Build your own Semantle-clone

Stuts 72. Hamburg

Luuk Suurmeijer

What is Semantle?

- Semantle: https://semantle.com/
- Pedantle: https://cemantle.herokuapp.com/
- Workshop materials: https://github.com/LuukSuurmeijer/semantle_workshop

Why Semantle?

Representations under the hood of Semantle are everywhere in NLP

Good illustration for the question: How to approximate meaning with computers?

It’s just fun! :-)

How does Semantle work?

- We need some way to compare words based on their meaning
- Needs to be computable by a machine

Distributional Hypothesis (Firth/Harris):
“*You shall know a word by the company it keeps*”

Thought experiment

- What is a *bardiwac* ?
  - He handed her a glass of *bardiwac*
  - After drinking too much South African *bardiwac,* John woke up with a massive headache
  - *Bardiwac* tastes the richest when the grapes are exposed to a lot of sunlight

Thought experiment

- What is a *bardiwac* ?
  - He handed her a glass of *bardiwac*
  - After drinking too much South African *bardiwac,* John woke up with a massive headache
  - *Bardiwac* tastes the richest when the grapes are exposed to a lot of sunlight

Bardiwac seems to be a red wine!

Count Matrices

We might be able to get somewhere if we *count* the different words in the context of some text → This is something computers can do!

Count Matrices

We might be able to get somewhere if we *count* the different words in the context of some text → This is something computers can do!

An automobile is a wheeled motor vehicle used for transporting passengers.
A car is a form of transport, usually with four wheels and the capacity to carry around five passengers.
Transport for the London games is limited, with spectators strongly advised to avoid the use of cars.
The London 2012 soccer tournament began yesterday, with plenty of goals in the opening matches.
Giggs scored the first goal of the football tournament at Wembley, North London.
Bellamy was largely a passenger in the football match, playing no part in either goal.

Count Matrices

We might be able to get somewhere if we *count* the different words in the context of some text → This is something computers can do!

An automobile is a wheeled motor vehicle used for transporting passengers.
A car is a form of transport, usually with four wheels and the capacity to carry around five passengers.
Transport for the London games is limited, with spectators strongly advised to avoid the use of cars.
The London 2012 soccer tournament began yesterday, with plenty of goals in the opening matches.
Giggs scored the first goal of the football tournament at Wembley, North London.
Bellamy was largely a passenger in the football match, playing no part in either goal.

Term vocab: ⟨*wheel, transport, passenger, tournament, London, goal, match*⟩

	wheel	*transport*	*passenger*	*tournament*	*London*	*goal*	*match*
automobile	1	1	1	0	0	0	0
car	1	2	1	0	1	0	0
soccer	0	0	0	1	1	1	1
football	0	0	1	1	1	2	1

Count Matrices

We might be able to get somewhere if we *count* the different words in the context of some text → This is something computers can do!

An automobile is a wheeled motor vehicle used for transporting passengers.
A car is a form of transport, usually with four wheels and the capacity to carry around five passengers.
Transport for the London games is limited, with spectators strongly advised to avoid the use of cars.
The London 2012 soccer tournament began yesterday, with plenty of goals in the opening matches.
Giggs scored the first goal of the football tournament at Wembley, North London.
Bellamy was largely a passenger in the football match, playing no part in either goal.

What design choices are made in this example?

Term vocab: ⟨*wheel, transport, passenger, tournament, London, goal, match*⟩

	wheel	*transport*	*passenger*	*tournament*	*London*	*goal*	*match*
automobile	1	1	1	0	0	0	0
car	1	2	1	0	1	0	0
soccer	0	0	0	1	1	1	1
football	0	0	1	1	1	2	1

Count Matrices

We might be able to get somewhere if we *count* the different words in the context of some text → This is something computers can do!

