Machine Learning

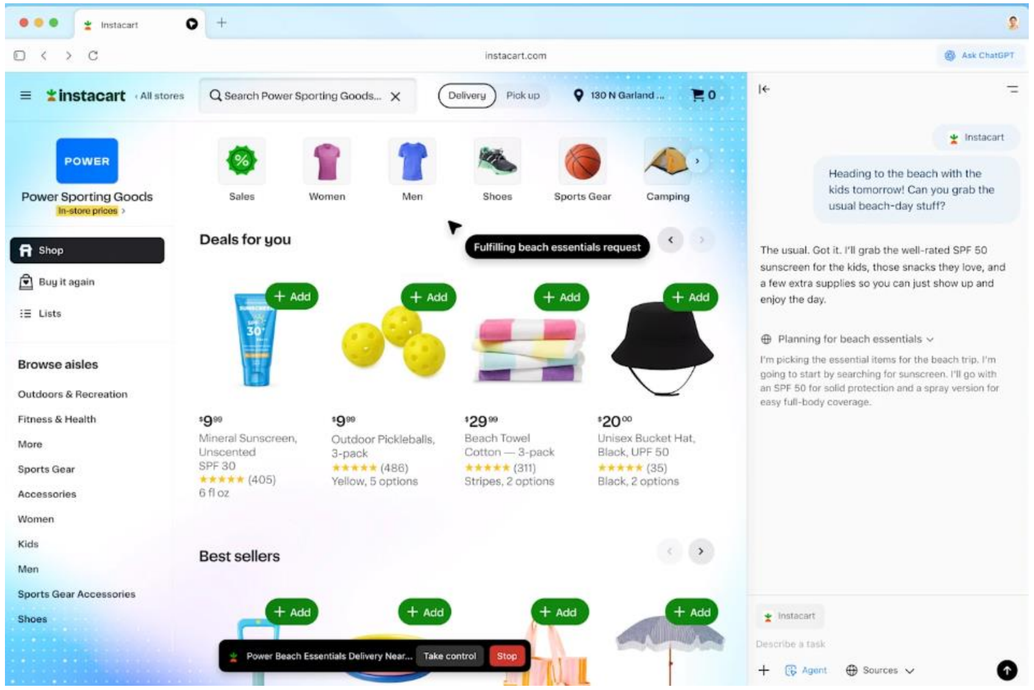
7. dimensionaliteitsreductie, word embeddings en RNN's

ML: Actueel

OpenAI launches an AI-powered browser: ChatGPT Atlas

TechCrunch

OpenAI announced Tuesday the launch of its AI-powered browser, ChatGPT Atlas, a major step in the company's quest to unseat Google as the main way people find information online.



DANΞ
@cryps1s · Follow

Yesterday we launched ChatGPT Atlas, our new web browser. In Atlas, ChatGPT agent can get things done for you. We're excited to see how this feature makes work and day-to-day life more efficient and effective for people.

ChatGPT agent is powerful and helpful, and designed to be Show more

6:40 PM · Oct 22, 2025

The glaring security risks with AI browser agents

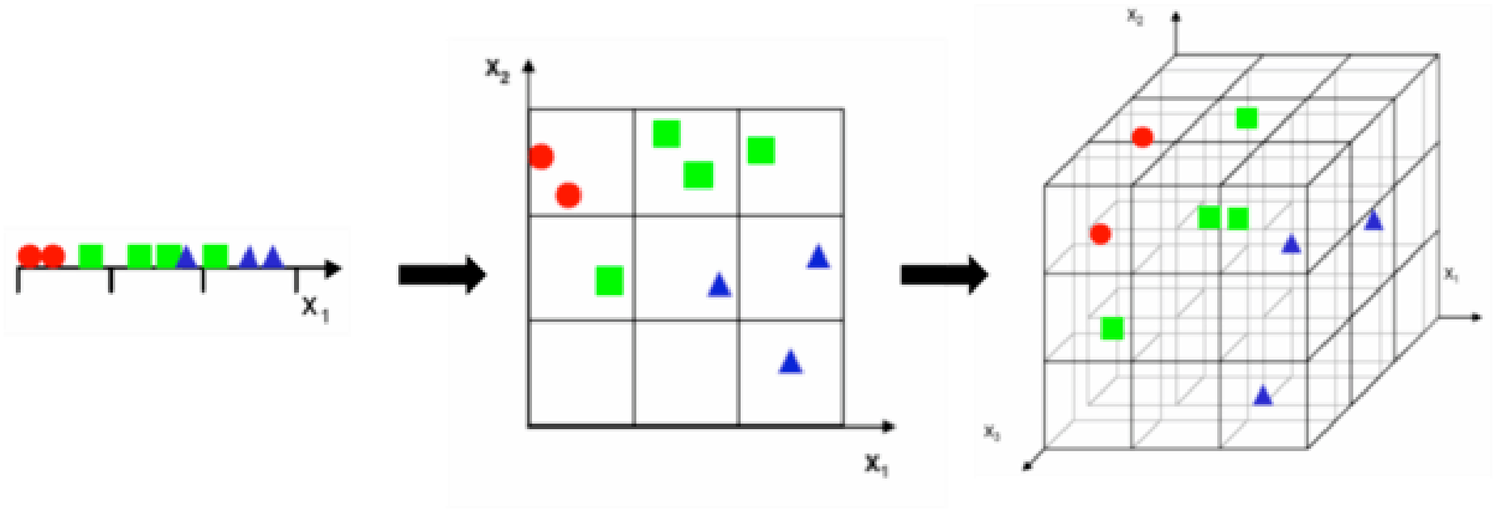
The main concern with AI browser agents is around "prompt injection attacks," a vulnerability that can be exposed when bad actors hide malicious instructions on a webpage. If an agent analyzes that web page, it can be tricked into executing commands from an attacker.

Onderwerpen

- Dimensionaliteitsreductie
  - Hoe verminder je het aantal features met zo min mogelijk informatieverlies?
- Word embeddings
  - Hoe codeer je woorden zodat een neuraal netwerk ermee kan werken?
- Recurrente neurale netwerken
  - Hoe bouw je een netwerk met context en geheugen, bijvoorbeeld voor NLP?

ml:dimensionality reduction

Dimensionality: Blessing or Curse?



Bron: commonlounge.com

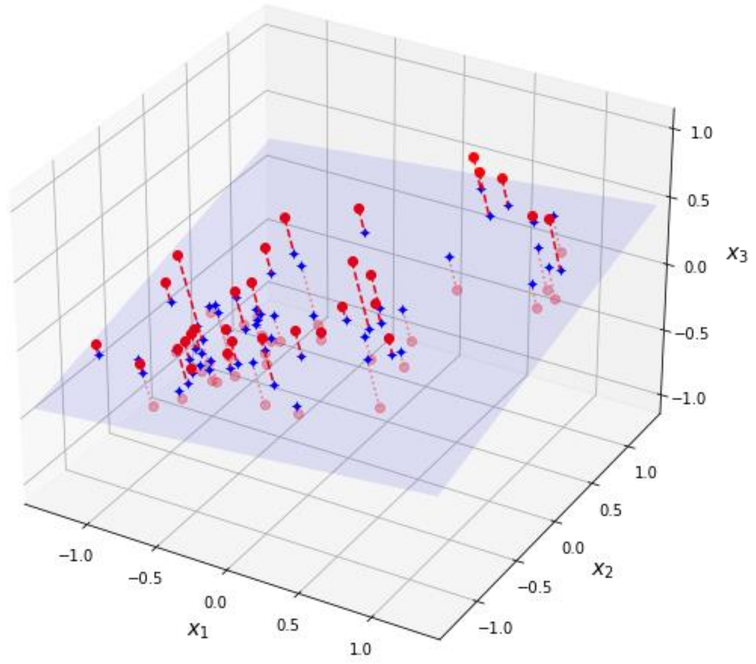
Curse of Dimensionality

- Elke feature erbij betekent een extra dimensie
  - Exponentiële groei
  - Performance-problemen
- Meer dimensies → datapunten komen verder uit elkaar te liggen
    - 2D-eenheidsvierkant: gem. afstand 0,52
    - 3D-eenheidskubus: gem. afstand 0,66
    - 1mD-eenheidshyperkubus: gem. afstand 408,25
  - Data wordt steeds ijler/*sparser*
  - **Exponentieel** meer data nodig, anders risico op **overfitting** van model
  - Clustering wordt erg lastig

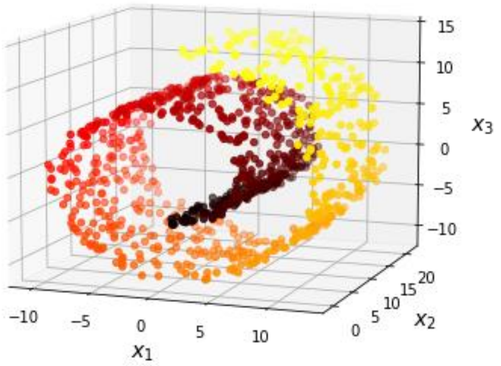
Dimensionaliteitsreductie

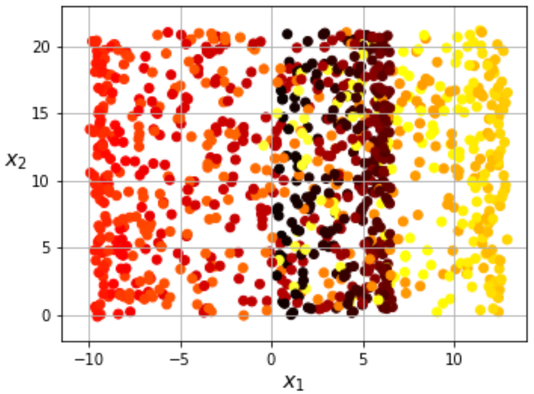
- Dus: problemen met performance, clustering en overfitting
- Daarom: aantal dimensies terugbrengen
- Diverse technieken
  - Projectie
  - Manifold Learning
  - Principal Components Analysis

