Aleksander Molak

What should I buy next?

How to leverage word embeddings to build an efficient recommender system.

About me

Aleksander Molak

https://www.linkedin.com/in/aleksandermolak/

- Innovation Lead & Researcher at **Lingaro**
- **NLP**, sequence models, **causal modeling**
- Projects for Fortune Global 100 companies
- Complex systems, psychology, neuroscience
- Running, vegan food, languages

Overview

Recommender systems

- Embeddings – refresher
- Recommender systems – refresher
- Recommendations with embeddings
- How to tune your embeddings?

Word embeddings

Word embeddings

*An **embedding** is a relatively low-dimensional space into which you can translate high-dimensional vectors.*

https://developers.google.com/machine-learning/crash-course/embeddings/video-lecture

Word embeddings

*An **embedding** is a relatively low-dimensional space into which you can translate high-dimensional vectors.*

REPRESENTATION

https://developers.google.com/machine-learning/crash-course/embeddings/video-lecture

Types of embeddings

Word embeddings

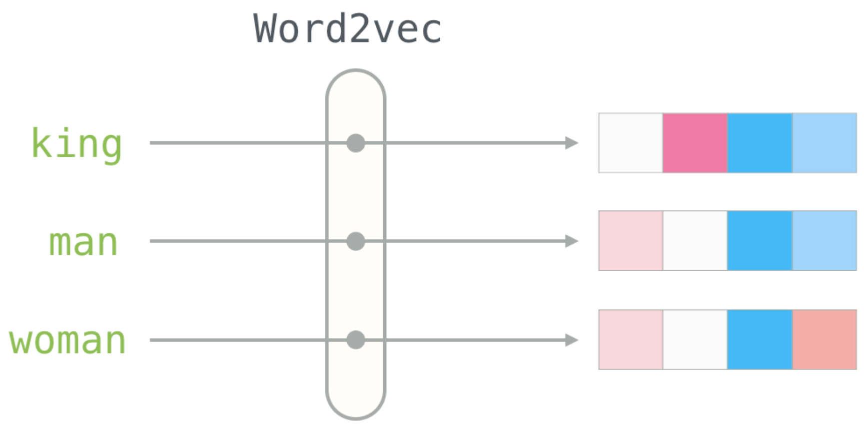
- **Static**

Word embeddings

- Static
- Dynamic

Word embeddings