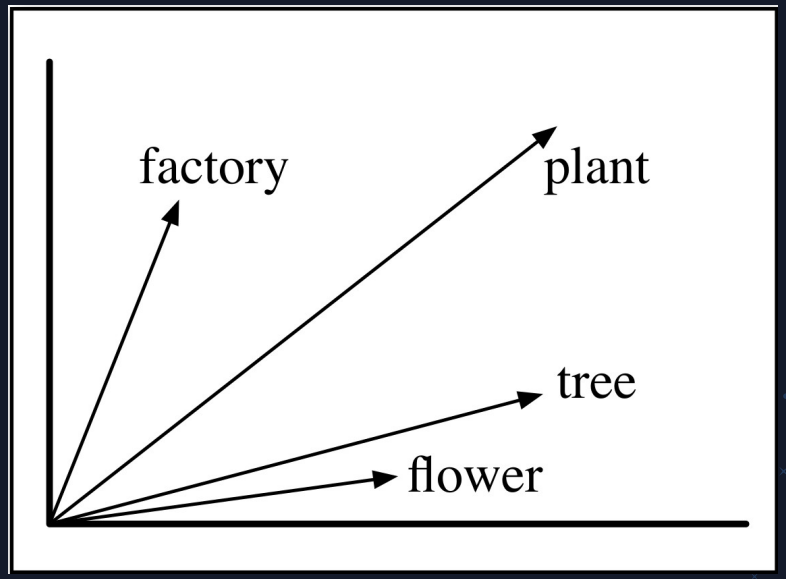
	Ever tried. Ever failed.	No matter. Try again.	Fail again. Fail better.
Ever	2	0	0
tried	1	0	0
failed	1	0	0
No	0	1	0
matter	0	1	0
Try	0	1	0
again	0	1	1
Fail	0	0	2
better	0	0	1

And here?

Vector Space Models

What's the point?

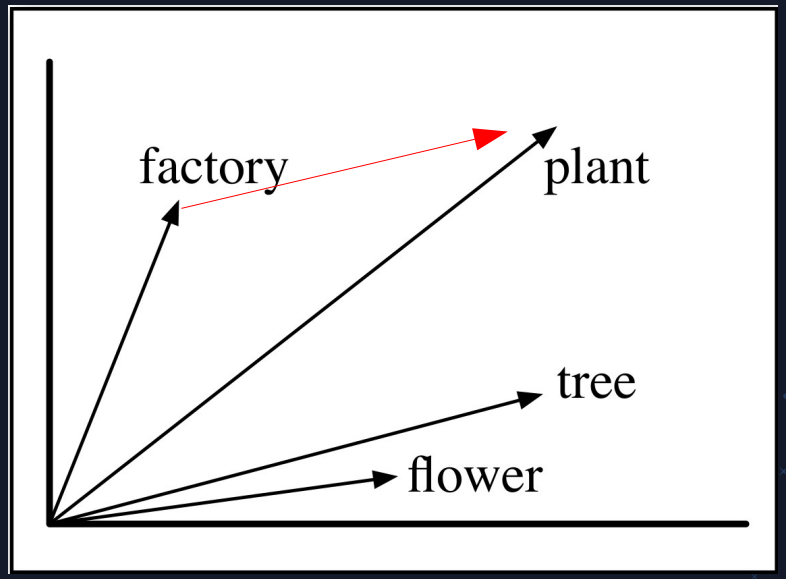
- This matrix encodes a *Vector Space*
- Vector Spaces have very attractive mathematical properties!



Vector Space Models

What's the point?

- This matrix encodes a *Vector Space*
- Vector Spaces have very attractive mathematical properties!