Projectie

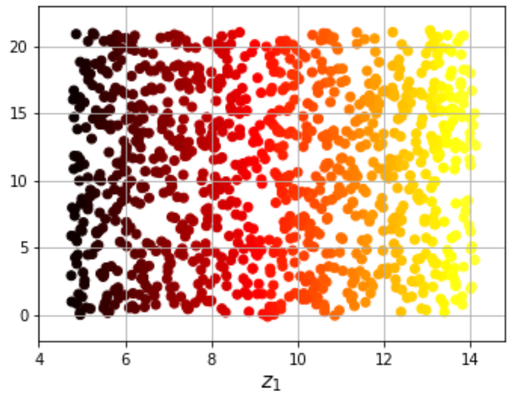


Manifold Learning





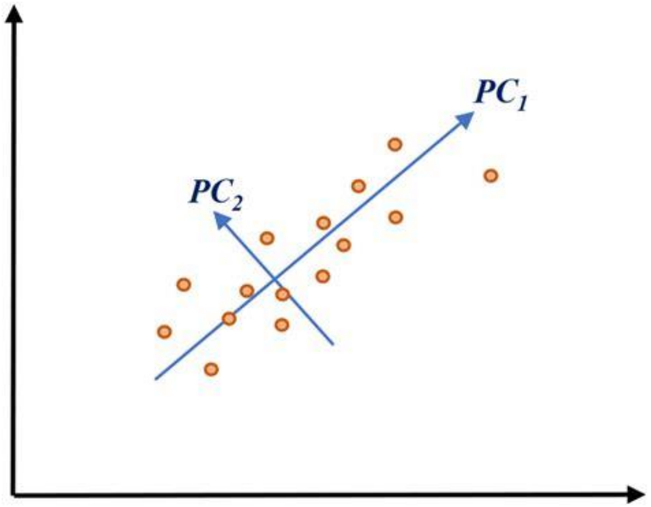
2D doorsnede, parallel aan x1



2D doorsnede, opgerold in de 3e dim.

Principal Components Analysis (PCA)

- Ook bekend als Factoranalyse
- Voorbeeld van **unsupervised** learning
  - Net als Clustering
- Transformatie van de oorspronkelijke data
- Aantal features = max. aantal dimensies
- In 2 dimensies vind je max. 2 PC’s
  - De eerste lijkt op lineaire regressie
  - De tweede staat er haaks op
- Meer dimensies => meer PC’s
  - Steeds haaks op het hyperplane van de vorige PC’s
- Idee: met minder dimensies (#PC < #dim) zoveel mogelijk van de feature-informatie behouden



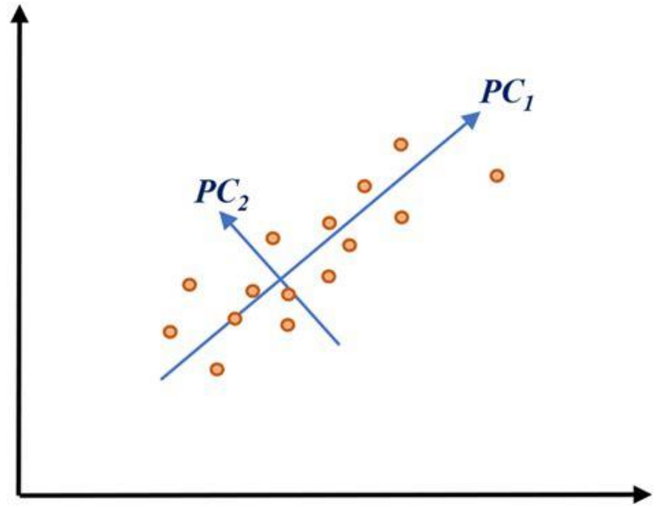
Bron: researchgate.net

Hoe zat het ook alweer: variantie

- $\sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n}$
- Gemiddelde gekwadrateerde afwijking van het gemiddelde
- Wortel van de variantie = standaarddeviatie
- Zegt dus iets over de **spreiding** van de data

Wat is een Principal Component (PC)?

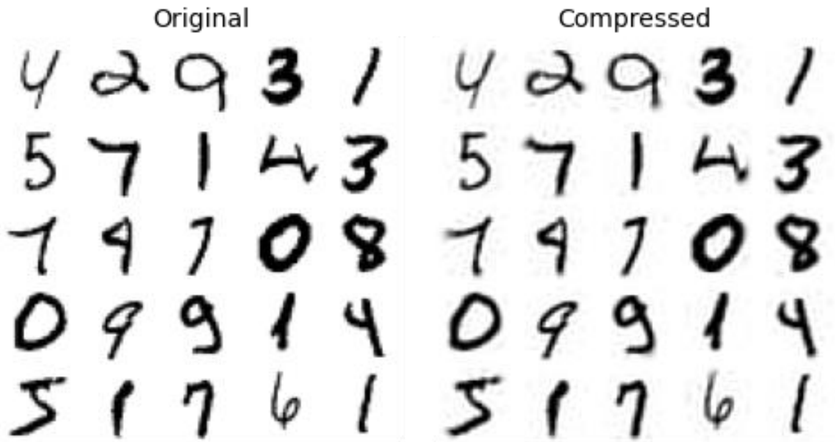
- Elke PC verklaart een percentage van de variantie van de data
  - Keuze van de as zodanig dat dit % zo hoog mogelijk is
  - Of: zo klein mogelijke MSD tussen data en projectie daarvan op de as
- Elke PC is een **combinatie van features**
  - Voorbeeld:
    - PC1 = 0.5 F1 + 0.2 F2
    - PC2 = -0.2 F1 + 0.05 F2