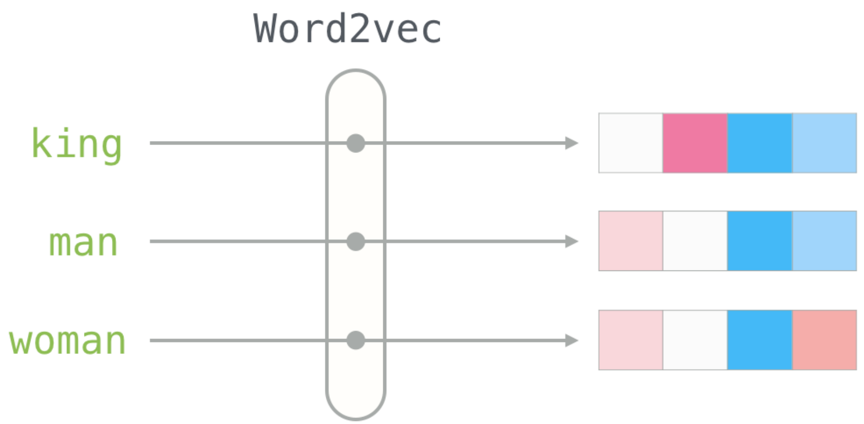
- **Static**
- **Dynamic**



Word2Vec – Jay Alammar – https://jalammar.github.io/illustrated-word2vec/

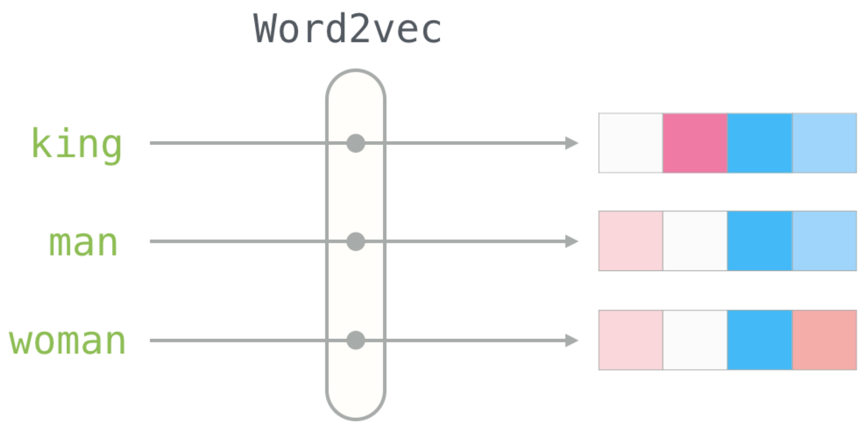
Word embeddings

- **Static**
- **Dynamic**



Word embeddings

- **Static**
- **Dynamic**

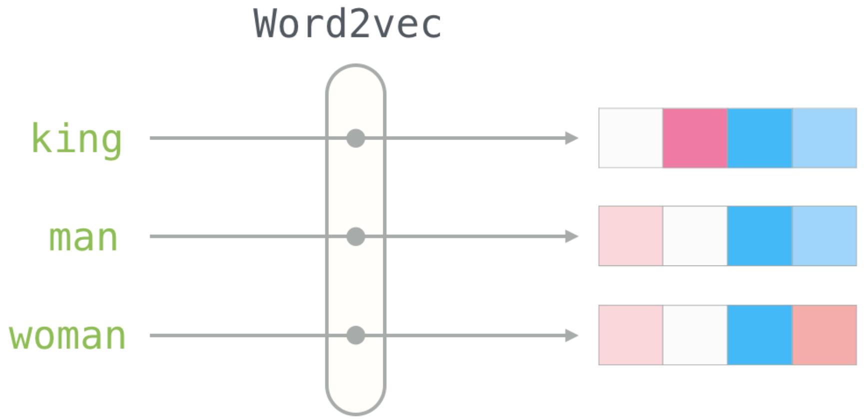


1. I went to the bank to withdraw some cash 💲
2. I spent a lovely day on the river bank! 🏞

Word2Vec – Jay Alammar – https://jalammar.github.io/illustrated-word2vec/

Word embeddings

- Static
- Dynamic



1. I went to the bank to withdraw some cash 💲