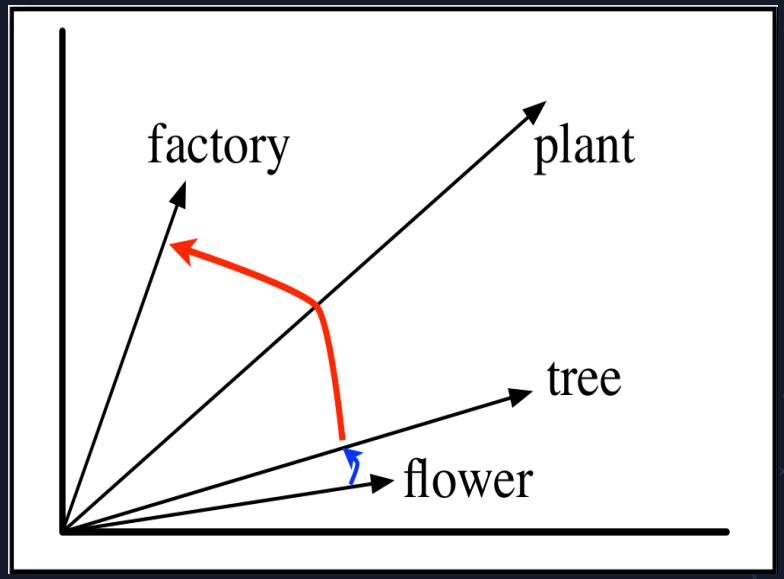


Adding/Subtracting vectors
factory + flower = plant

Vector Space Models

What's the point?

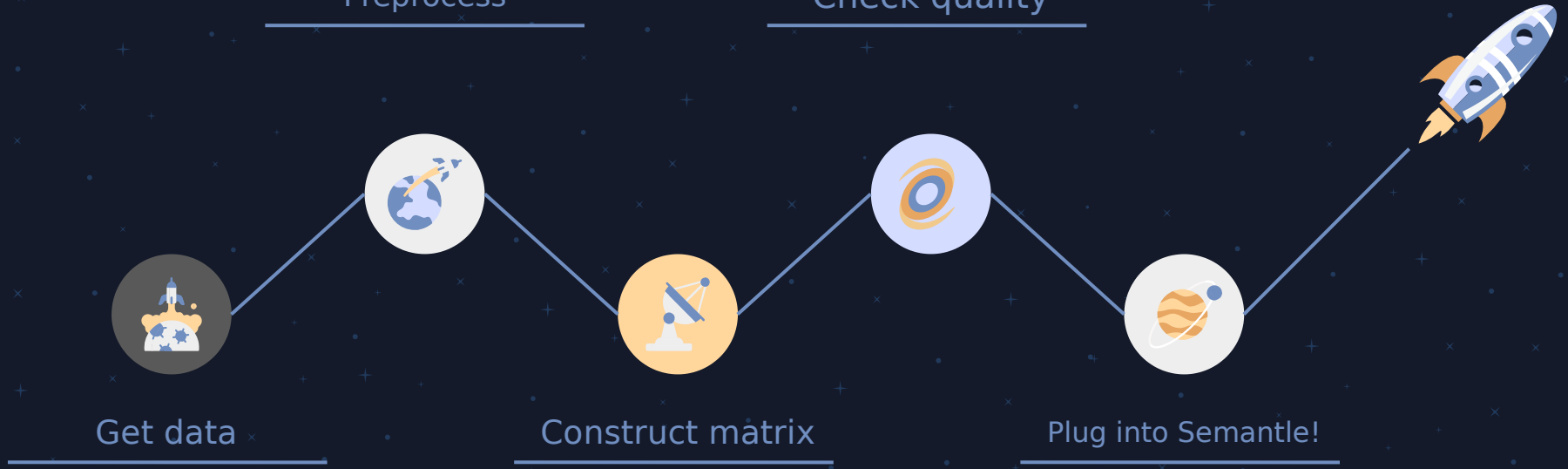
- This matrix encodes a *Vector Space*
- Vector Spaces have very attractive mathematical properties!



Adding/Subtracting vectors
factory + flower = plant

Computing angles between vectors (more on that later)

Steps


Preprocess
Check quality
Get data
Construct matrix
Plug into Semantle!

Preprocessing

- Tokenization
  - Split text into sensible word units
- Lemmatizing
  - Worse, worst => bad
- Stemming
  - Cried, cries => cri
- Stopword removal
  - Punctuation, function words, etc.
- Annotation
  - POS tags, semantic roles, etc.

Drawbacks of count matrices

- Language is Zipfian! → Most words never occur in most contexts

  1) Surprising (rare) occurrences contribute very little to the vector representation
  2) Matrix becomes very sparse (almost all entries are zero)

Boosting rare events: PMI

- Pointwise Mutual Information: How dependent is a word on a particular context?