Bron: researchgate.net

Toepassingen van PCA

- Dimensionaliteitsreductie
- Compressie

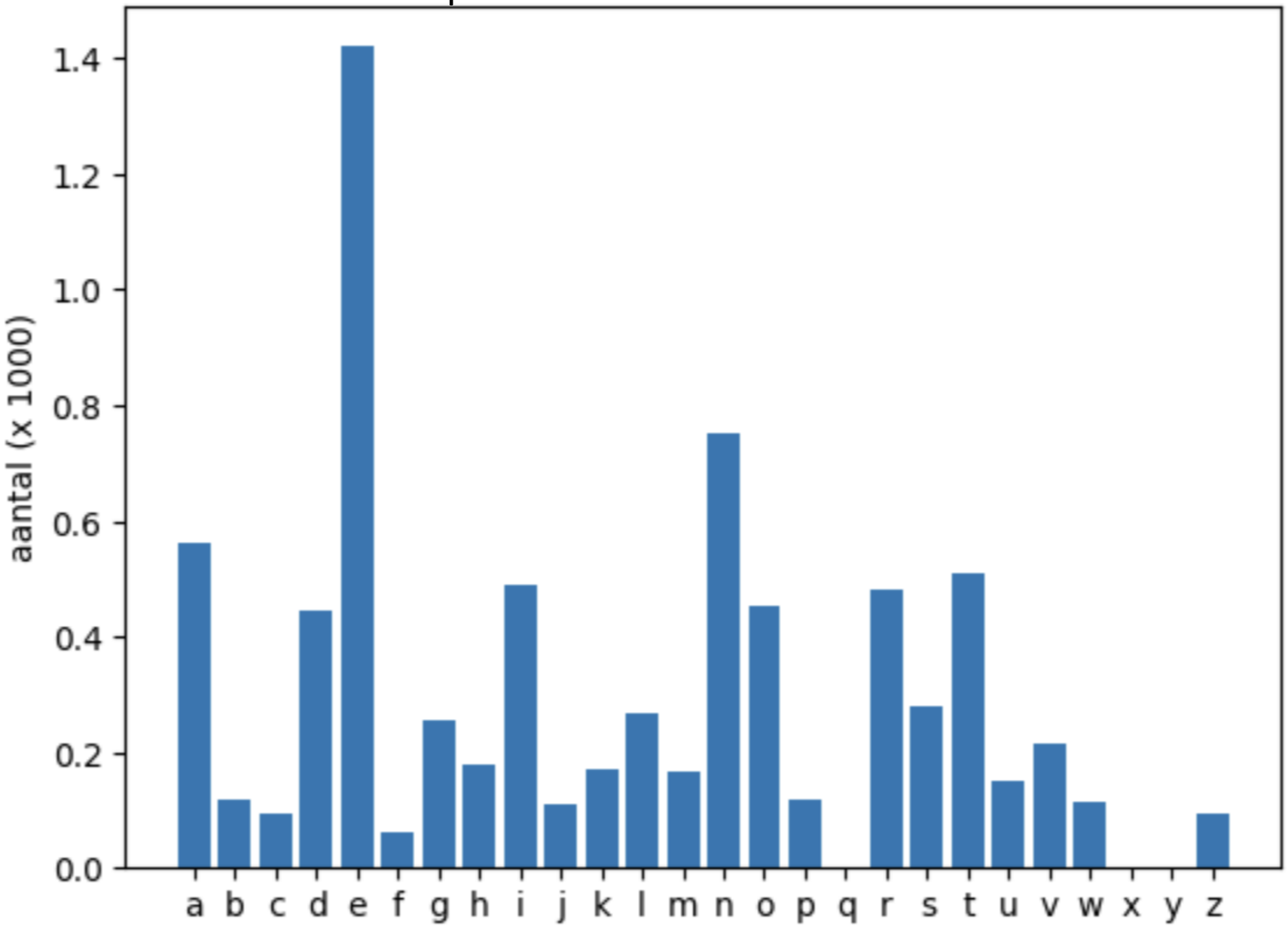


Intermezzo: live coding

- Notebook over PCA met scikit-learn

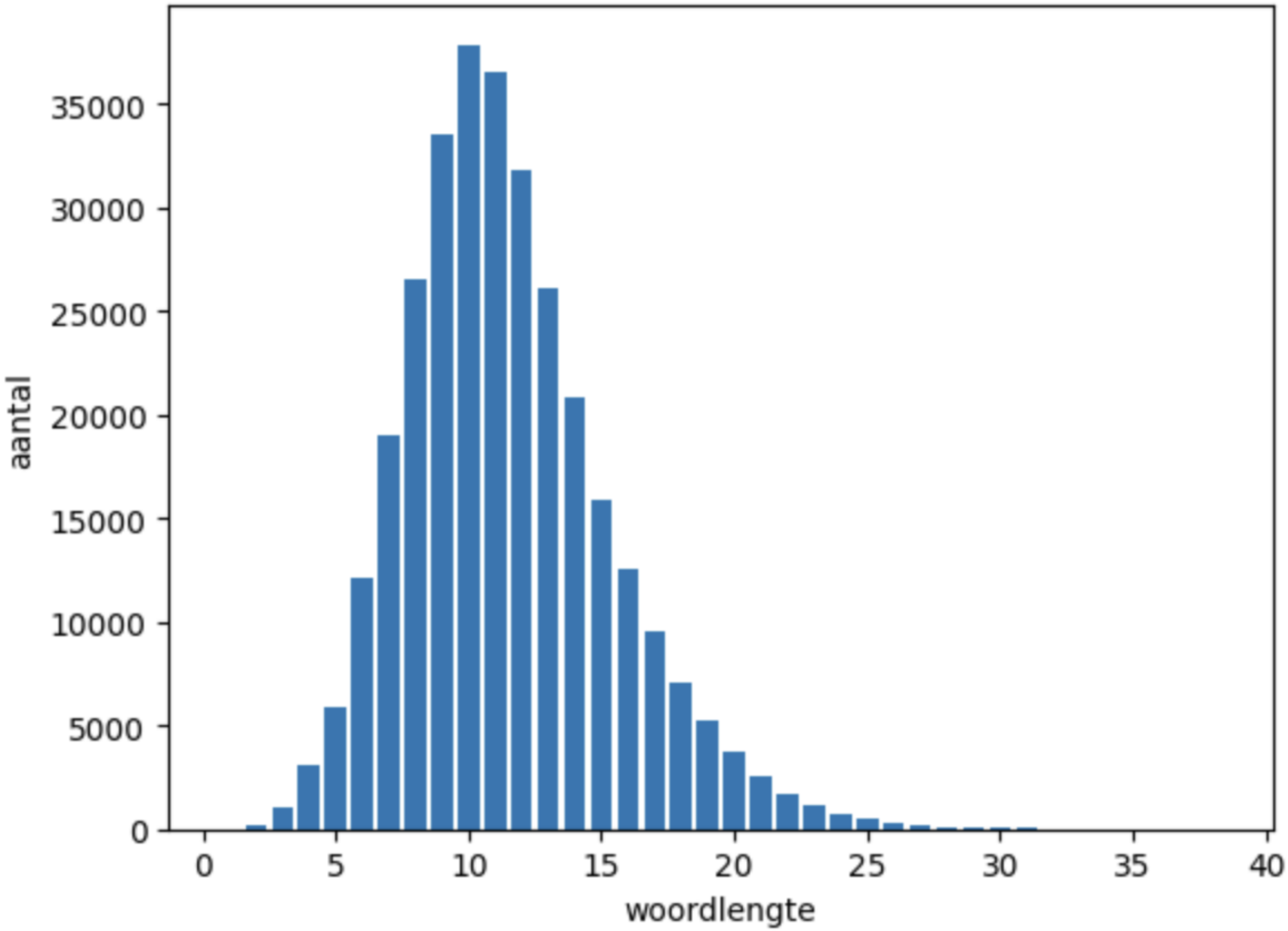
taal coderen: naïeve aanpak

Letterfrequenties in het Nederlands



https://www.sttmedia.com/characterfrequency-dutch

woordgrootte in het Nederlands



https://www.sttmedia.com/characterfrequency-dutch

```python
w_freqs = list(df['percentage'])
l_freqs = list(ldf['percentage']/100)

result = ''

# we maken een stuk tekst van veertig woorden
for _ in range(40):
    w_lengte = np.random.choice(np.arange(len(w_freqs)), p=w_freqs)
    for _ in range(w_lengte):
        result += chr(ord('a') + np.random.choice(np.arange(26), p=l_freqs))
    result += ' '

result
```

p : ***1-D array-like, optional***

The probabilities associated with each entry in a. If not given, the sample assumes a uniform distribution over all entries in `a`.

(https://numpy.org/doc/stable/reference/random/generated/numpy.random.choice.html)

evahhose te eoiertneak rdreadg idhs adi al dsdlo hgsbn egveint ui edj vilejrees liiniga epejithv atree tf nojatt ea nod trnnir etesoejmltrhal ztnilsentcr nrsrmeg gru dlptr asone beaoebjee leimf touhrl tmr lntvfeueal ge iaeeeavwbl oeer runvaliaw se hrnd baed iee heuvt naeeattn dgrnretilm er jsehvpet nmkp eydienee algfi ttp dtsp atpdjed rea vn penr koeee aenogra tne dn edetvsns tbidl zidive nkablsehi anaerodt tv pone eded nue tltr eoonggt btoa oond dsr norleeo ac ioioeu nrsbndn mrh

Op basis van *digrammen*

	A	B	C	D	…	Z
A	AA	AB	AC	AD		AZ
B	BA	BB	BC	BD		BZ
C	CA	CB	CC	CD		CZ
…						
Z	ZA	ZB	ZC	ZD		ZZ

Nederlandse digrammen

	digram	aantal	percentage
0	aa	27107958	1.655611
1	ab	952105	0.058150
2	ac	3883815	0.237203
3	ad	3747459	0.228875
4	ae	632509	0.038630

	digram	aantal	percentage
117	en	85425765	5.217355
121	er	52558556	3.209999
82	de	52042236	3.178465
13	an	36996593	2.259557
498	te	30808553	1.881624
108	ee	29967844	1.830278
160	ge	28092308	1.715730
221	in	27499971	1.679553
0	aa	27107958	1.655611
123	et	27080811	1.653953

deangemp iejawoorelro alkewestriietereds tond bbendeanza enomenet ik geienonu ekenpelutabe ineeti prllopooha geap oedale nkinenwaveus olzoerie etdtgear aadeuiziod viitonno ti bieglaonletsta beftaodeziui enlftrseellisi inreesatieitalbedihe stjnitil dodeee oeritejler epvede enonuu hewigechrdel etgeoouiin iegeou jcanrore keineepm enoobios areeseeiijak ljtkdoagetaaer keedinukgeas tsrdvekemershi lgthrgaatrdu geklelmstevren

iets minder naïeve aanpak


weten welke wet geldt
we t
weten welke wet geldt
et e
weten welke wet geldt
te n
weten welke wet geldt
n w
weten welke wet geldt
w e
weten welke wet geldt
we l
weten welke wet geldt
el k
weten welke wet geldt
lk e
input
voorspelling

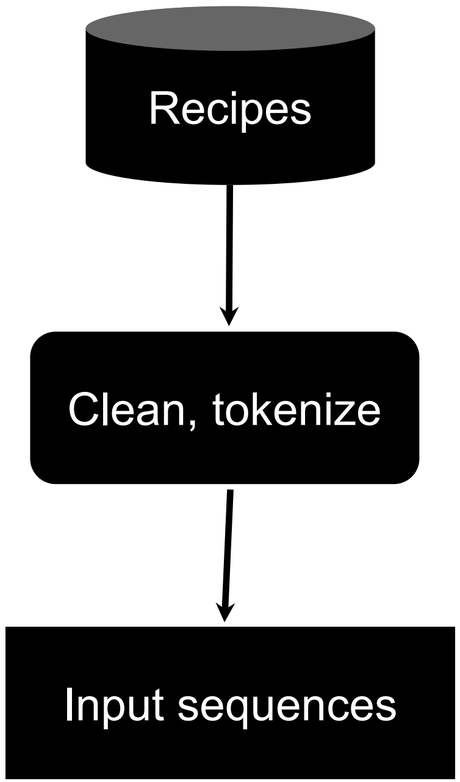
{'we':{'t':1, 'l':1}}

```
with open('data/wiki.txt', 'r') as f:
    data = ''.join([line.strip().lower() for line in f.readlines()])

model = NGramModel(4)
model.fit(data)
print(model.predict('afge', 300))
```

afgeleidt tot voorbeeldelijk, gen.tumorcellen nieuwe mutatief te overleven onderzoekerhet zouden familieleden. sommige tumorsuppressorgenen ande een tegen de typen als remmers.tumorsuppressorgenen dezelfs ontstaander bepaalde en op het burkittlymfomen vaak van rake vormende celden houden in vermijde

word embeddings



Recipe steps

“Stir until the cheese is melted and all ingredients are incorporated, about 3 minutes.”