2. I spent a lovely day on the river bank!

Word2Vec – Jay Alammar – https://jalammar.github.io/illustrated-word2vec/

Word embeddings

- **Static – flavors**

Word embeddings

- **Static – flavors**
  - Word2Vec
  - FastText
  - Glove

Word embeddings

- **Static – flavors**
  - Word2Vec
  - FastText
  - Glove

Word embeddings

Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Word embeddings

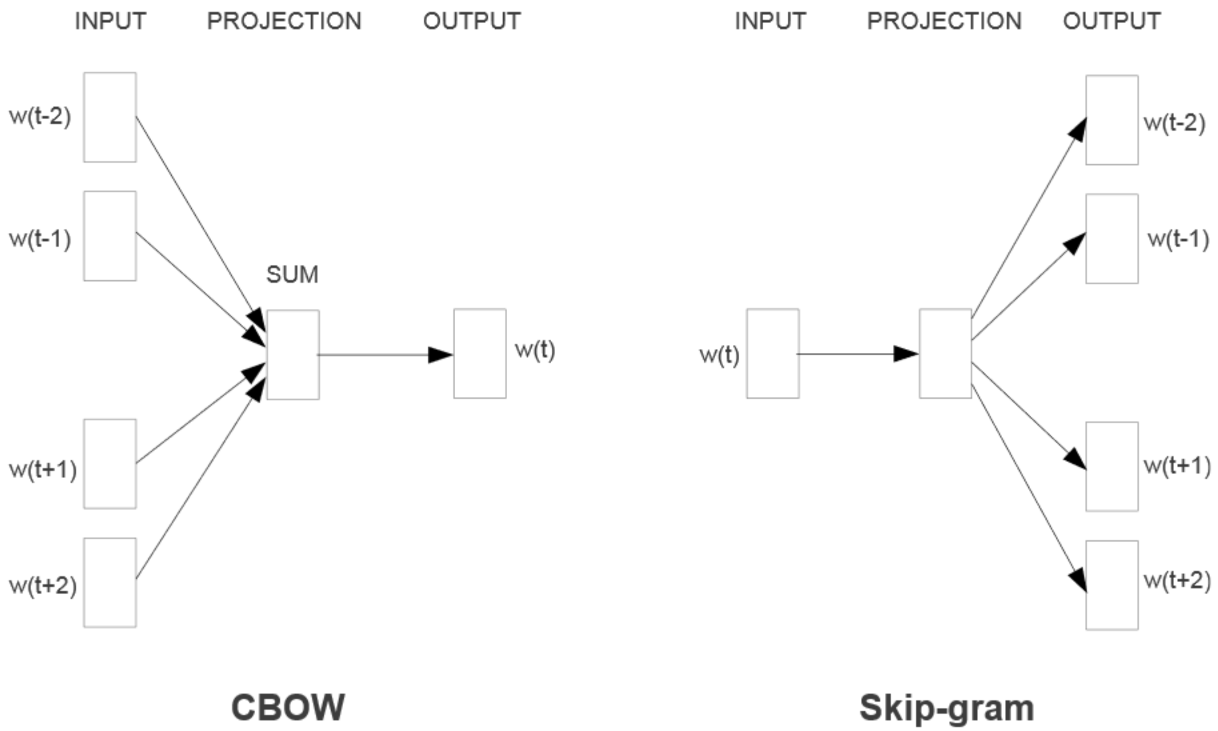


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

Word embeddings

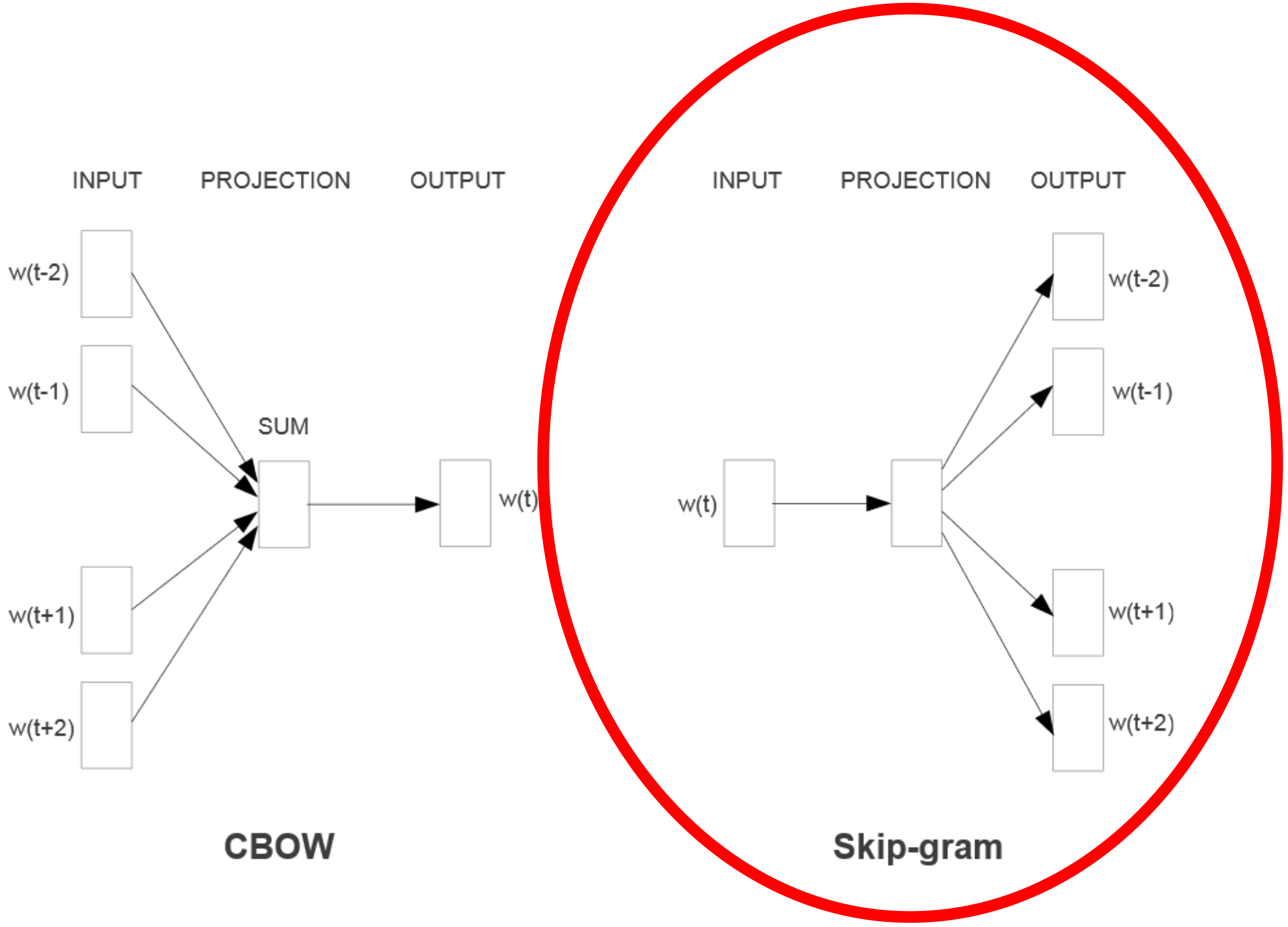
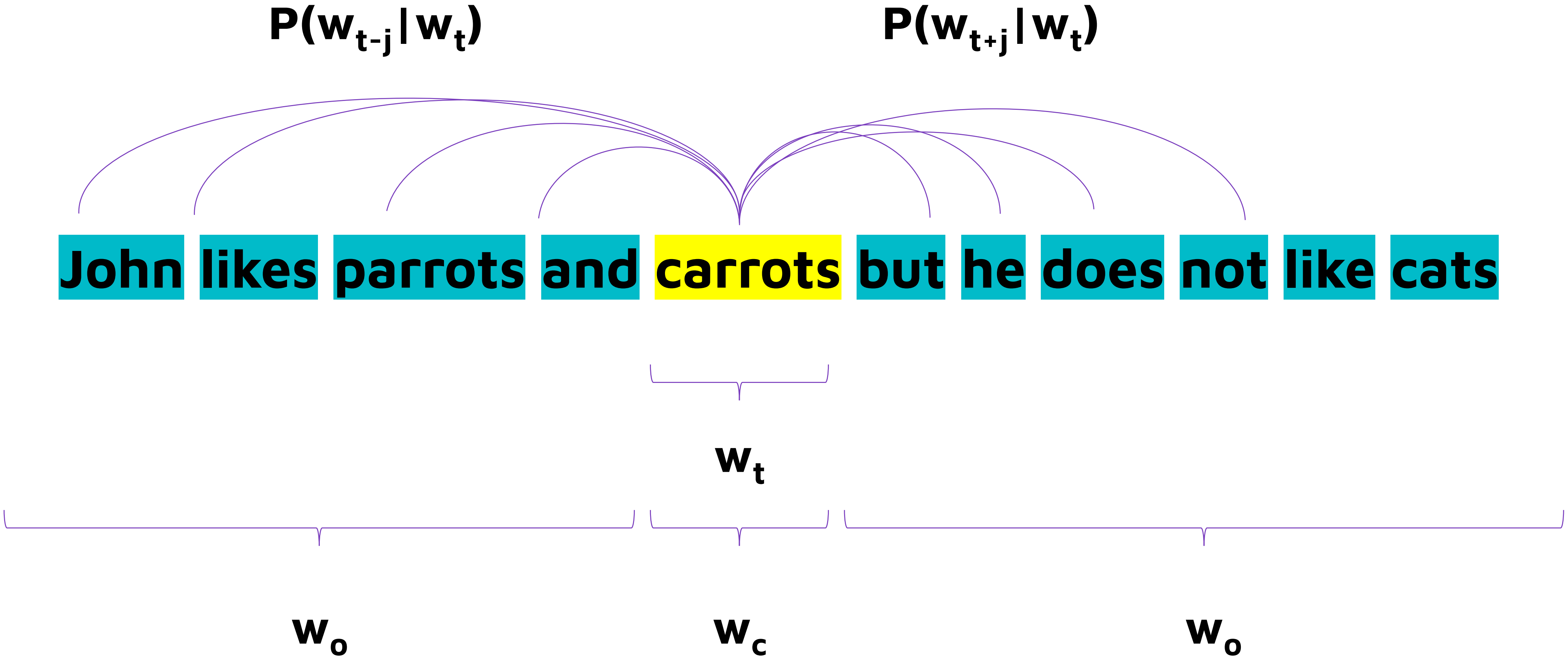