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^{n_r}\sum_{j=1}^{n_c} f_{ij}} \tag{1}$$

$$p_{i*} = \frac{\sum_{j=1}^{n_c} f_{ij}}{\sum_{i=1}^{n_r}\sum_{j=1}^{n_c} f_{ij}} \tag{2}$$

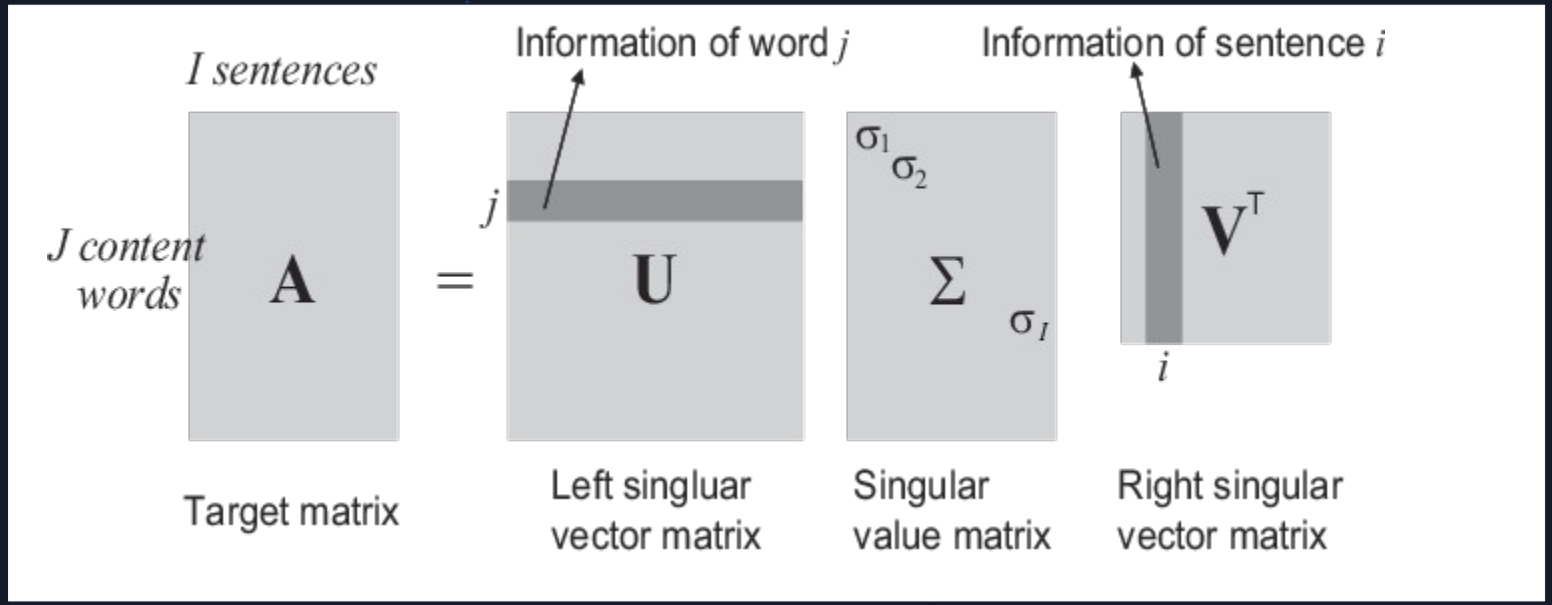
$$p_{*j} = \frac{\sum_{i=1}^{n_r} f_{ij}}{\sum_{i=1}^{n_r}\sum_{j=1}^{n_c} f_{ij}} \tag{3}$$

$$\mathrm{pmi}_{ij} = \log\left(\frac{p_{ij}}{p_{i*}p_{*j}}\right) \tag{4}$$

$$x_{ij} = \begin{cases} \mathrm{pmi}_{ij} & \text{if } \mathrm{pmi}_{ij} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Data Sparsity

- In order to make the matrix less sparse, we can use dimensionality reduction measure like *Singular Value Decomposition, Prinicipal Component Analysis* (won't talk about the details here)



Comparing vectors

- We can compare vectors by calculating the angle between them. This is known as the *cosine similarity*
  - Number between -1 and 1
  - cos(a,b) = 1 → *a* and *b* point in the same direction
  - cos(a,b) = -1 → *a* and *b* point in opposite directions
  - cos(a,b) = 0 → *a* and *b* are orthogonal (perpendicular)