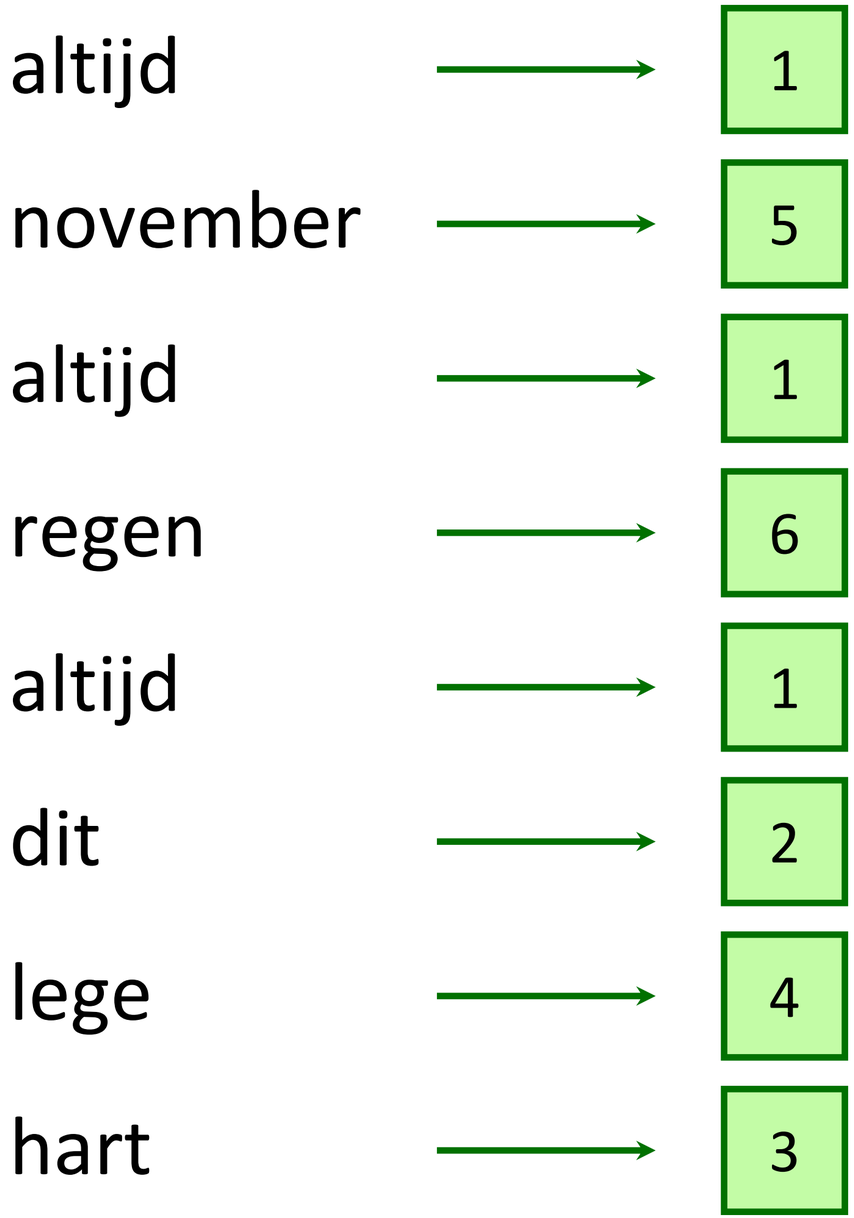
“flip back over and toast in a 400 degree oven for about 8 to 10 minutes.”

Corpus sentence

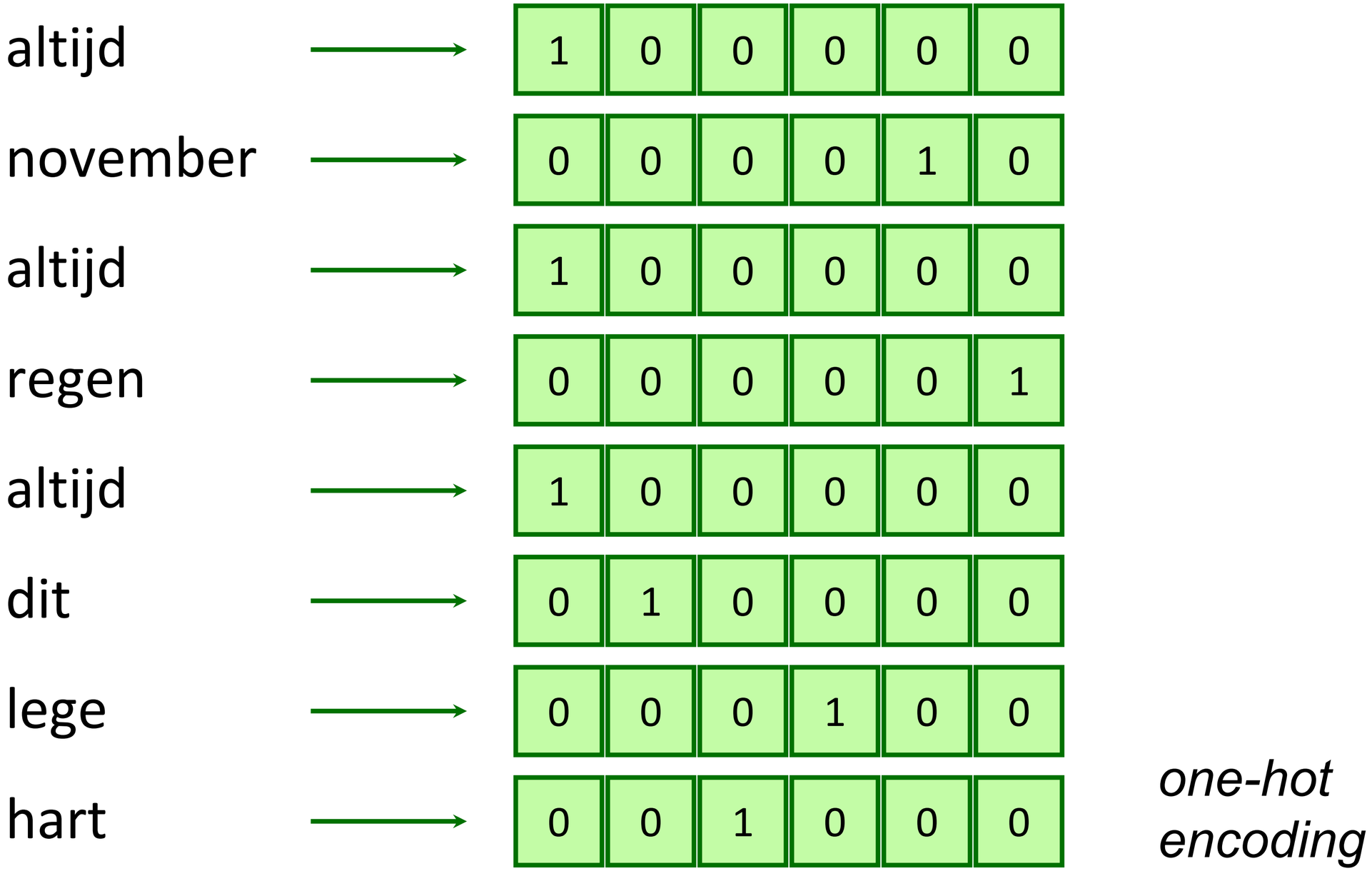
['stir', 'cheese', 'melt', 'ingredient', 'incorporate', 'minute']

['flip', 'toast', 'degree', 'oven', 'minute']

[['stir', 'cheese', 'melt', 'ingredient', 'incorporate', 'minute’], [‘melt’, ’butter’, ‘heavy’, ‘bottom’, ‘saucepan’, ‘medium’, ‘heat’, ‘whisk’, ‘flour’], ..., ['flip', 'toast', 'degree', 'oven', 'minute’]]


