Word embeddings

How it's made?

Problem statement

- How to make language computer readable?

Naive Ideas

Assign random IDs to words

- eat:1, apple:2, milk:3, drink:4
- apple + apple = drink ???

Treat words as categorical, independent variables

- eat:[1,0,0,0] apple:[0,1,0,0] milk:[0,0,1,0] drink:[0,0,0,1]

Problem statement

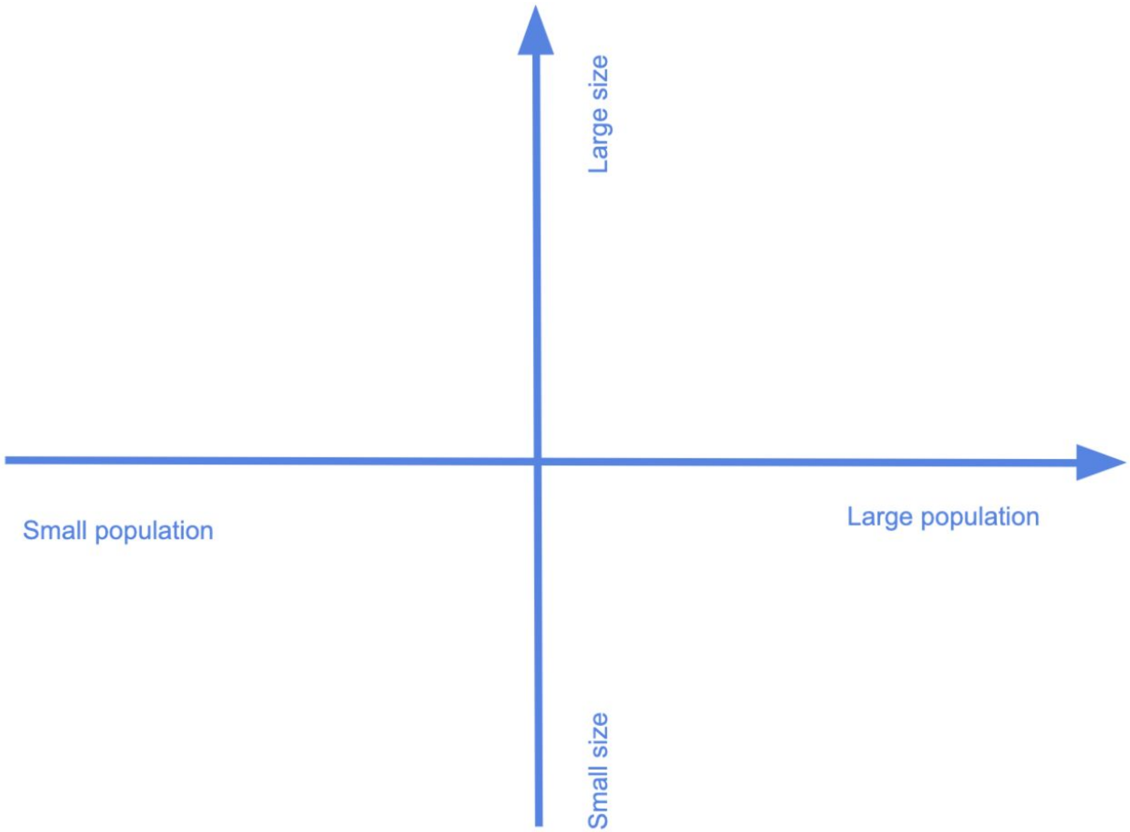
- How to make language computer readable?
- How to encode meaning and similarity