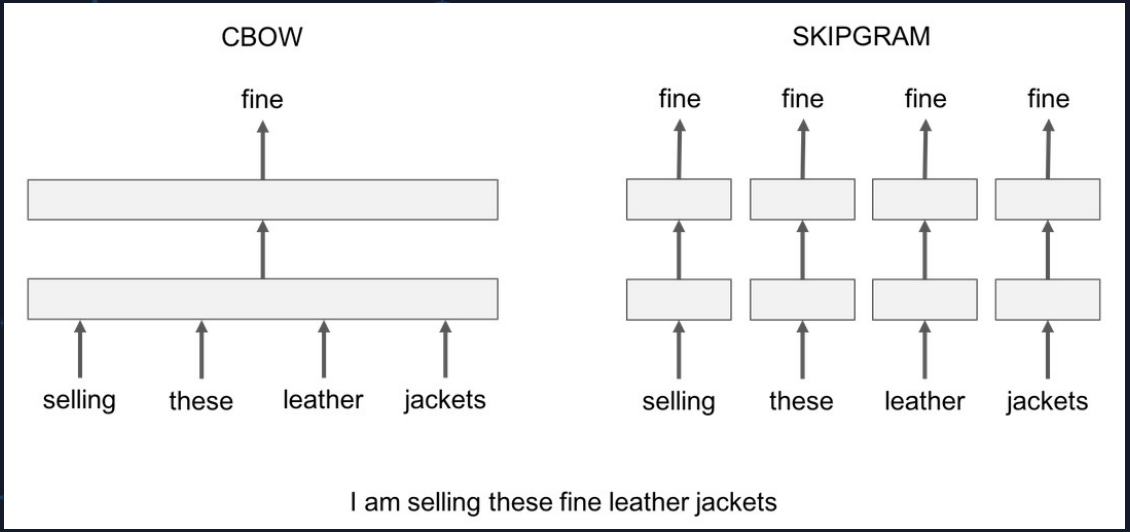
$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \cdot \sum_{i=1}^{n} y_i^2}} \tag{10}$$

$$= \frac{\mathbf{x} \cdot \mathbf{y}}{\sqrt{\mathbf{x} \cdot \mathbf{x}} \cdot \sqrt{\mathbf{y} \cdot \mathbf{y}}} \tag{11}$$

$$= \frac{\mathbf{x}}{\|\mathbf{x}\|} \cdot \frac{\mathbf{y}}{\|\mathbf{y}\|} \tag{12}$$