altijd
november
altijd
regen
altijd
dit
lege
hart
1
5
1
6
1
2
4
3

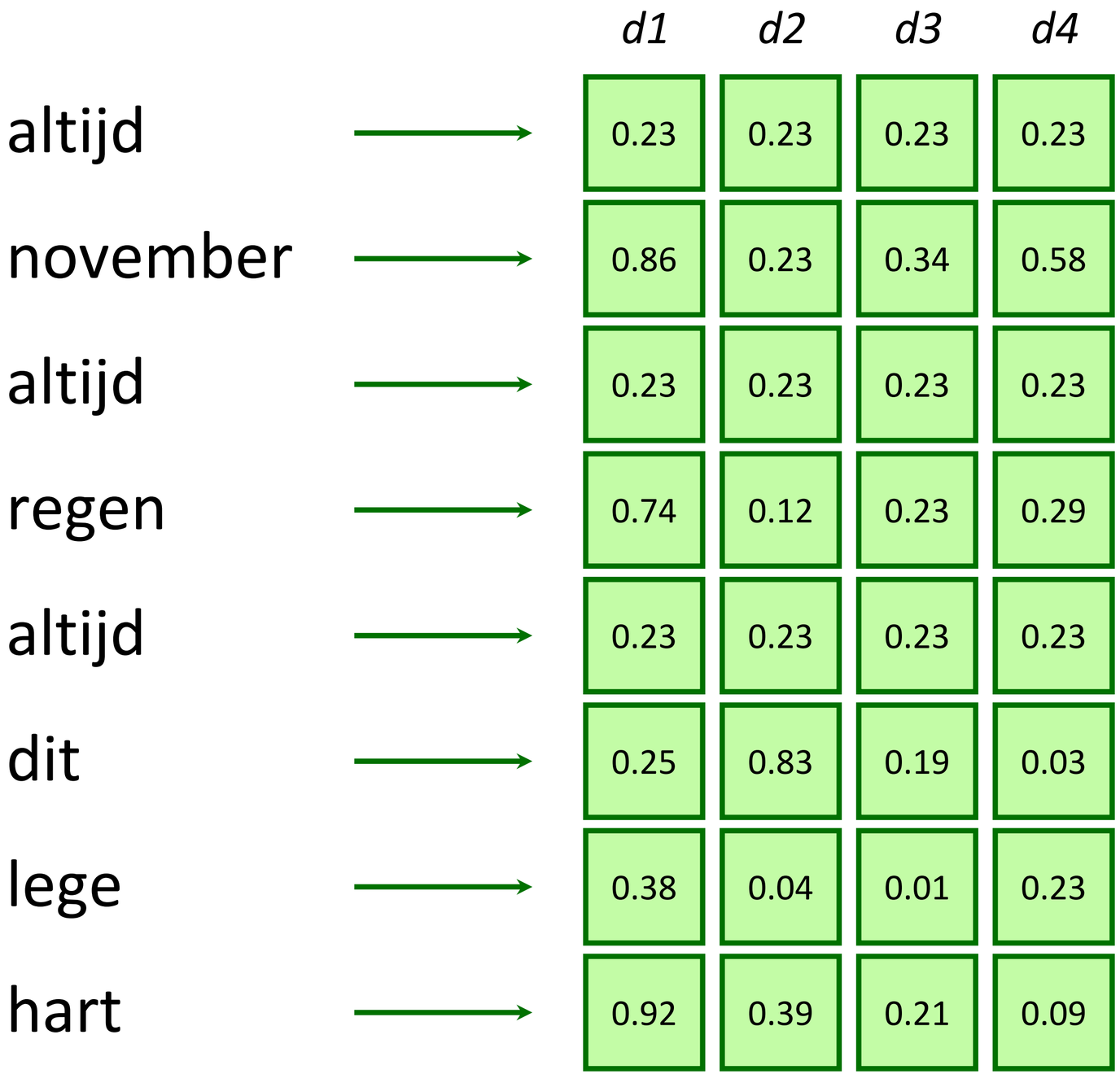
bag of words

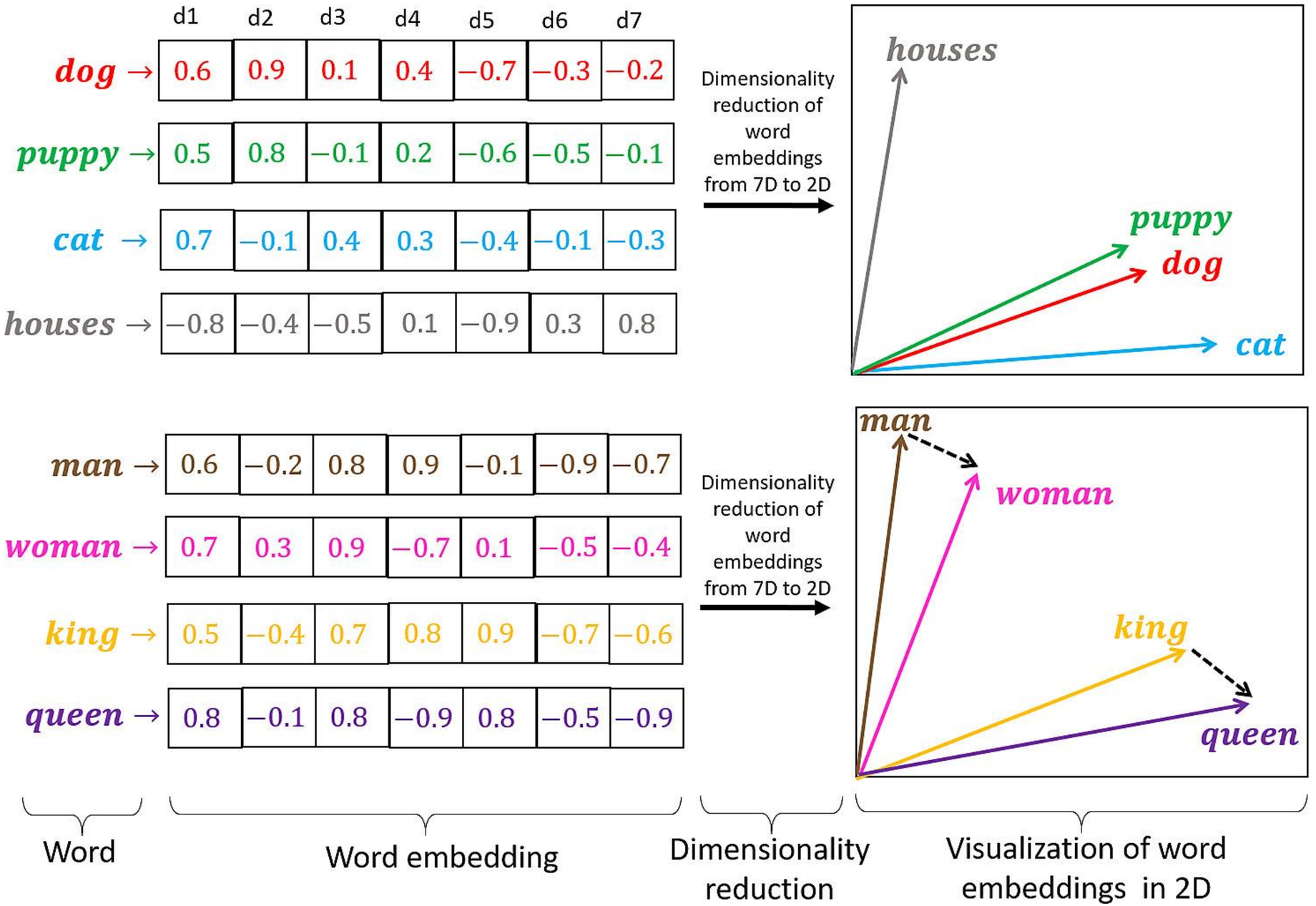

altijd
november
altijd
regen
altijd
dit
lege
hart
1 0 0 0 0 0
0 0 0 0 1 0
1 0 0 0 0 0
0 0 0 0 0 1
1 0 0 0 0 0
0 1 0 0 0 0
0 0 0 1 0 0
0 0 1 0 0 0
one-hot encoding

Probleem van homoniemen

Er zit geen koffie meer in **de bus**.

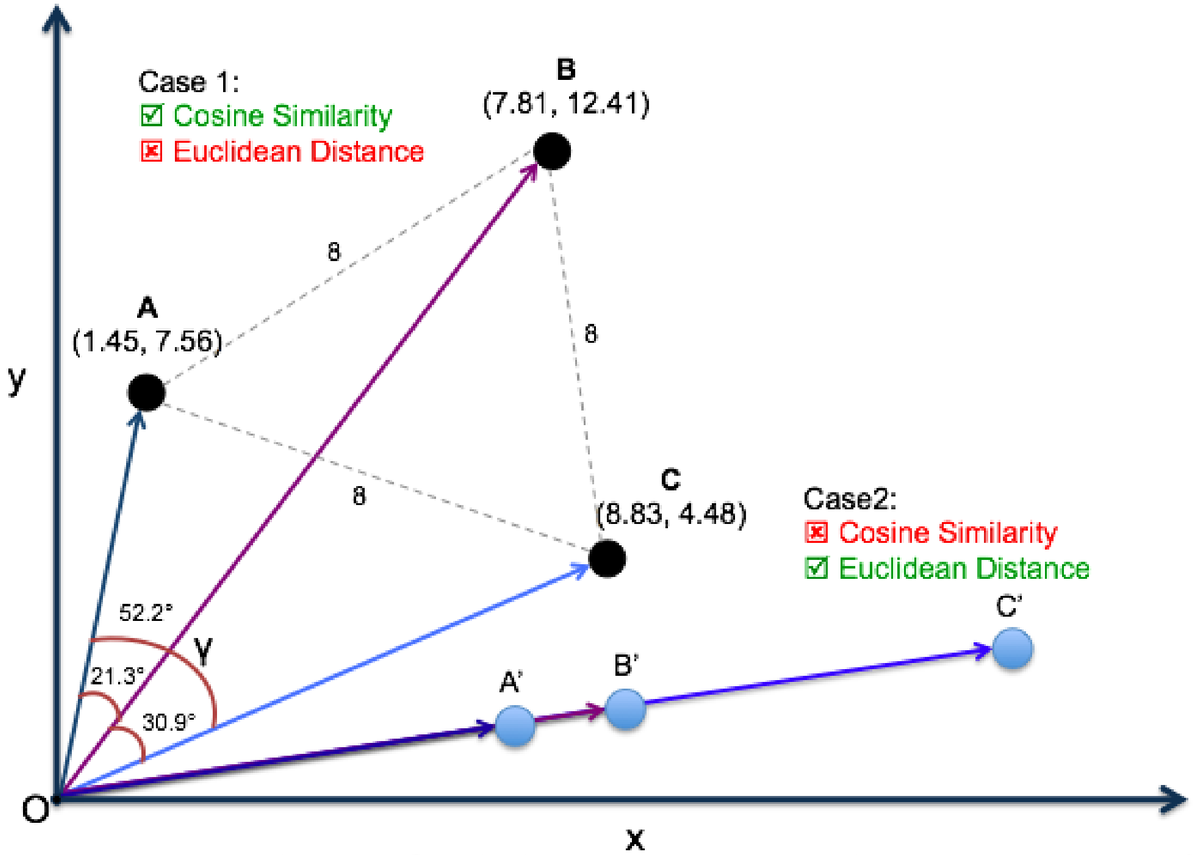
Nu is **de bus** weer te laat!


d1
d2
d3
d4
altijd
0.23
0.23
0.23
0.23
november
0.86
0.23
0.34
0.58
altijd
0.23
0.23
0.23
0.23
regen
0.74
0.12
0.23
0.29
altijd
0.23
0.23
0.23
0.23
dit
0.25
0.83
0.19
0.03
lege
0.38
0.04
0.01
0.23
hart
0.92
0.39
0.21
0.09