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

Word embeddings



Word embeddings

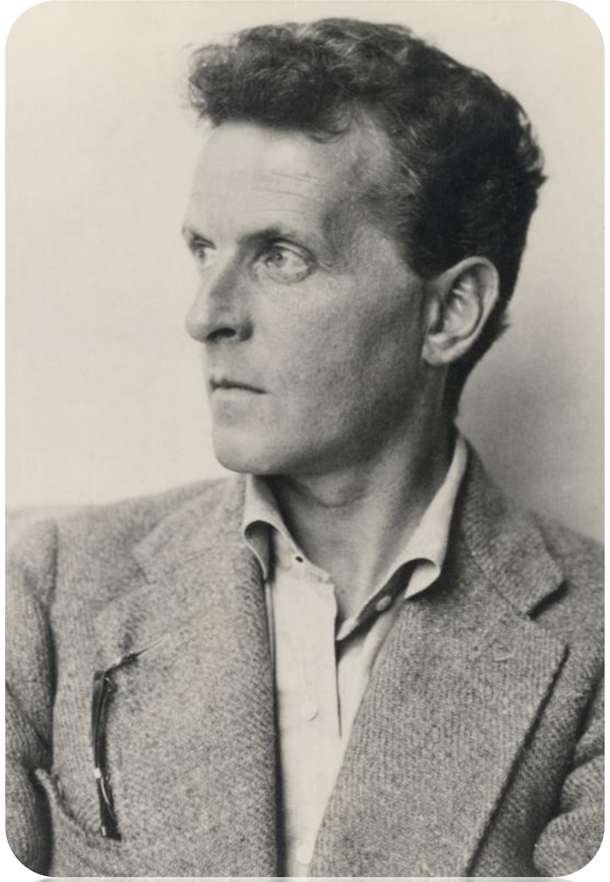
You shall know a word by the company it keeps.

John Rupert Firth

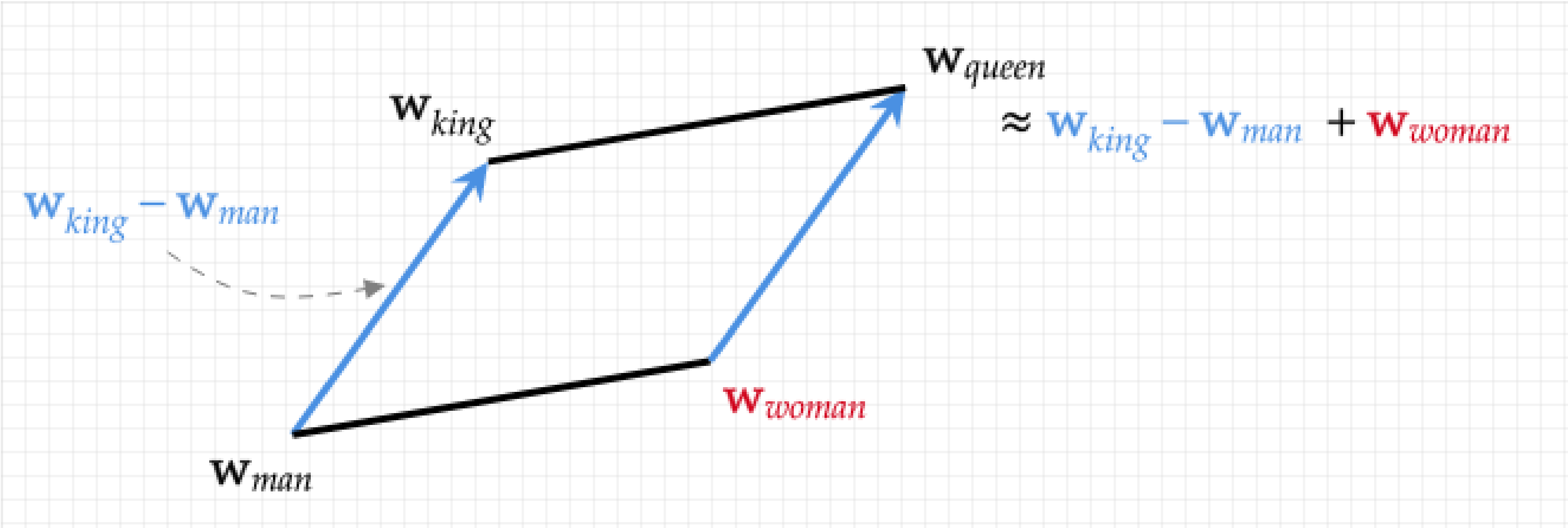
Word embeddings

For a large class of cases – though not for all – in which we employ the word "meaning" it can be defined thus: the meaning of a word is its use in the language.