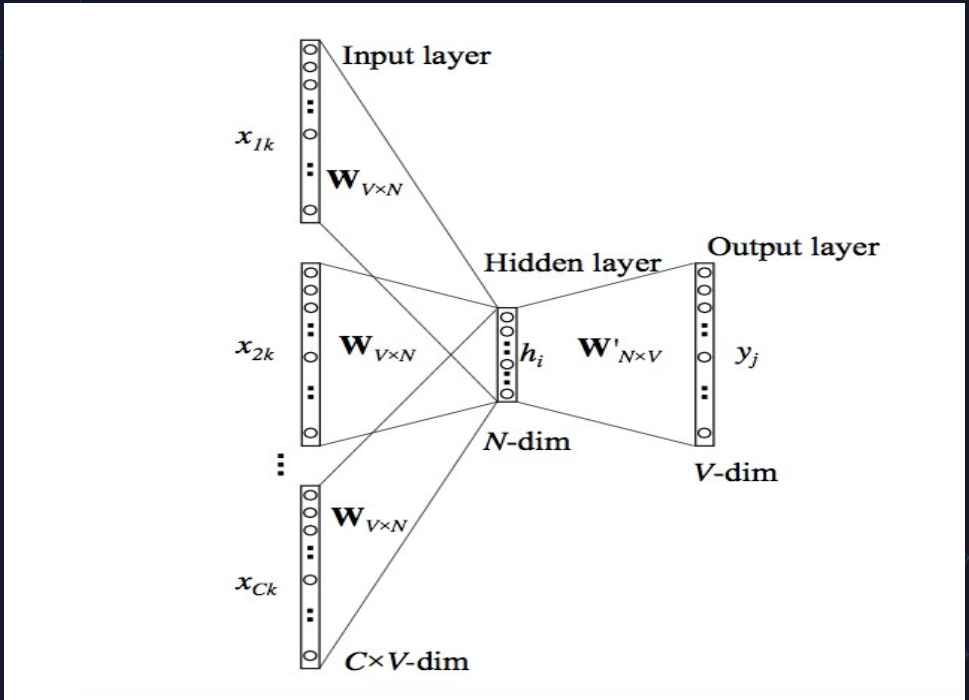
Beyond counts: Word Embeddings

Idea: Train a neural network to predict words from contexts and use the hidden representations as “embeddings”
- Popular algorithm: Word2Vec



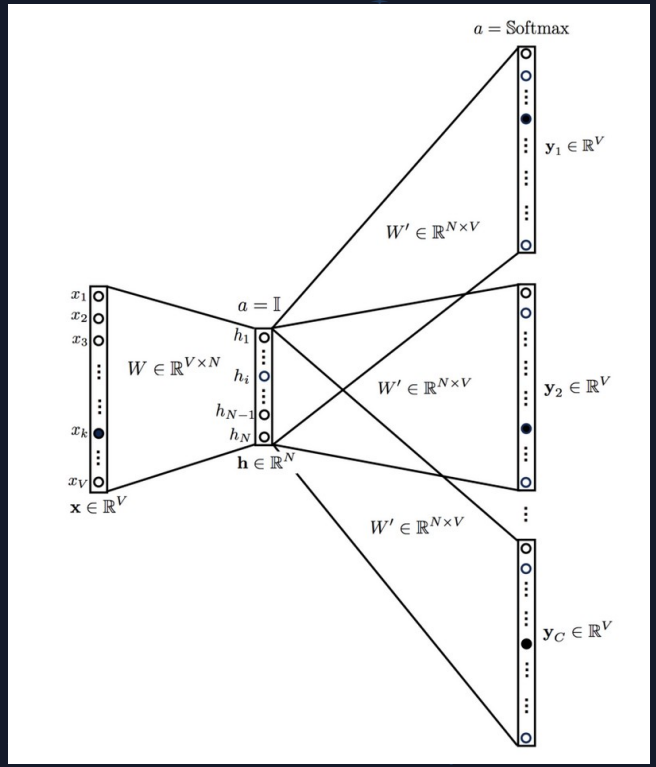
Beyond counts: Word Embeddings

Example of CBOW (Continuous bag of words)



Beyond counts: Word Embeddings

Example of Skipgram



Beyond counts: Word Embeddings

Skipgram:
- Better for small amount of data
- Better vectors for rare words

CBOW:
- Faster algorithm
- Better with large amount of data

Python implementation of word2vec:
https://github.com/LakheyM/word2vec/blob/master/word2vec_SGNS_git.ipynb

Other popular algorithm: FastText (uses ‘subword’ information):
fasttext.cc

Evaluating Word Embeddings

How can we tell our vectors/embeddings are ‘good’?
- Word analogy tasks

V(king) – V(man) + V(woman) = ?

V(france) – V(paris) + V(rome) = ?

V(Cu) – V(copper) + V(gold) = ?

Evaluating Word Embeddings

How can we tell our vectors/embeddings are ‘good’?
- Word analogy tasks

V(king) – V(man) + V(woman) = V(queen)

V(france) – V(paris) + V(rome) = V(italy)

V(Cu) – V(copper) + V(gold) = V(Au)

The vector is ‘correct’ if the predicted word has the highest cosine similarity to the intended vector (or if its in the top N closest)

Some resources

These slides + some papers + exercises
https://github.com/LuukSuurmeijer/semantle_workshop

Algorithm for playing Codenames using word2vec
https://github.com/thomasahle/codenames

Algorithm for playing Semantle using word2vec
https://github.com/womogenes/semantle-bot

Python package with pretrained word2vec embeddings
https://pypi.org/project/gensim/

Ready to launch your own Vector Space Rocket?