Desired solution

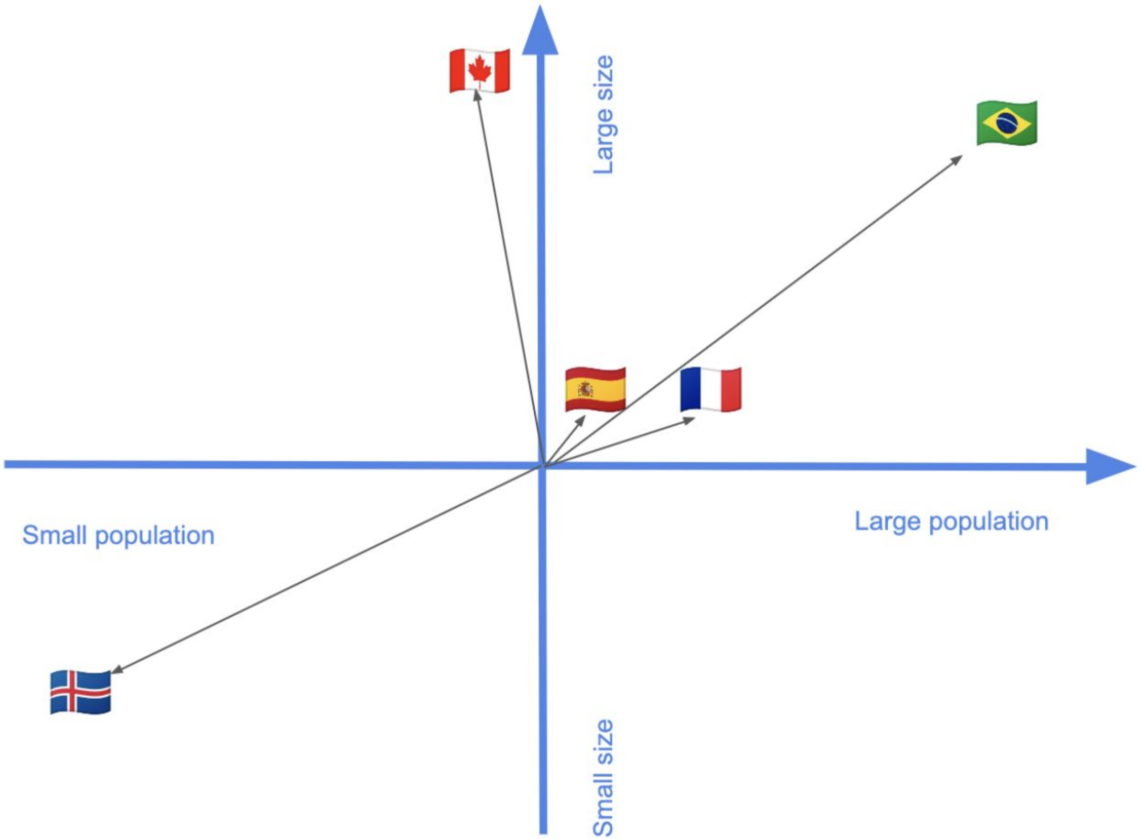
- A vector space with axis of “meaning” (semantic features)

- Similar words are closer in the vector space than non-similar words

Country	Population (mil pp)	Size (mil km^2)
🇫🇷 France	68	0.5
🇪🇸 Spain	47	0.5
🇧🇷 Brazil	214	8.5
🇮🇸 Iceland	0.4	0.1
🇨🇦 Canada	38	9



Country	Population (mil pp)	Size (mil km²)
🇫🇷 France	68	0.5
🇪🇸 Spain	47	0.5
🇧🇷 Brazil	214	8.5
🇮🇸 Iceland	0.4	0.1
🇨🇦 Canada	38	9



Normalize by size of vectors, measure similarity by angle

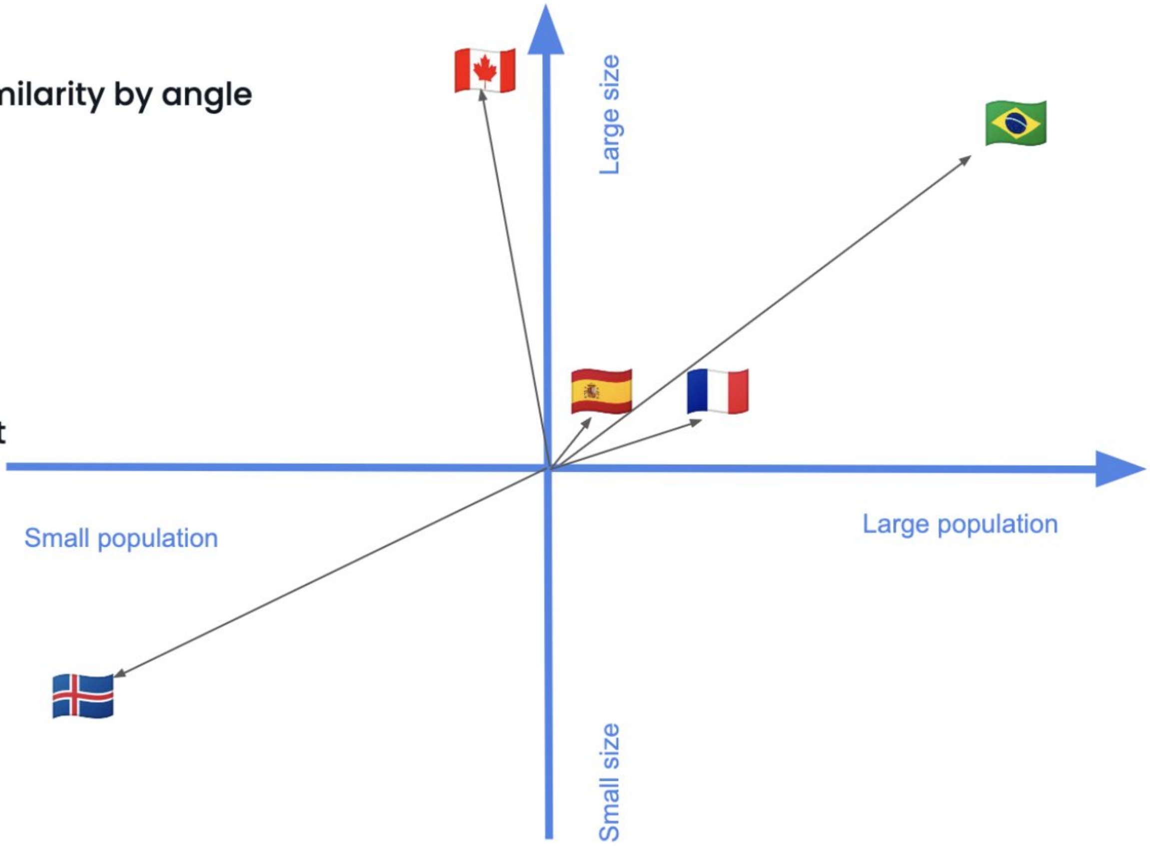
🇧🇷 [0.9, 0.8] – ||🇧🇷|| = 1.2
🇫🇷 [0.3, 0.2] – ||🇫🇷|| = 0.36
🇪🇸 [0.2, 0.2] – ||🇪🇸|| = 0.28
🇨🇦 [-.1, 0.9] – ||🇨🇦|| = 0.9
🇮🇸 [-.8, -.8] – ||🇮🇸|| = 1.13