Ludwig Wittgenstein



Word embeddings



Source: https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.ed.ac.uk%2Finformatics%2Fnews-events%2Fstories%2F2019%2Fking-man-woman-queen-the-hidden-algebraic-struct&psig=AOvVaw25ANF5cSKCjgQV6zk0rdf8&ust=1601706755751000&source=images&cd=vfe&ved=0CA0QjhxqFwoTCID6h_uklewCFQAAAAAdAAAAABAK

Recommender systems

Recommender systems

Recommender systems

- Collaborative filtering

Recommender systems

- Collaborative filtering
- Content-based

Recommender systems

- Collaborative filtering
- Content-based

Recommendations with embeddings

Recommendations with embeddings

- **Static embeddings – flavors**

Recommendations with embeddings

- **Static embeddings – flavors**
  - Word2Vec
  - FastText
  - Glove

Recommendations with embeddings

- **Static embeddings – flavors**
  - Word2Vec
  - FastText
  - Glove

Recommendations with embeddings

- **Word2Vec – training regimes**
  - Word2Vec
  - MetaProd2Vec

Recommendations with embeddings

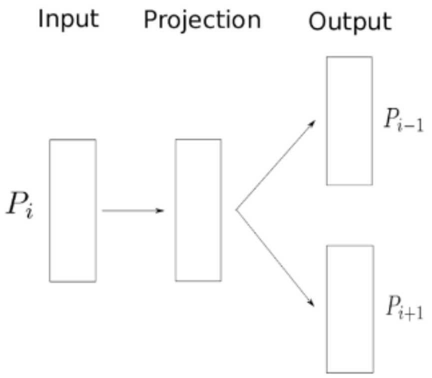


Figure 1: Prod2Vec Neural Net Architecture.

Recommendations with embeddings

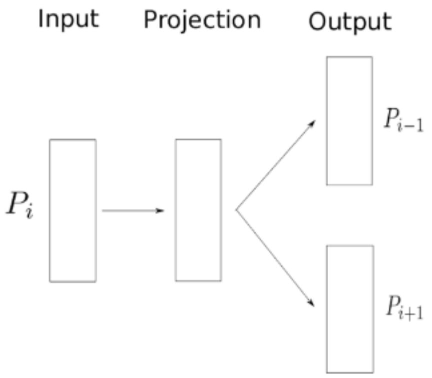


Figure 1: Prod2Vec Neural Net Architecture.

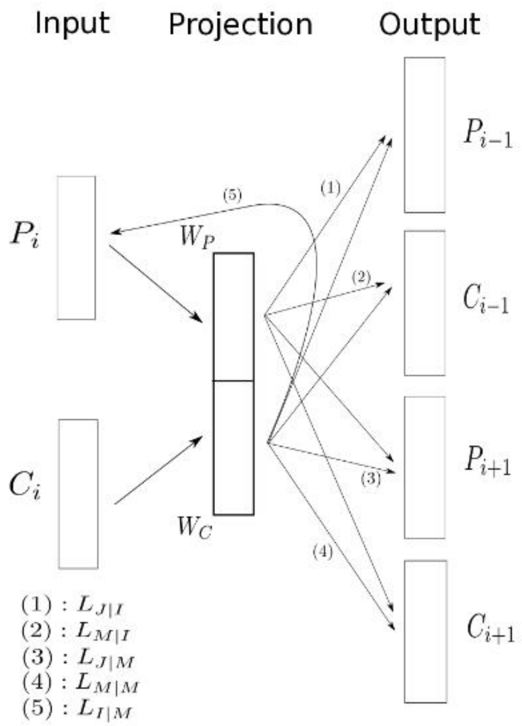


Figure 2: Meta-Prod2Vec Neural Net Architecture.

Implementation

Recommendations with embeddings

```
from gensim.models import Word2Vec
from gensim.models import KeyedVectors
```

Recommendations with embeddings

```
# Train the model
model = Word2Vec(train_data, **model_params, callbacks=callbacks)
```

