Word vectors

Our BOW approach until now

Representing a document by word frequency counts

Result of preprocessing and vectorizing:

0. `He took the dog for a walk to the dog playground`

$\Rightarrow$ `took dog walk dog playground`

$\Rightarrow$ `'took':1, 'dog': 2, walk: 1, playground: 1`

Consider these other sentences

1. He took the doberman for a walk to the dog playground
2. He took the cat for a walk to the dog playground
3. He killed the dog on his walk to the dog playground

The vectorized representations of these sentences have a "distance" (dissimilarity) of 1 each, but arguably, sentences 0 and 1 should be "closer" than others

Our BOW approach until now

Representing a document by word frequency counts

Result of preprocessing and vectorizing:

0. `He took the dog for a walk to the dog playground`

$\Rightarrow$ `took dog walk dog playground`

$\Rightarrow$ `'took':1, 'dog': 2, walk: 1, playground: 1`

Consider these other sentences

1. He took the doberman for a walk to the dog playground
2. He took the cat for a walk to the dog playground
3. He killed the dog on his walk to the dog playground

The vectorized representations of these sentences have a "distance" (dissimilarity) of 1 each, but arguably, sentences 0 and 1 should be "closer" than others

Our BOW approach until now

Representing a document by word frequency counts

Result of preprocessing and vectorizing:

0. `He took the dog for a walk to the dog playground`

$\Rightarrow$ `took dog walk dog playground`

$\Rightarrow$ `'took':1, 'dog': 2, walk: 1, playground: 1`

Consider these other sentences

1. He took the doberman for a walk to the dog playground
2. He took the cat for a walk to the dog playground
3. He killed the dog on his walk to the dog playground

The vectorized representations of these sentences have a "distance" (dissimilarity) of 1 each, but arguably, sentences 0 and 1 should be "closer" than others

Our BOW approach until now

Representing a document by word frequency counts

Result of preprocessing and vectorizing:

0. `He took the dog for a walk to the dog playground`

$\Rightarrow$ `took dog walk dog playground`

$\Rightarrow$ `'took':1, 'dog': 2, walk: 1, playground: 1`

Consider these other sentences

1. He took the doberman for a walk to the dog playground
2. He took the cat for a walk to the dog playground
3. He killed the dog on his walk to the dog playground

The vectorized representations of these sentences have a "distance" (dissimilarity) of 1 each, but arguably, sentences 0 and 1 should be "closer" than others

Our BOW approach until now

Representing a document by word frequency counts

Result of preprocessing and vectorizing:

0. `He took the dog for a walk to the dog playground`

$\Rightarrow$ `took dog walk dog playground`

$\Rightarrow$ `'took':1, 'dog': 2, walk: 1, playground: 1`

Consider these other sentences

1. He took the doberman for a walk to the dog playground
2. He took the cat for a walk to the dog playground
3. He killed the dog on his walk to the dog playground

The vectorized representations of these sentences have a "distance" (dissimilarity) of 1 each, but arguably, sentences 0 and 1 should be "closer" than others

Our BOW approach until now

- Our vectorizers gave a random ID to each word
- What if we instead would represent each word by another vector representing its meaning?
- For, instance, 'doberman' and 'dog' should be represented by vectors that are close to each other in space, while 'kill' and 'walk' should be far from each other.

$\Rightarrow$ That's the idea behind word embeddings!

Or, more broadly: Can computers understand meanings, semantic relationships, different types of contexts?

Our BOW approach until now

- Our vectorizers gave a random ID to each word
- What if we instead would represent each word by another vector representing its meaning?
- For, instance, 'doberman' and 'dog' should be represented by vectors that are close to each other in space, while 'kill' and 'walk' should be far from each other.

$\Rightarrow$ That's the idea behind word embeddings!

Or, more broadly: Can computers understand meanings, semantic relationships, different types of contexts?