Cosine similarity - normalized dot product

cosine(a,b) = a · b / (||a|| × ||b||)

cosein(🇫🇷,🇪🇸) = 0.1 / 0.1008 =0.99
cosine(🇧🇷,🇨🇦) = 0.63 / 1.08 = 0.58
cosine(🇧🇷,🇮🇸) = -1.36 / 1.36 = -0.99

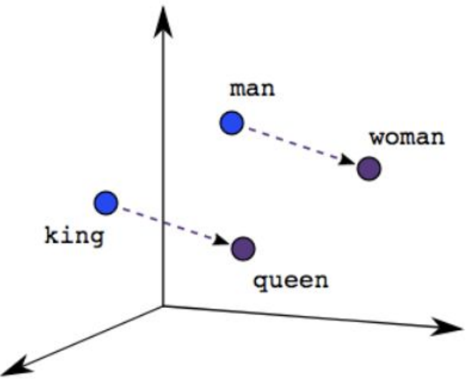
Values go from [-1, 1]



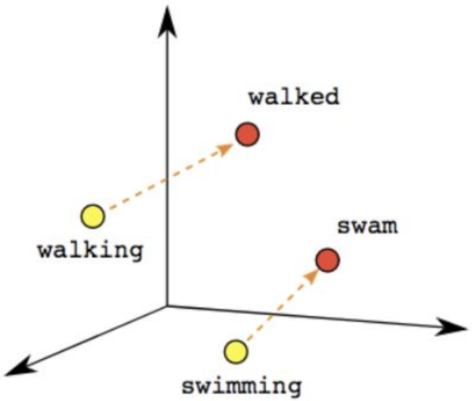
This can be done!

- Word embeddings can be automatically calculated
- Using self-supervised machine learning
- Based on word co-occurrences

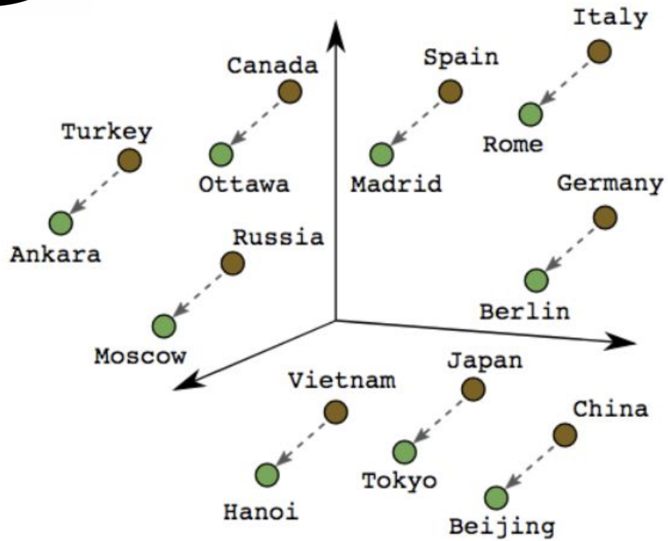
Word2Vec (2013)