Afstands-/gelijkheidsmaten voor vectoren

- Euclidisch: afstand tussen de punten in de ruimte
- Cosinusgelijkheid: hoek tussen de vectoren



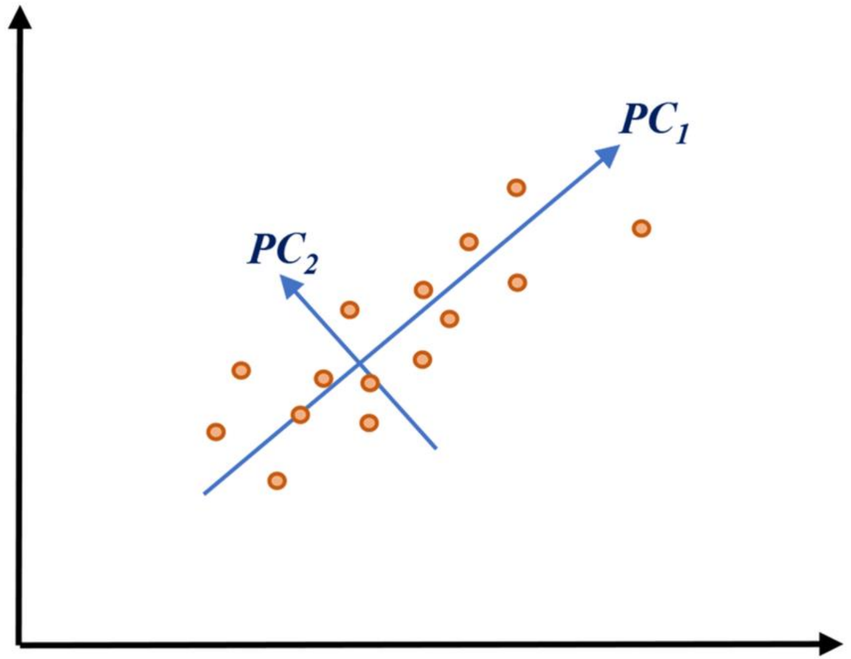
Bron:https://medium.com/@sasi24/cosine-similarity-vs-euclidean-distance-e5d9a9375fc8

$$d(\mathbf{A},\mathbf{B}) = d(\mathbf{B},\mathbf{A}) = \sqrt{(A_1 - B_1)^2 + (A_2 - B_2)^2 + \cdots + (A_n - B_n)^2}$$
$$= \sqrt{\sum_{i=1}^{n} (A_i - B_i)^2}$$

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

PCA (herhaling)

- Reductie van aantal features (dimensionality reduction)
- Bijvoorbeeld voor visualisatie of compressie
- Maakt nieuwe features (PC's) uit combinaties van de bestaande features
- Elke PC verklaart zoveel mogelijk variantie in de (resterende) data
- Kies de top-*n* PC's



Bron:researchgate.net

NLPL Repository WebVectors RegisterExplorer

ID	Download link	Vector size	Window	Corpus	Vocabulary size	Algorithm ▲	Lemmatization
32	Download	100	10	Basque CoNLL17 corpus	426736	Word2Vec Continuous Skipgram	False
33	Download	100	10	Bulgarian CoNLL17 corpus	628026	Word2Vec Continuous Skipgram	False
34	Download	100	10	Catalan CoNLL17 corpus	799020	Word2Vec Continuous Skipgram	False
35	Download	100	10	ChineseT CoNLL17 corpus	1935503	Word2Vec Continuous Skipgram	False
36	Download	100	10	Croatian CoNLL17 corpus	928316	Word2Vec Continuous Skipgram	False
37	Download	100	10	Czech CoNLL17 corpus	1767815	Word2Vec Continuous Skipgram	False
38	Download	100	10	Danish CoNLL17 corpus	1655886	Word2Vec Continuous Skipgram	False
39	Download	100	10	Dutch CoNLL17 corpus	2610658	Word2Vec Continuous Skipgram	False
40	Download	100	10	English CoNLL17 corpus	4027169	Word2Vec Continuous Skipgram	False
41	Download	100	10	Estonian CoNLL17 corpus	926795	Word2Vec Continuous Skipgram	False
42	Download	100	10	Finnish CoNLL17 corpus	2433286	Word2Vec Continuous Skipgram	False
43	Download	100	10	French CoNLL17 corpus	2567698	Word2Vec Continuous Skipgram	False
44	Download	100	10	Galician CoNLL17 corpus	363106	Word2Vec Continuous Skipgram	False
45	Download	100	10	German CoNLL17 corpus	4946997	Word2Vec Continuous Skipgram	False
46	Download	100	10	Greek CoNLL17 corpus	1183194	Word2Vec Continuous Skipgram	False
47	Download	100	10	Hebrew CoNLL17 corpus	672384	Word2Vec Continuous Skipgram	False

http://vectors.nlpl.eu/repository/

Verschillende algoritmen voor word-embeddings

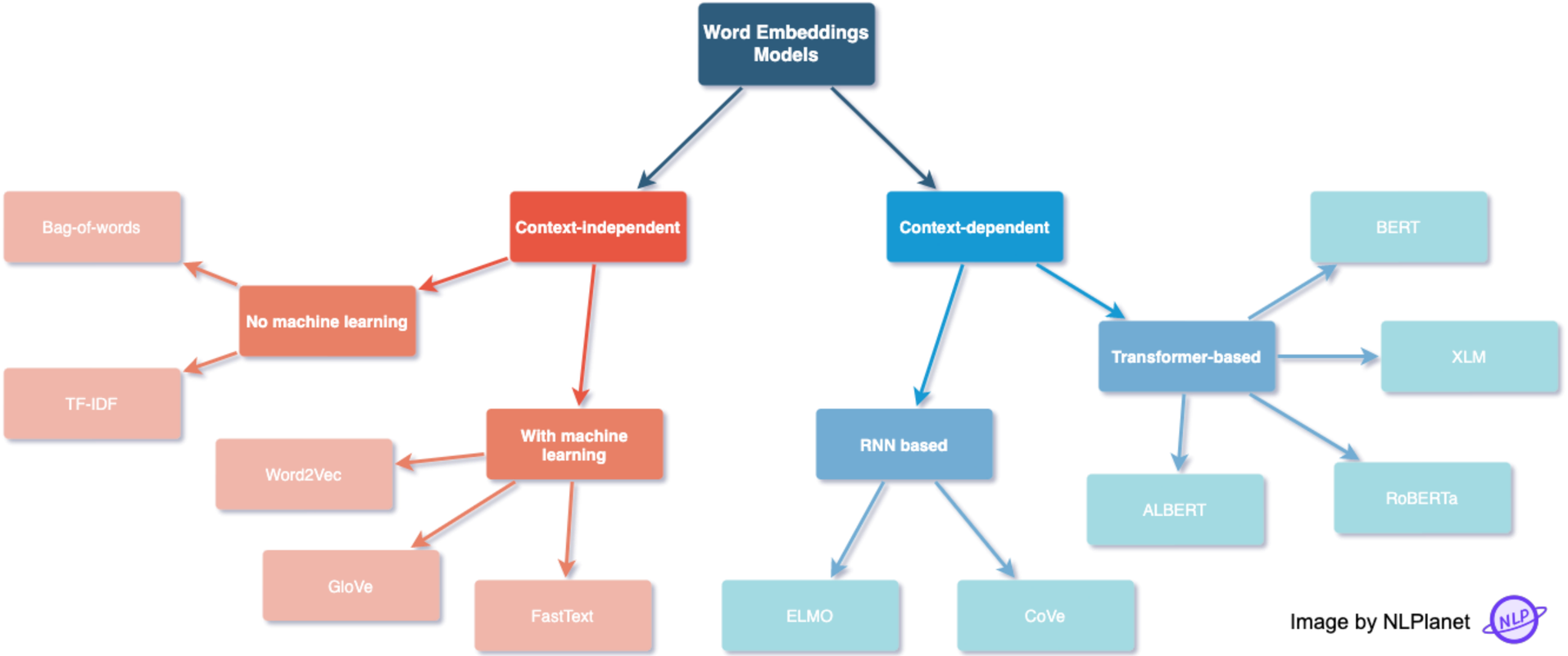
Skipgram: Voorspelt contextwoorden gegeven een doelwoord. Goed voor semantische relaties.

CBOW (Continuous Bag of Words): Voorspelt een doelwoord gegeven de contextwoorden. Snel en efficiënt voor frequentere woorden.

FastText: Breidt zowel Skipgram als CBOW uit door subwoorden te gebruiken, wat zorgt voor betere prestaties bij onbekende woorden en woorden met complexe morfologie.

