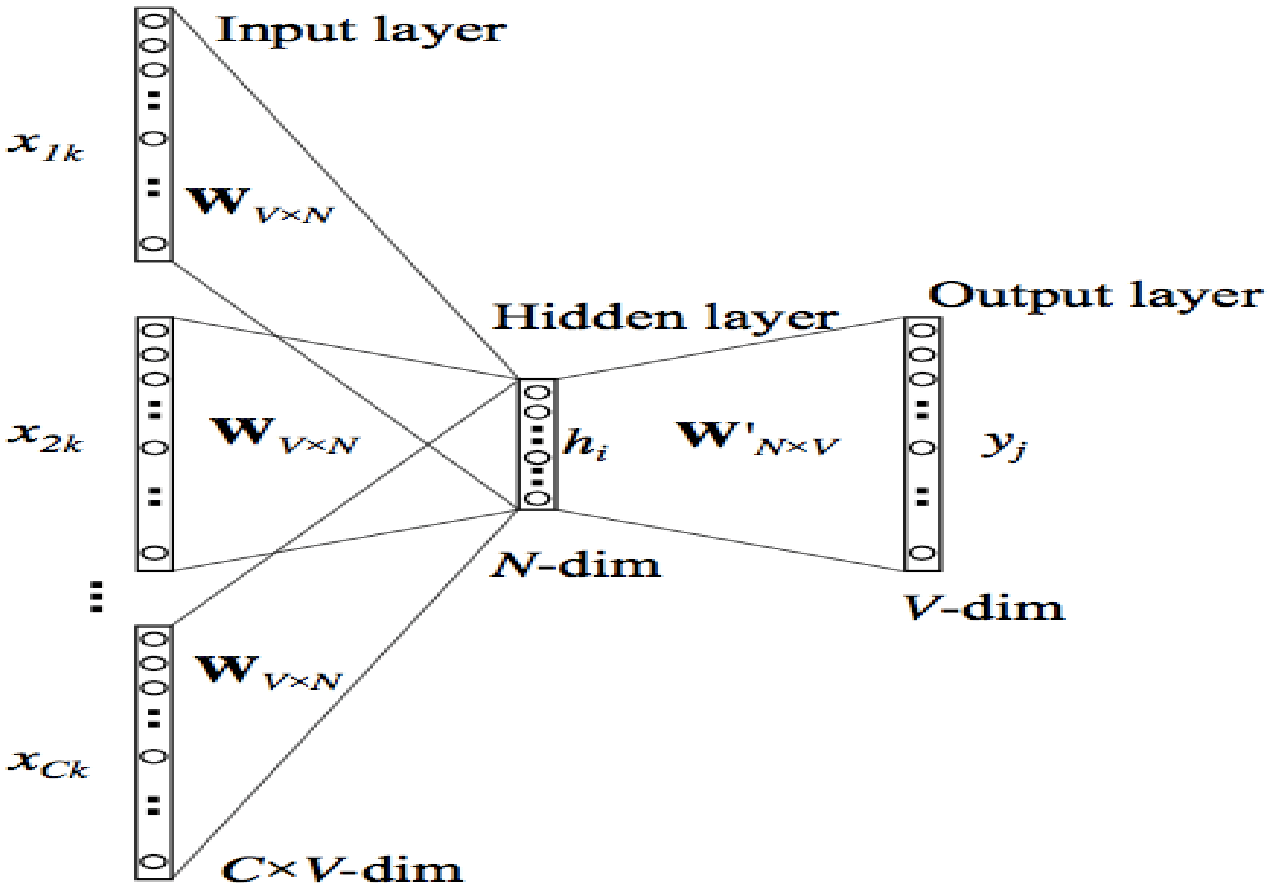

Text Corpus
Window Size = 2
The quick brown fox jumps over the red dog
The quick brown fox jumps over the red dog
The quick brown fox jumps over the red dog
The quick brown fox jumps over the red dog
Training Samples
(The , quick)
(The , brown)
(quick,the)
(quick , brown)
(quick,fox)
(brown , the)
(brown , quick)
(brown , fox)
(brown , jumps)
(fox , quick)
(fox , brown)
(fox , jumps)
(fox , over)

CBOW

- The context words form the input layer. Each word is encoded in one-hot form.
- A single hidden and output layer.



multiple context words



CBOW: Training Objective

- The training objective is to maximize the conditional probability of observing the actual output word (the focus word) given the input context words, with regard to the weights.
- In our example, given the input (“an”, “efficient”, “method”, “for”, “high”, “quality” , “distributed”, “vector”), we want to maximize the probability of getting “learning” as the output.