Male-Female



Verb Tense



Country-Capital

Distributional semantics

Words occurring in a similar context
tend to have similar meaning

Context ~ Meaning

Distributional semantics

To start my day, I drink hot **coffee** in the mornings

Sipping some warm **tea** helps me wake up in the morning

Context:

coffee drink, hot, mornings

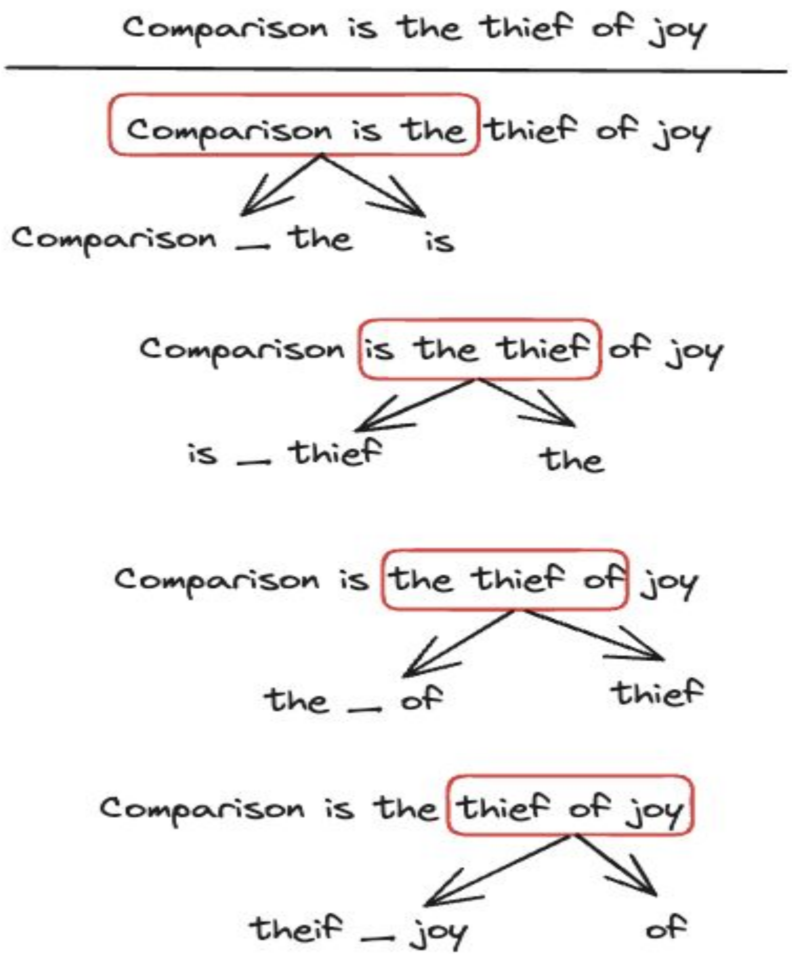
tea sipping, warm, wake up, morning

ML Objective

Given context words,
predict target word

DataSet

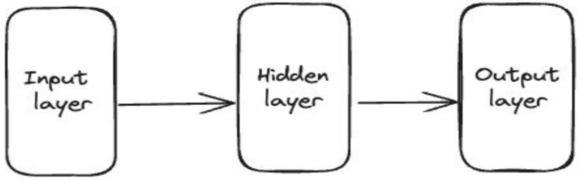
Text corpus (coherent text)
Self supervised
Masked target word, use sliding window