Een veelgebruikt model, **Word2vec**, gebruikt een combinatie van CBOW en Skipgram om de embeddings te leren.

classificatie van embedding-modellen



voorgetrainde modellen

https://radimrehurek.com/gensim/

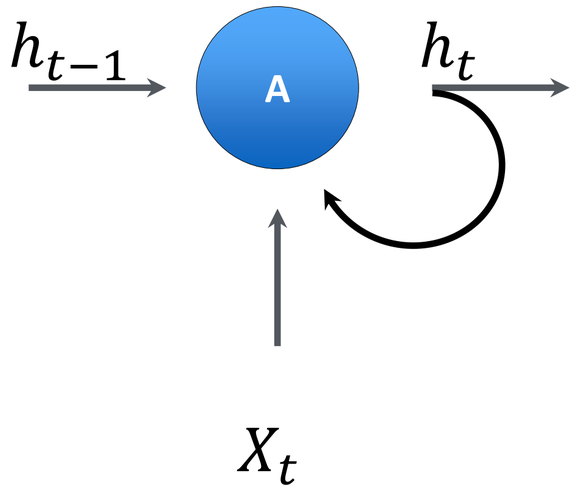
https://github.com/google-research/bert

https://spacy.io/

recurrente neurale netwerken

sequentiële data

geluidsgolven zijn sequentiële data, net als tekst 😎

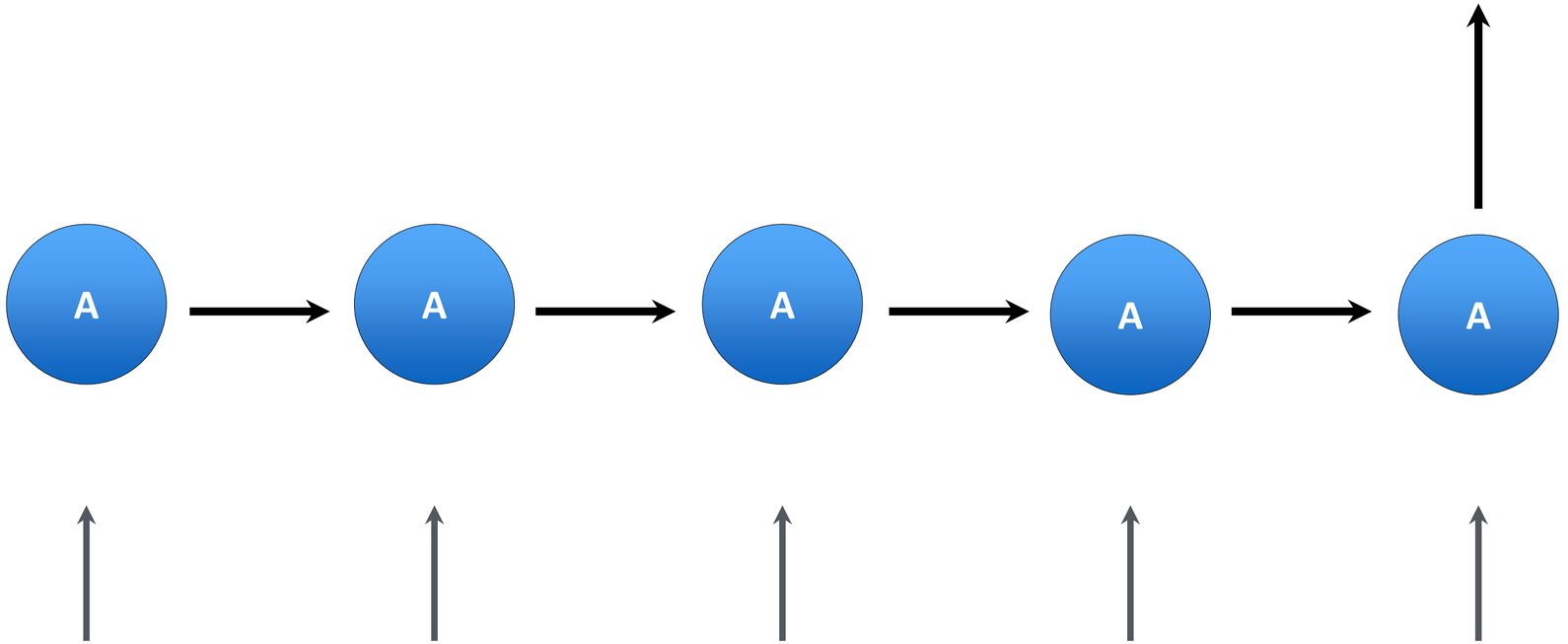

h_{t-1}
A
h_t
X_t

RNN’s: vier soorten

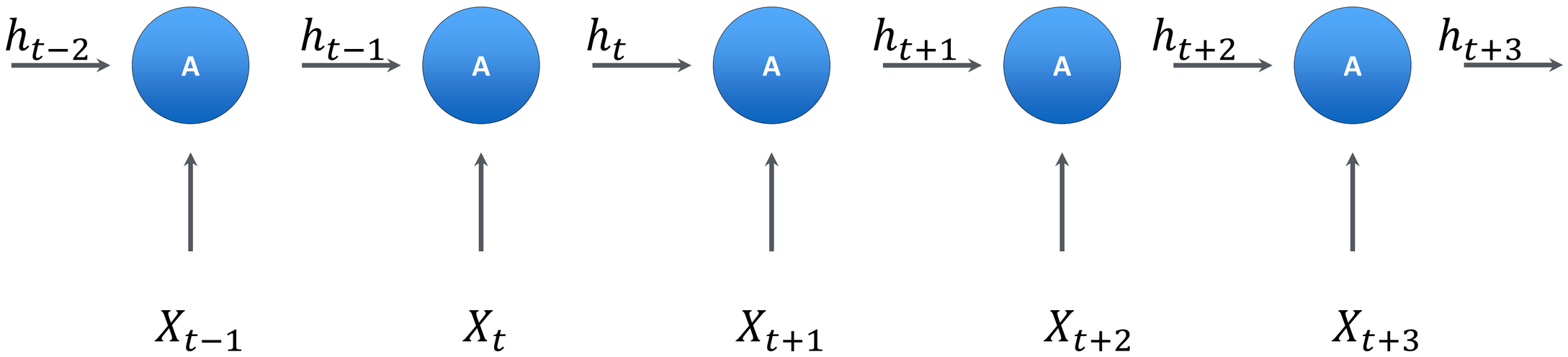
- Sequence to Sequence
  - een tijdreeks voorspellen o.b.v. het verleden, bijv. beurskoersen
- Sequence to Vector
  - alleen de laatste output gebruiken
  - bijv. sentiment-analyse
- Vector to Sequence
  - genereren van tekst, muziek, captions bij plaatjes e.d.
- Encoder-Decoder
  - bestaat uit sequence-to-vector en vector-to-sequence
  - bijv. vertalen, prompts beantwoorden
  - ligt aan de basis van het Transformer-model

Sequence to Vector

unrolled through time

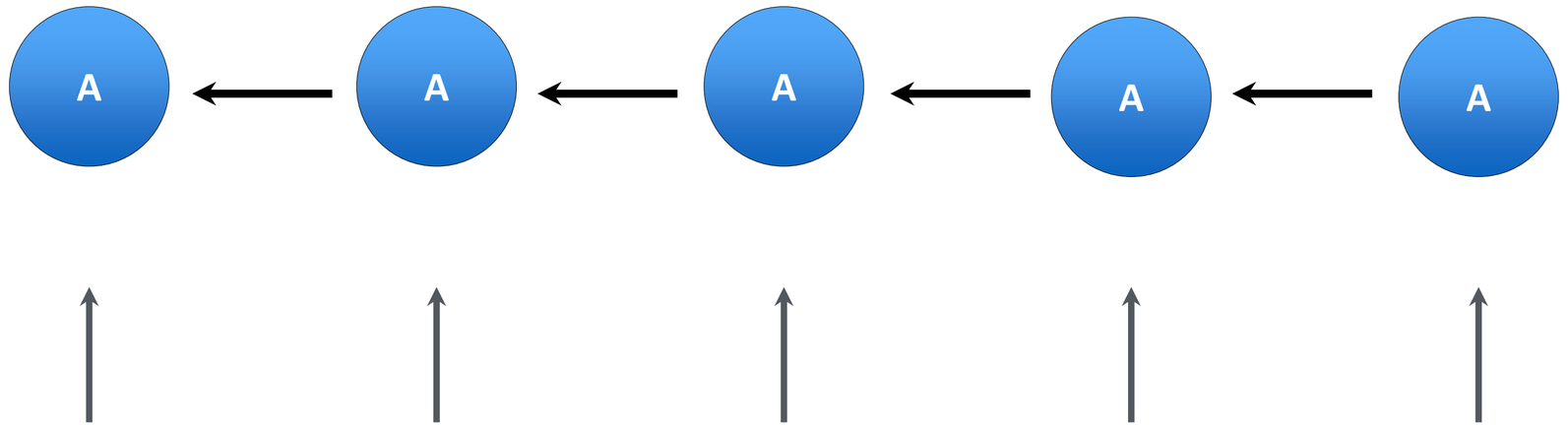


wat doet het weer morgen?


h_{t-2}
A
h_{t-1}
A
h_t
A
h_{t+1}
A
h_{t+2}
A
h_{t+3}
X_{t-1}
X_t
X_{t+1}
X_{t+2}
X_{t+3}

Backpropagation through Time

$$\delta = \hat{y} - y$$



wat doet het weer morgen?