CBOW: Input to Hidden Layer

- Since our input vectors are one-hot, multiplying an input vector by the weight matrix W_1 amounts to simply selecting a row from W_1.

$$\begin{array}{cccc} \text{input} & W_1 & & \text{hidden} \\ 1\times V & V\times N & & 1\times N \\ \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} & \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix} & = & \begin{bmatrix} e & f & g & h \end{bmatrix} \\ & W_1 & & \end{array}$$

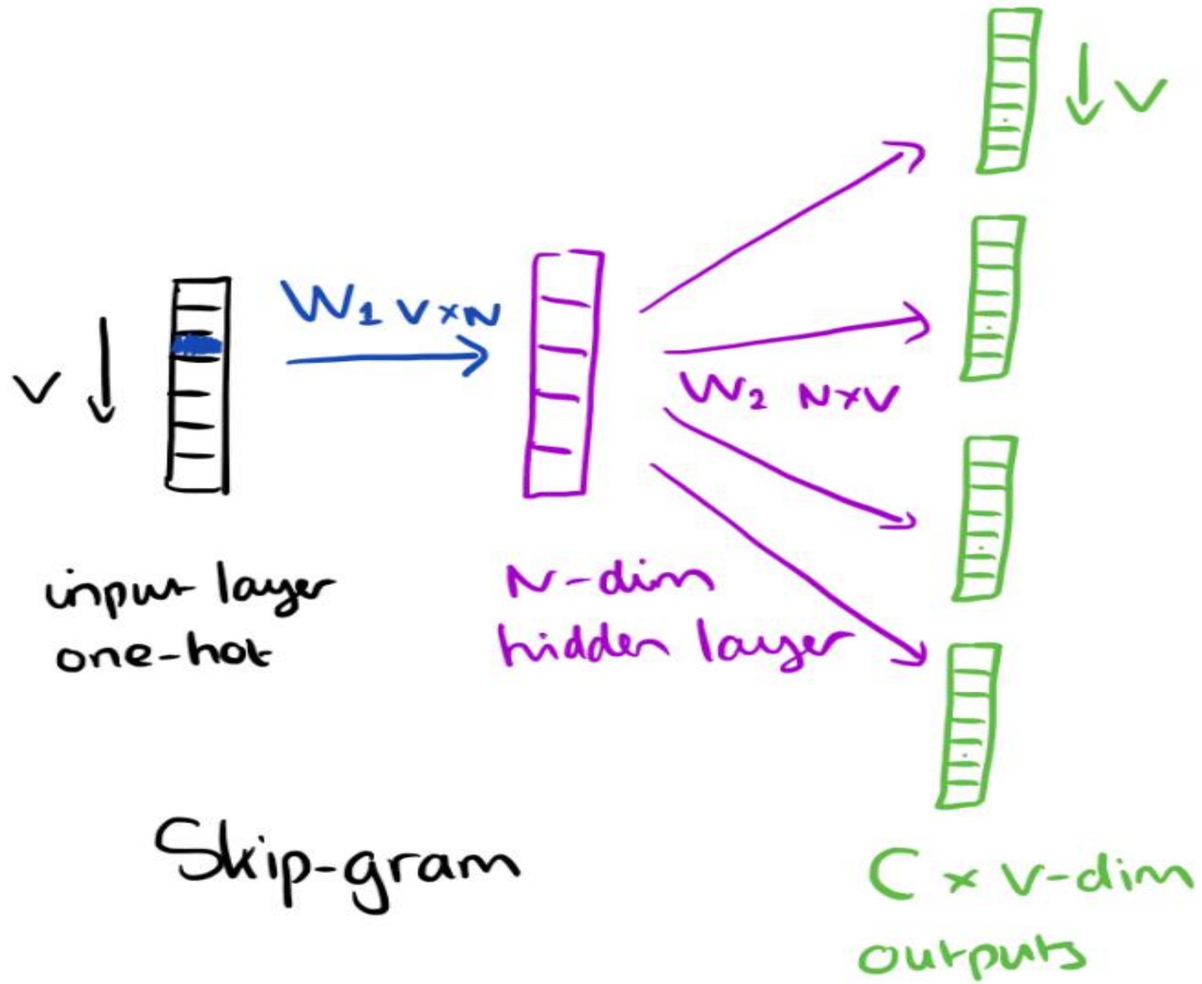
- Given C input word vectors, the activation function for the hidden layer h amounts to simply summing the corresponding ‘hot’ rows in W1 and dividing by C to take their average.