Word2Vec vs Meta-Prod2Vec

Hyperparameters

Recommendations with embeddings

Sub-sampling

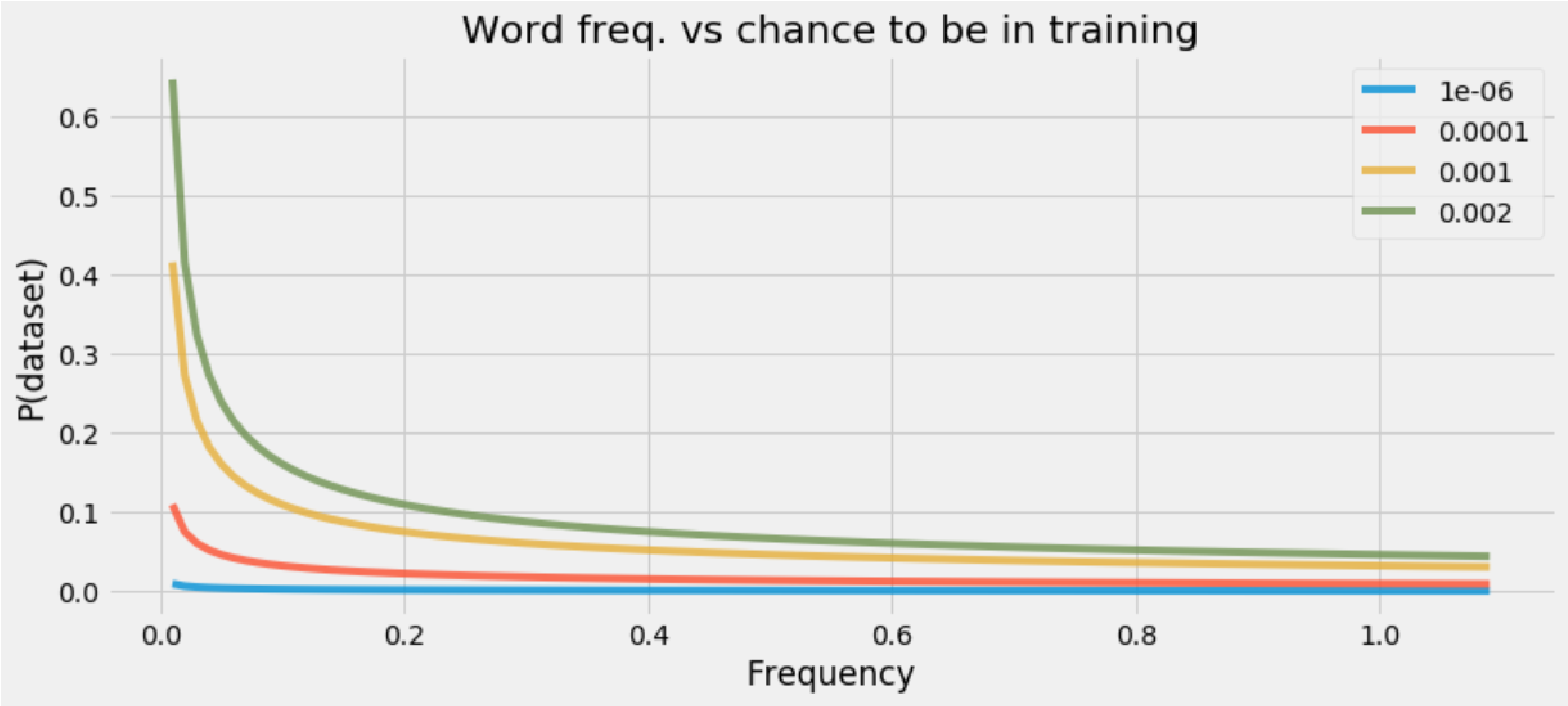
Sub-sampling reduces probability of using the most frequent (or the least frequent) words in the training.

Note that the formula comes from Google implementation of W2V and it's slightly different from the one in he paper.