Our BOW approach until now

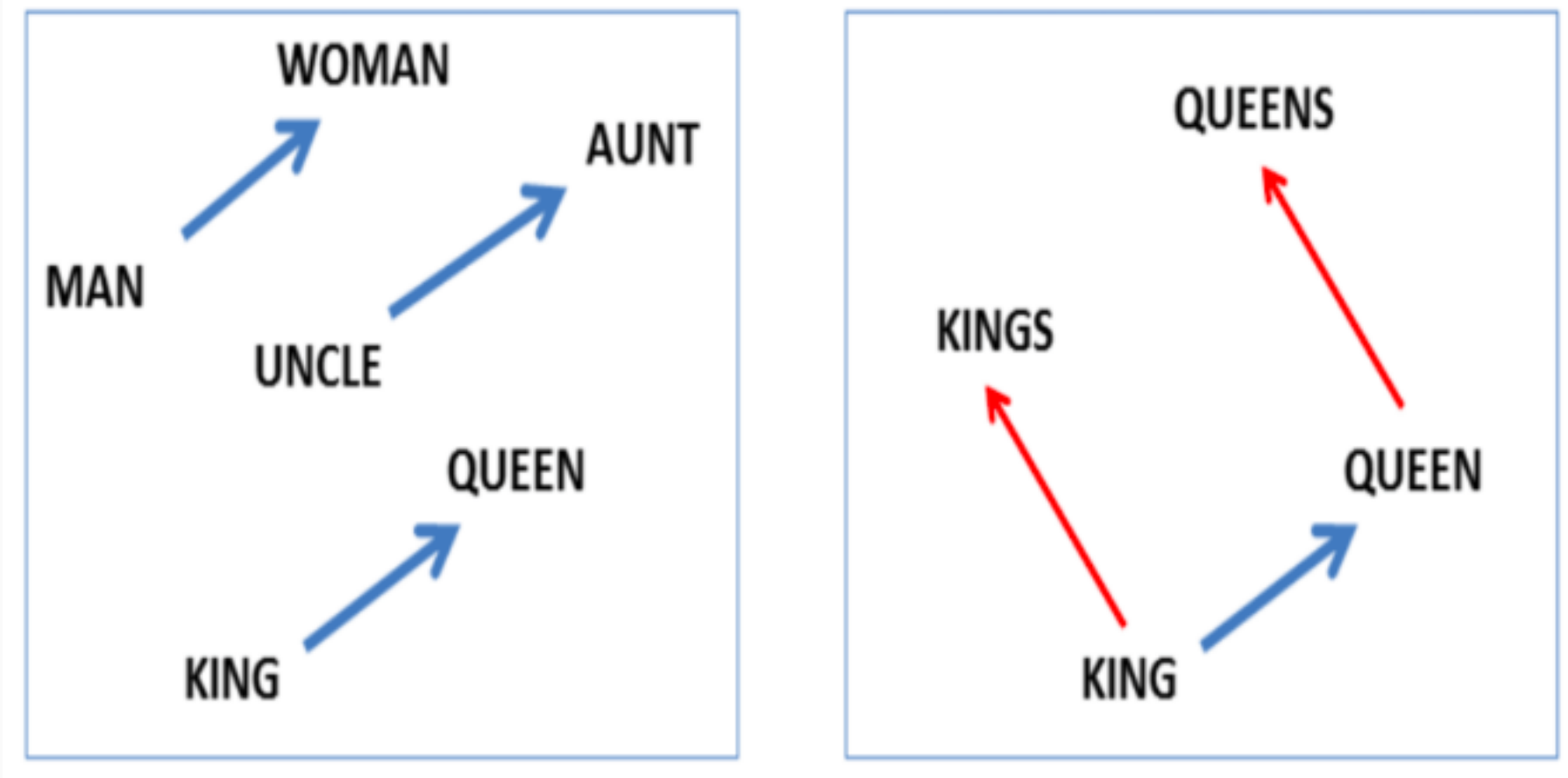
- Our vectorizers gave a random ID to each word
- What if we instead would represent each word by another vector representing its meaning?
- For, instance, 'doberman' and 'dog' should be represented by vectors that are close to each other in space, while 'kill' and 'walk' should be far from each other.

$\Rightarrow$ That's the idea behind word embeddings!

Or, more broadly: Can computers understand meanings, semantic relationships, different types of contexts?

You can literally calculate with words!

And answer questions such as "Man is to woman as king is to _____?"



semantic relationships vs. syntactic relationships