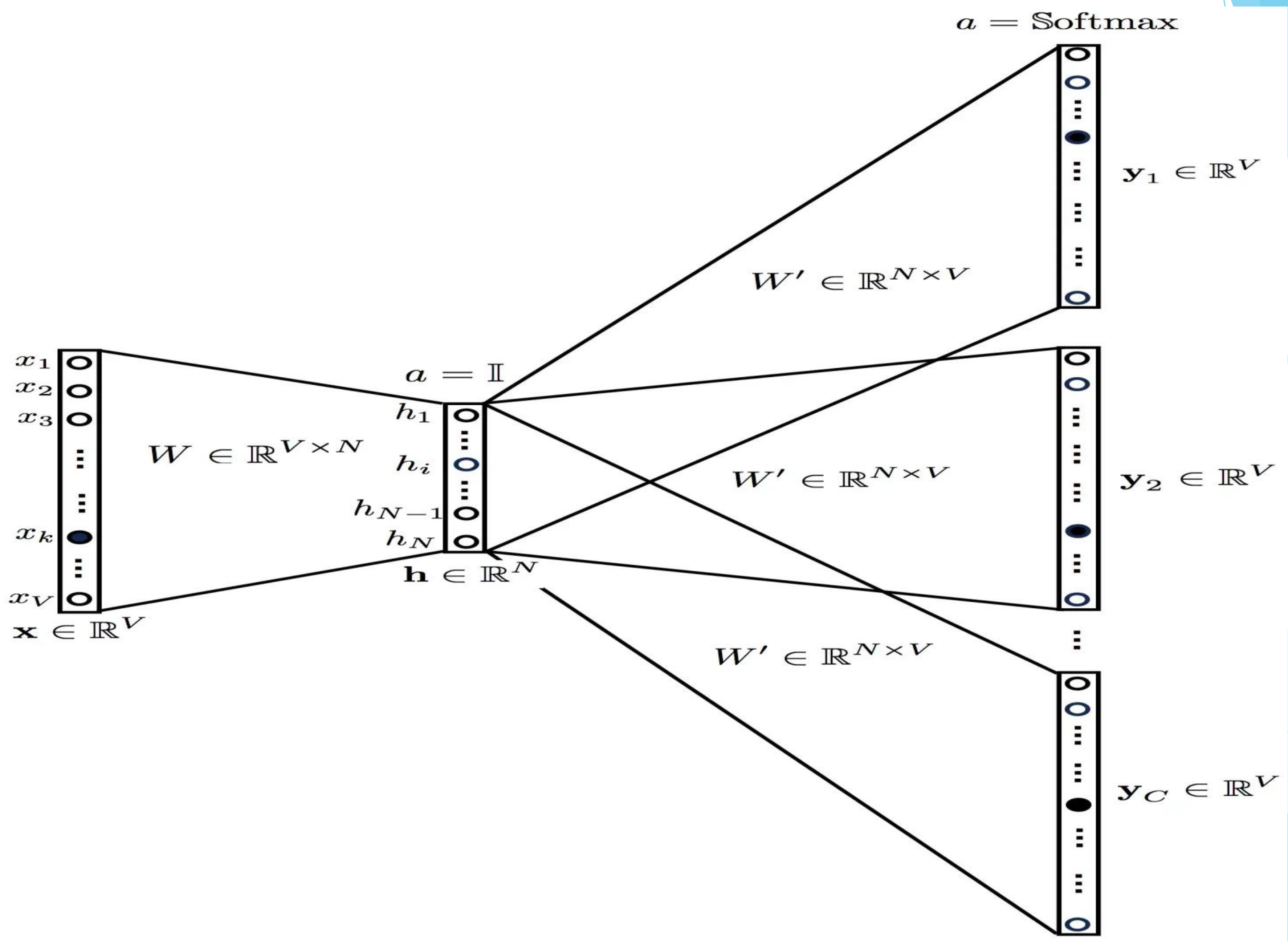
CBOW: Hidden to Output Layer

- From the hidden layer to the output layer, the second weight matrix W_2 can be used to compute a score for each word in the vocabulary, and softmax can be used to obtain the posterior distribution of words.

Skip-gram Model

- The skip-gram model is the opposite of the CBOW model. It is constructed with the focus word as the single input vector, and the target context words are now at the output layer:




$a = \text{Softmax}$
$\mathbf{y}_1 \in \mathbb{R}^V$
$W' \in \mathbb{R}^{N \times V}$
x_1
x_2
x_3
x_k
x_V
$\mathbf{x} \in \mathbb{R}^V$
$W \in \mathbb{R}^{V \times N}$
$a = \mathbb{I}$
h_1
h_i
h_{N-1}
h_N
$\mathbf{h} \in \mathbb{R}^N$
$W' \in \mathbb{R}^{N \times V}$
$\mathbf{y}_2 \in \mathbb{R}^V$
$W' \in \mathbb{R}^{N \times V}$
$\mathbf{y}_C \in \mathbb{R}^V$

Skipgram : Objective Fn

- The activation function for the hidden layer simply amounts to copying the corresponding row from the weights matrix W_1 (linear) as we saw before.
- At the output layer, we now output C multinomial distributions instead of just one.
- The training objective is to minimize the summed prediction error across all context words in the output layer.
- In our example, the input would be “learning”, and we hope to see (“an”, “efficient”, “method”, “for”, “high”,“quality”, “distributed”, “vector”) at the output layer.

Skip-gram Model

- Predict surrounding words in a window of length c of each word
- Objective Function: Maximize the log probability of any context word given
- the current center word:

$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} log\, p(w_{t+j}|w_t)$$

Thank You