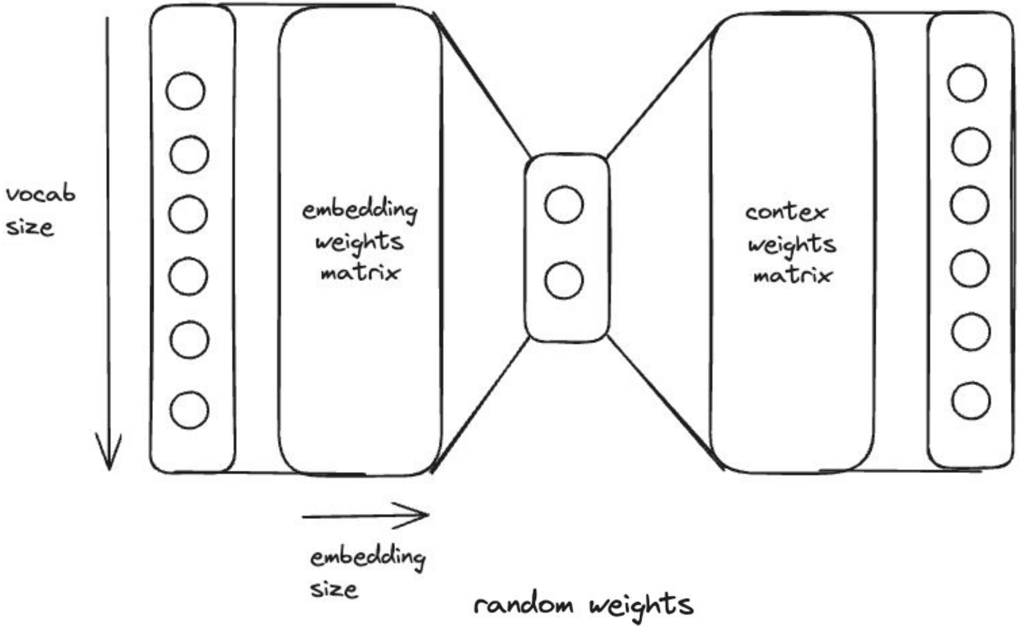


Set up

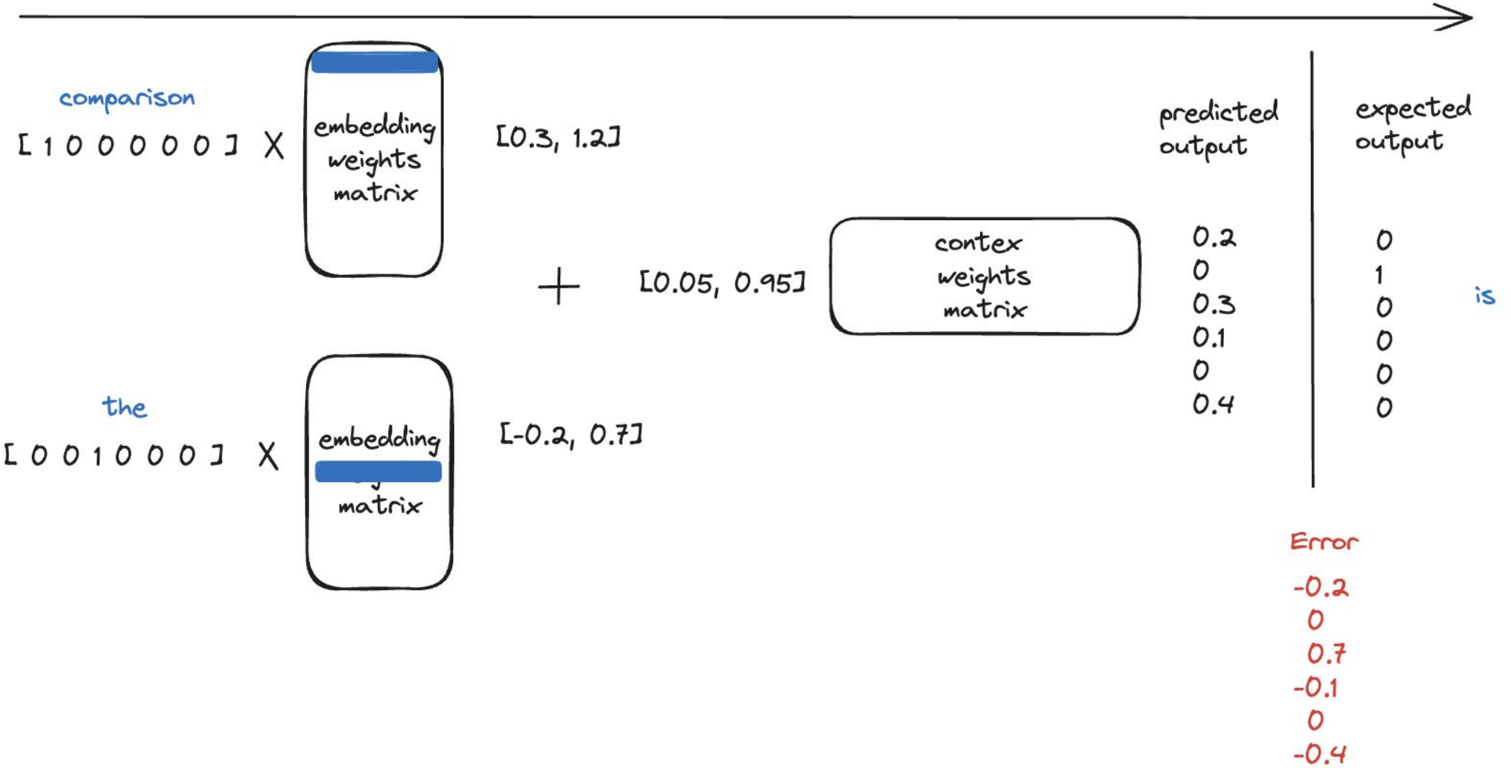
Vocabulary	1-hot encoding
comparison	[1 ,0 ,0 ,0 ,0 ,0]
is	[0, 1, 0, 0, 0, 0]
the	[0, 0, 1, 0, 0, 0]
thief	[0, 0, 0, 1, 0, 0]
of	[0, 0, 0, 0, 1, 0]
joy	[0, 0, 0, 0, 0, 1]