$$P(w_j) = \left(\sqrt{\frac{z(w_j)}{k}} + 1\right)\frac{k}{z(w_j)}$$

** Default value for $k = .001$

Recommendations with embeddings



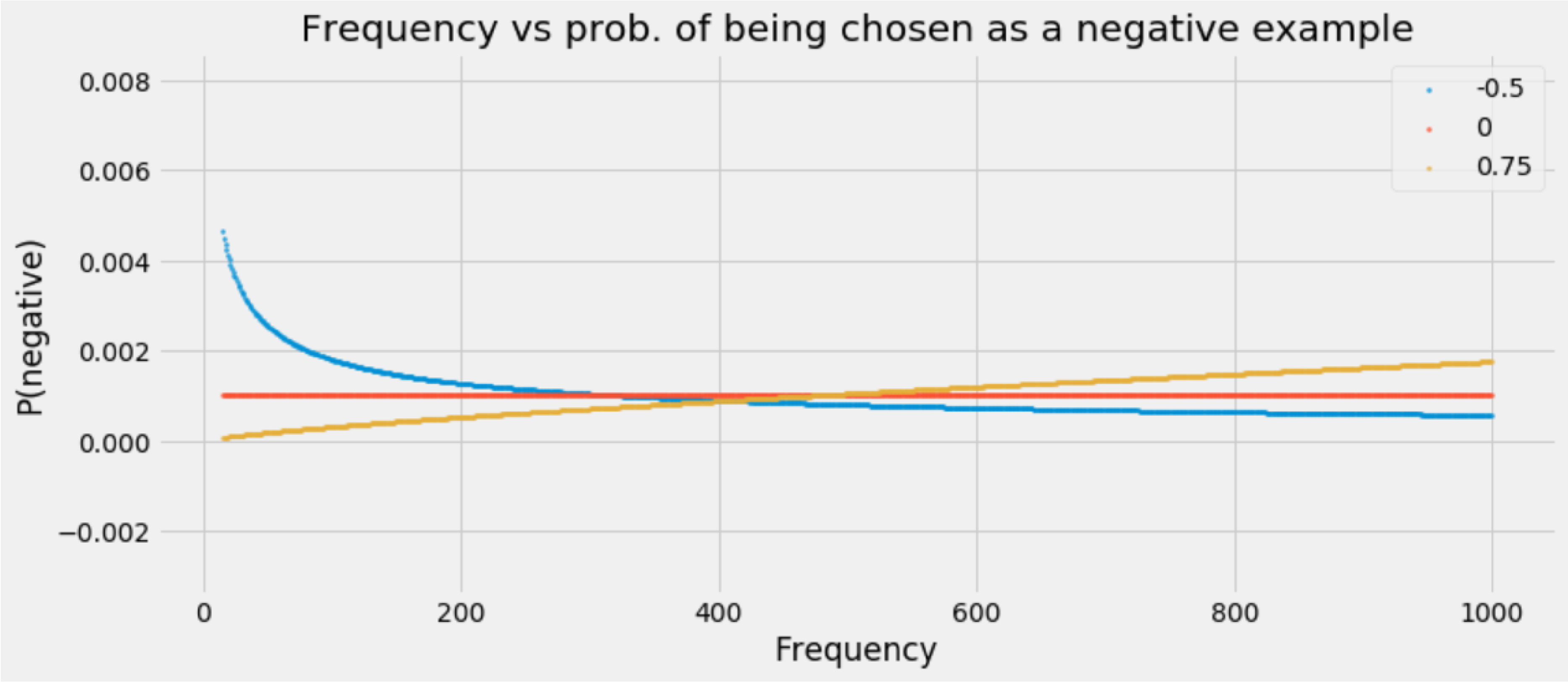
Recommendations with embeddings

Negative sampling

Instead of all negative examples (all words not present in the context), we will just update weights for a couple of randomly selected words.

$$P(w_i) = \frac{f(w_i)^k}{\sum_{j=0}^{N} \left(f(w_j)^k\right)}$$

Recommendations with embeddings



Recommendations with embeddings

Word2vec applied to Recommendation: Hyperparameters Matter

Hugo Caselles-Dupré[1,2*]
[1] Flowers Laboratory (ENSTA ParisTech & INRIA)
[2] Softbank Robotics Europe
Paris, France
caselles@ensta.fr

Florian Lesaint
Deezer SA
Paris, France
flesaint@deezer.com

Jimena Royo-Letelier
Deezer SA
Paris, France
jroyo@deezer.com

Recommendations with embeddings

```
# Initialize callbacks
callbacks = [EpochLogger()]

# Set model params
EMB_DIM = 300
N_CORES = 4
WINDOW = 9   # For traditional w2v you might want to try the 90th percentile of len(rows)
N_EPOCHS = 200
NEG_EXP = -.5
MIN_COUNT = 5
SAMPLE = 0

model_params = dict(size = EMB_DIM,
                    sg = 1,
                    negative = 5,
                    iter = N_EPOCHS,
                    ns_exponent = NEG_EXP,
                    sample = SAMPLE,
                    workers = N_CORES,
                    window = WINDOW,
                    min_count = MIN_COUNT)
```

```
# Train the model
model = Word2Vec(train_data, **model_params, callbacks=callbacks)
```

Pros and cons

Pros 🥰

- **Easy to maintain**
- **Supports continuous training**
- **Fast to train**
- **Ultra fast retrieval** (e.g. with HNSW*)
- **Can be easily served in a serverless fashion** (depending on the no. of tokens)

* Paper: https://arxiv.org/ftp/arxiv/papers/1603/1603.09320.pdf
C++ / Python implementation: https://github.com/nmslib/hnswlib

Cons 😥

- Not suitable for cold-start scenarios
- Might be noisy
- Not easily interpretable

Summary

Summary

- Embedding-based systems are fast and easily maintainable
- Word2Vec and Meta-Prod2Vec cover a broad spectrum of cases
- HNSW gives you an amazing recall / speed ratio
- It’s worth to pay attention to hyperparams if you want good results

Thank you!

Thank you!

Aleksander Molak

LinkedIn

https://www.linkedin.com/in/aleksandermolak/

Twitter

@AleksanderMolak