δ δ δ δ δ

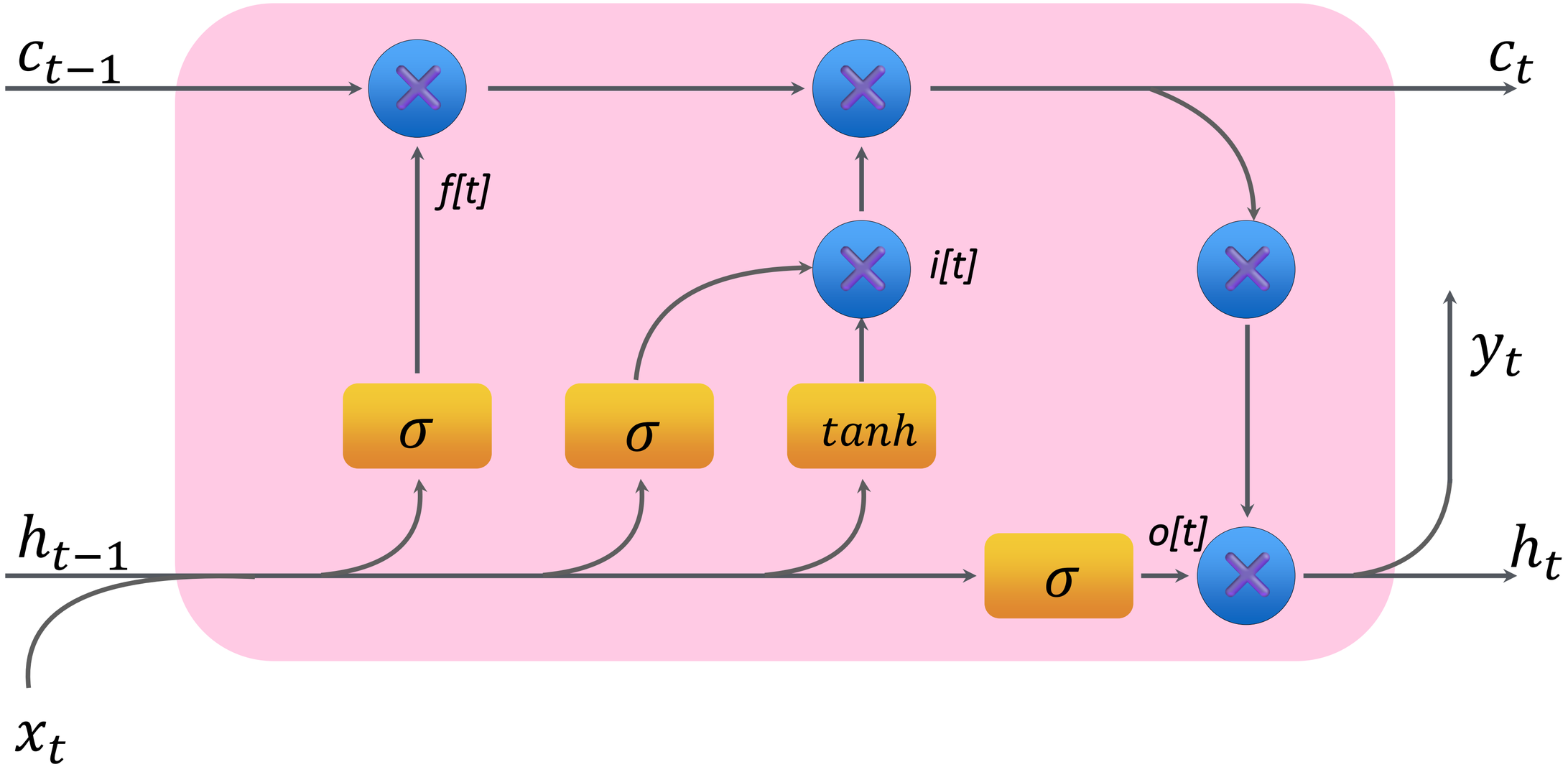
Vanishing Gradient

Long Short-Term Memory

Because recurrent neural networks (RNNs) were known to suffer from the **vanishing gradient problem**, the long short-term memory (LSTM) was an improvement over them. The improvement was the introduction of a *gating function* into the state dynamics of RNNs. LSTMs use a hidden cell state vector $\boldsymbol{c}$ to store long-term information.

Marinescu 2019, p.119 - aangepast

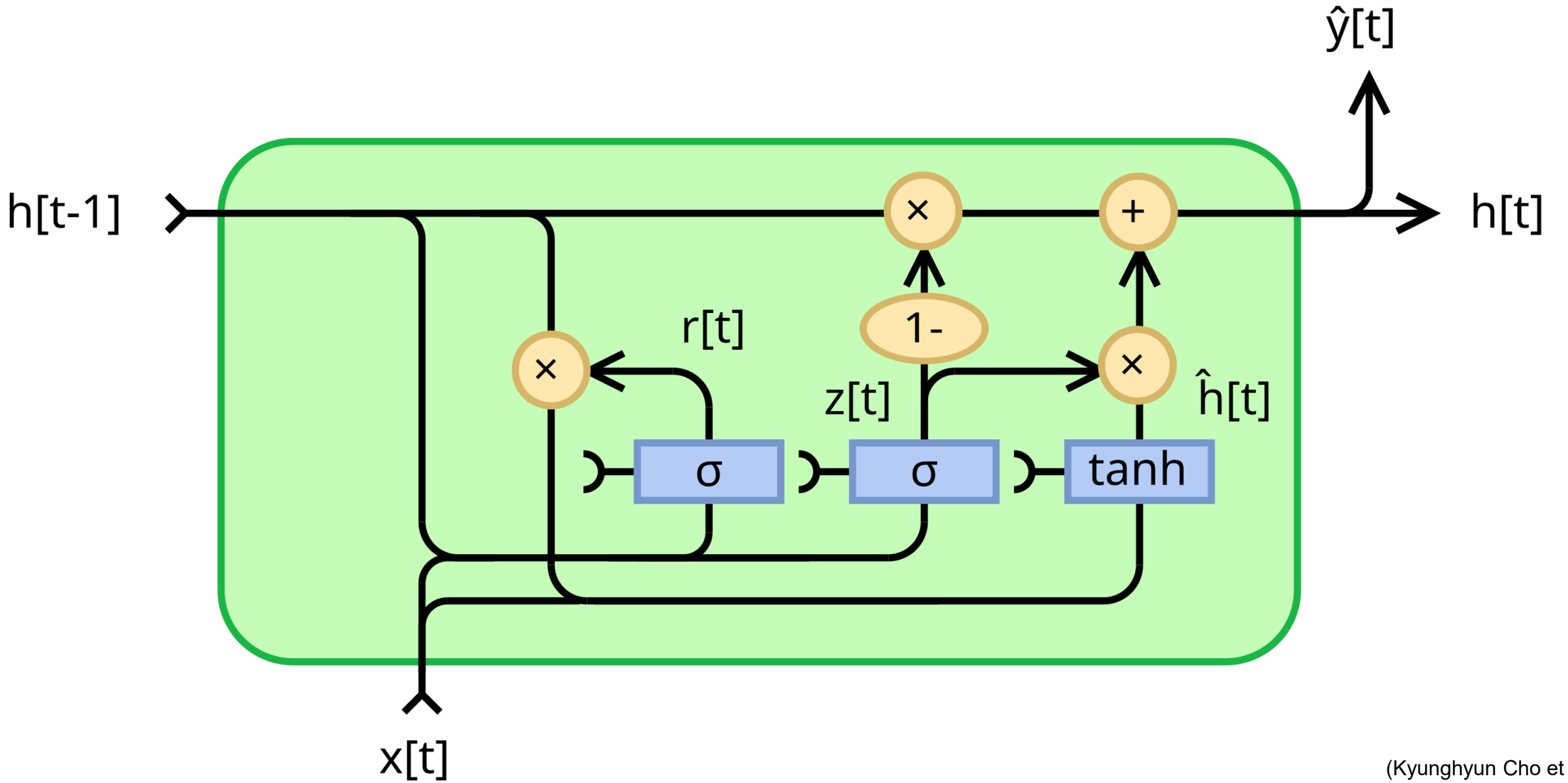
All this is achieved by means of three gates: a forget gate ***f[t]***, an input gate ***i[t]***, and an output gate ***o[t]***.



Marinescu 2019, p.119

Gated Recurrent Unit

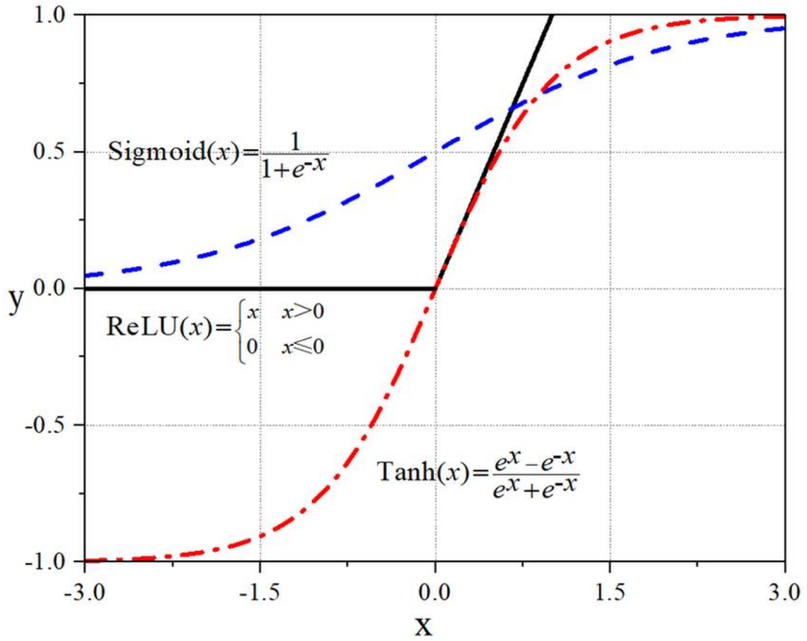
Minder complex dan LSTM. ***r[t]*** = update gate, ***z[t]*** = gate controller



(Kyunghyun Cho et al., 2019)

Nadelen van RNN's

- Exploding/vanishing (unstable) gradients
  - Mogelijke oplossingen: goede initialisatie, verzadigende activatiefuncties (σ, tanh), normalisatie, dropout
- Beperkt kortetermijn-geheugen
  - LSTM en GRU lossen dit deels op
- Trainen kost veel tijd
- Niet te parallelliseren
- Niet te *stacken*



Sigmoid, tanh en ReLU. Bron: researchgate.net

Daarom…

Afsluiting: live Notebook over tekstgeneratie met een RNN