Model Architecture

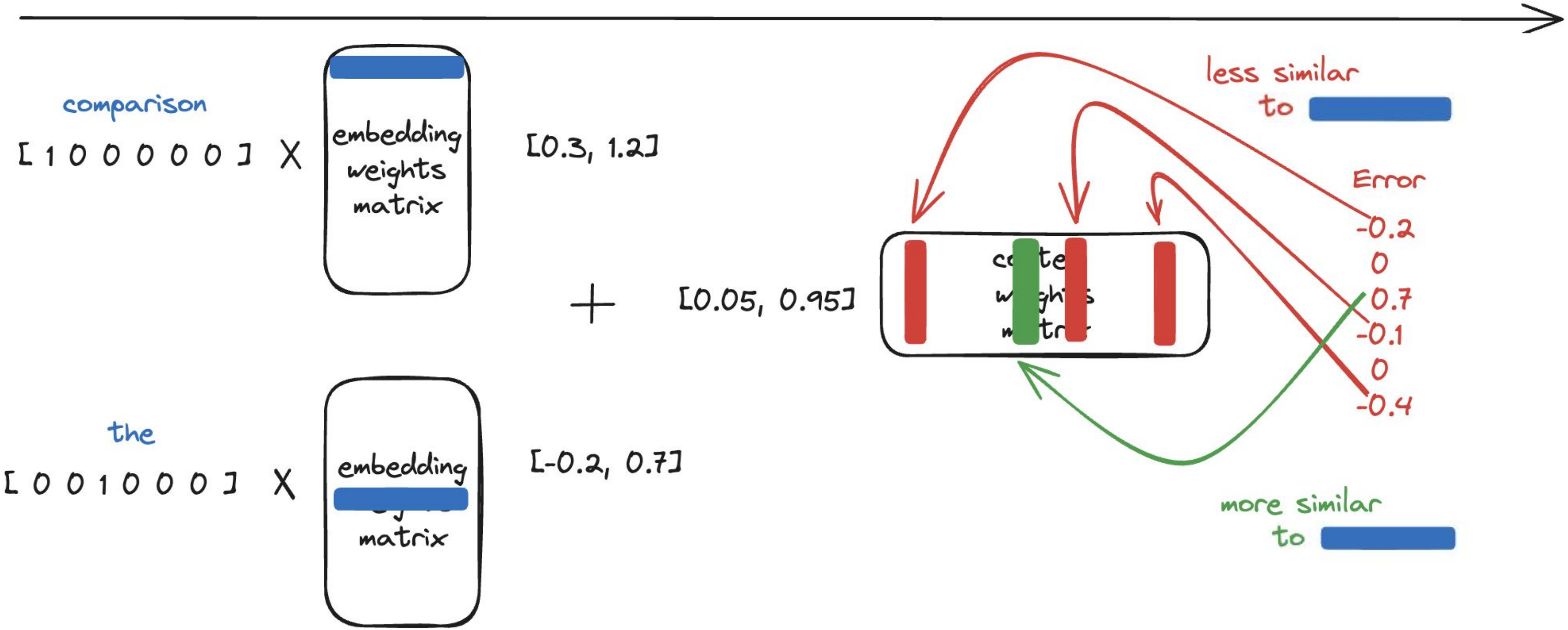




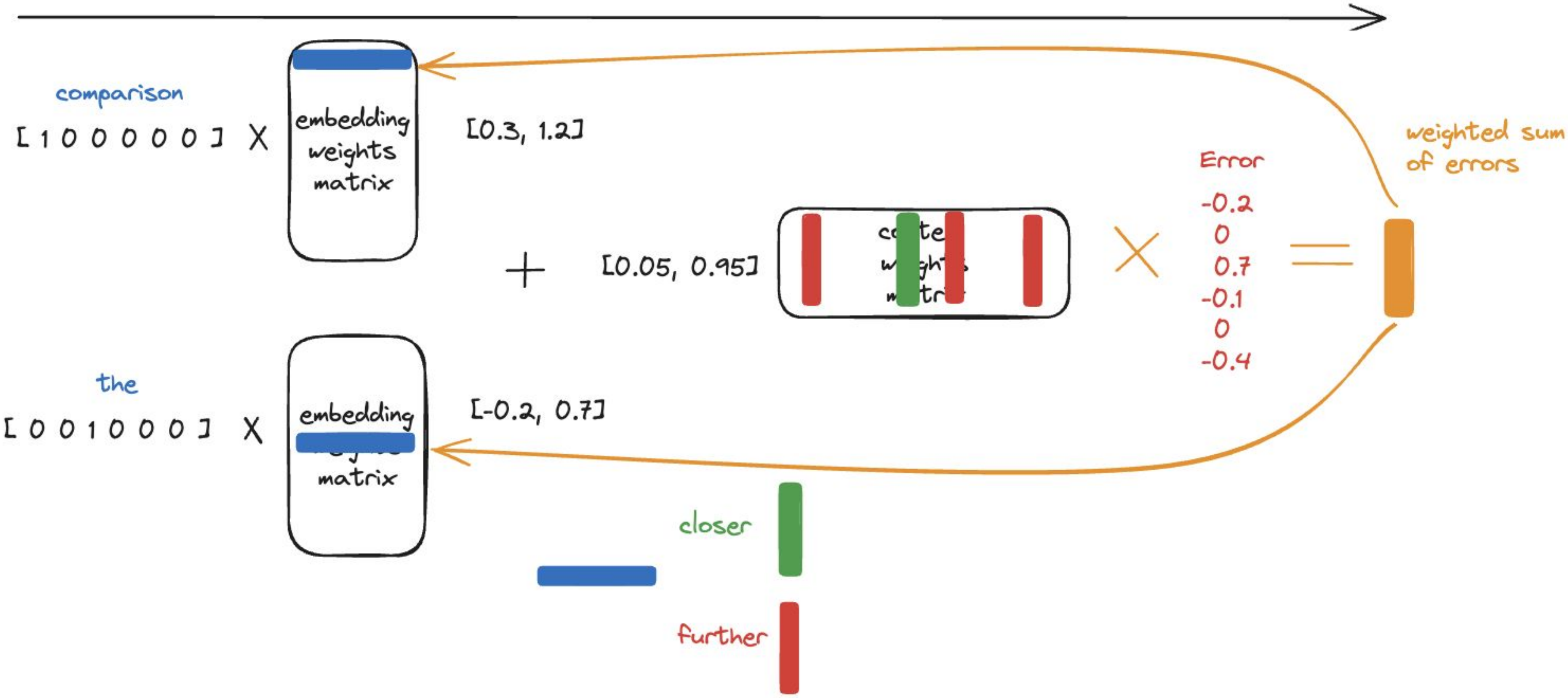
Prediction - Forward Pass



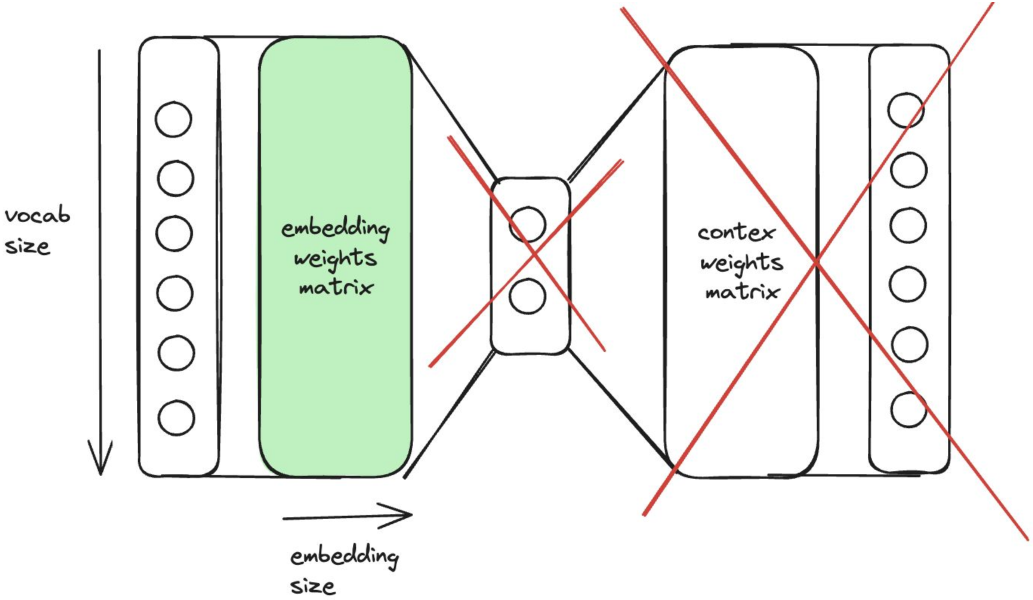
Error Back Propagation 1



Error Back Propagation 2



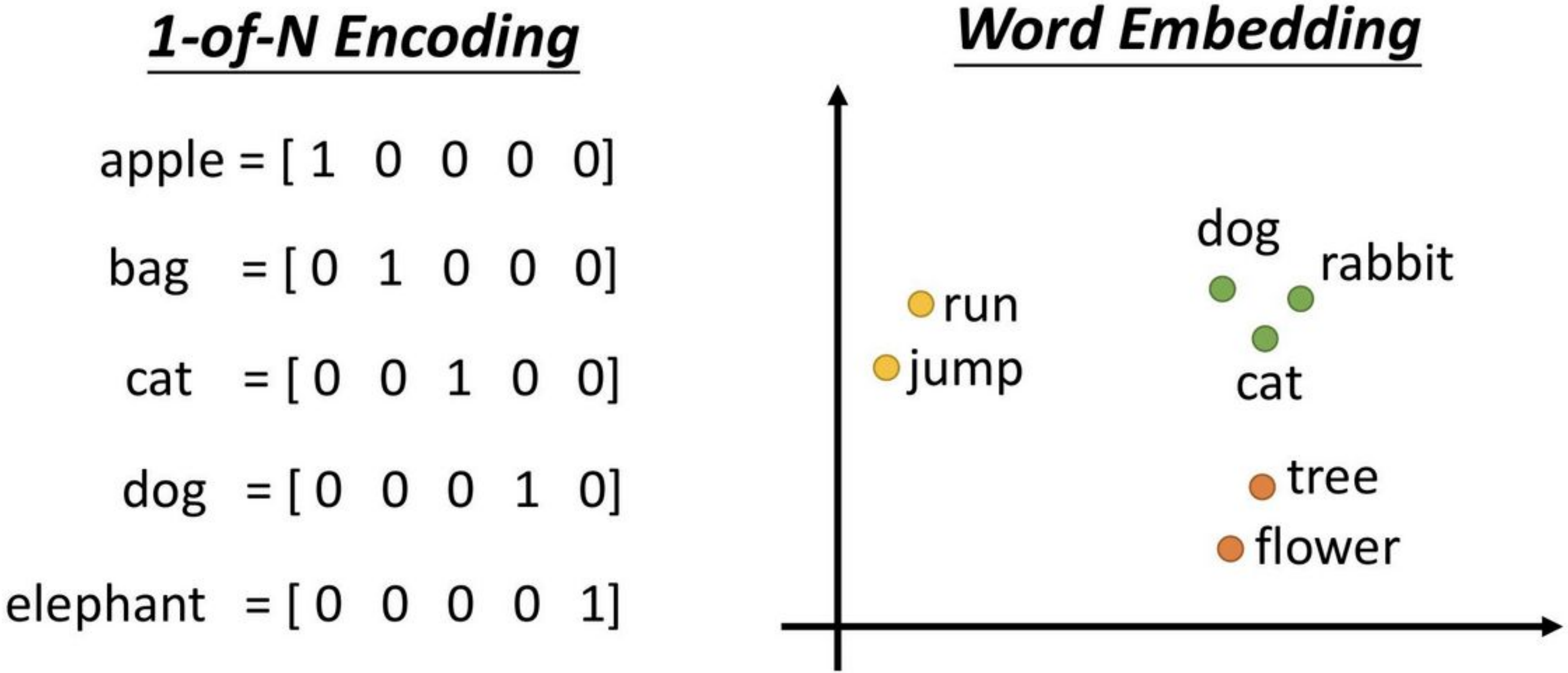
New goal



- target word prediction will not be great

- in the process we generate high-quality word embeddings, which can be used to train language models more efficiently

Result


1-of-N Encoding
apple = [1 0 0 0 0]
bag = [0 1 0 0 0]
cat = [0 0 1 0 0]
dog = [0 0 0 1 0]
elephant = [0 0 0 0 1]
Word Embedding
dog
rabbit
run
jump
cat
tree
flower

Surprising result

- word meaning can be represented well
  using vectors calculated, based on co-occurrence

- despite starting from a blank state (random vectors)
- despite not being taught a single rule of English syntax
- despite not being associated with a knowledge graph

Summary - How it's made?

- First it seems like magic
- Matrix multiplications
  and error backpropagation
- Still feels like magic