Task

Agenda: Words → d-dim Vectors

(capture Semantics)

Prev: BoW, TFIDF

e.g: good; great; best

Plan: Let us build/re-invent techniques together (by building on ideas/concepts we already know)

Case:

With over 50,000 publications and research papers on the corona-virus family till date, it has become difficult to search across and get useful insights for medical practitioners.

As a *DataScientist* at Google, you are tasked with solving this problem with the help of Machine Learning.

[Efficient Estimation of Word Representations in Vector Space]

[GloVe: Global Vectors for Word Representation]

How can we solve for this problem?

- Can we match keywords from user queries that are present in abstract?
- If we do keyword matching, will we be able to understand the user's intent? For e.g: 'origin' and 'discovery'
- Should we consider the context of the words, then?

Let us build a search engine using Word Embeddings

Datset

Dataset Corpus

Dataset

$d_1 \rightarrow w_3\ w_6\ w_5\ w_3\ \cdots$

$d_2 \rightarrow w_1\ w_{12}\ w_6\ w_3\ \cdots$

$\vdots$

$d_n \rightarrow \cdots$

$w_i \rightarrow$ d-dim rep (semantic)

Semantically sensible

V_1 good

V_2 great

V_1 & V_2 → 2 similar

Cosine-sim $(V_1, V_2) \rightarrow 1$

good , terrible → V_3

cos-sim (V_1, V_3) : V. low

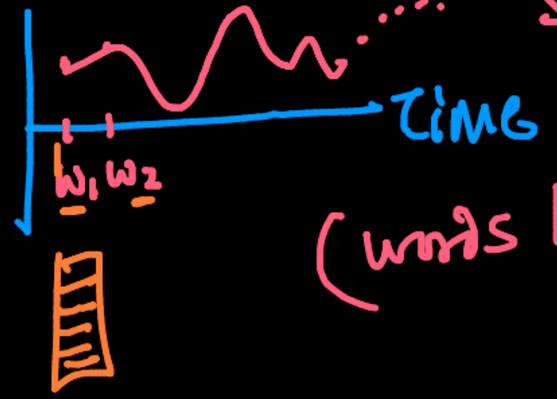
Task:

$w_i \rightarrow v_i \in \mathbb{R}^d$ using corpus

Brain storm:

1. Clustering (?) → Whose use the vectors to cluster?

2. Time-Series prediction → Classical ... RNN / LSTM / GRUs



(words to scalars?)

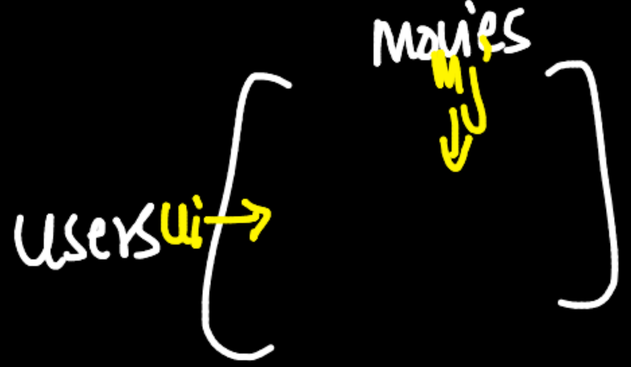
3. PCA → [data matrix ?]

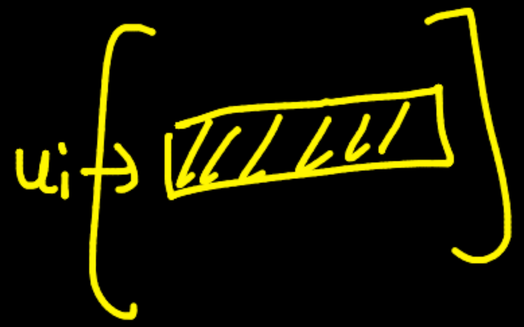
HINT: Combine 1 & 3 → MF ✓

MF Recap:

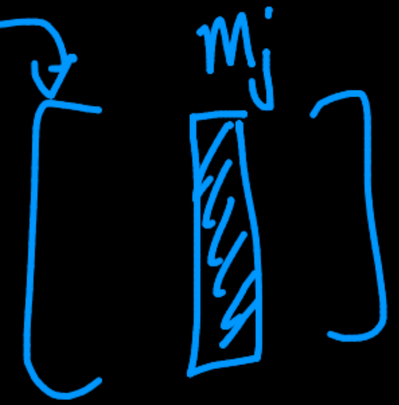
Data Matrix

$A_{n \times m} = B_{n \times d} \cdot C_{d \times m}$



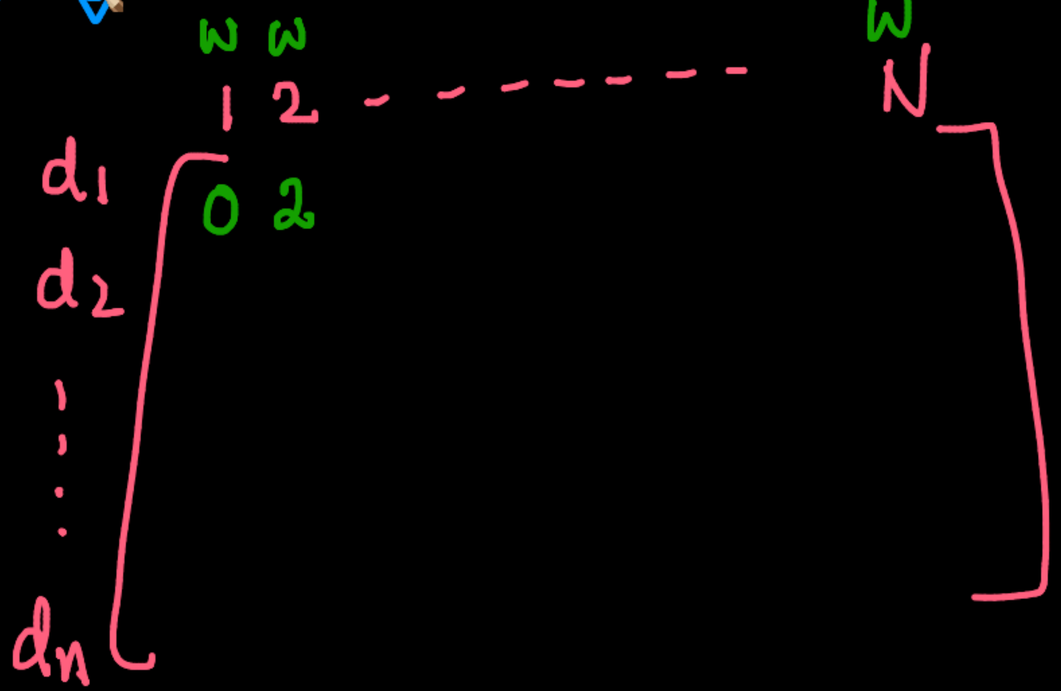


$n \times d$



Q: what is my data matrix in NLP task

doc-term occusance



Corpus : n-docs

Voc. N

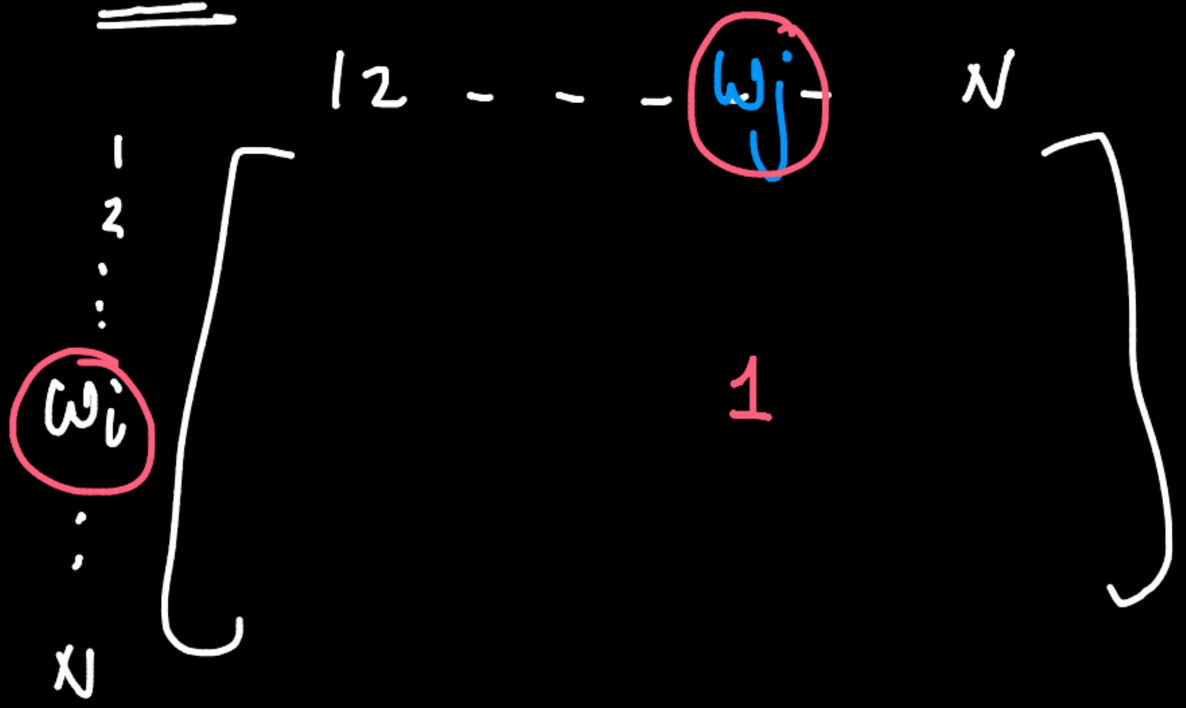
→ Problem? → very Sparse

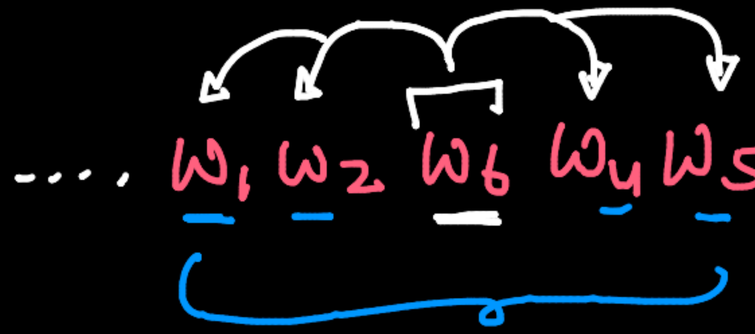
→ N is large

→ Sequence is ignored

Data Matrix:

Co-occurence - Matrix





window of size 5
(context)

↓

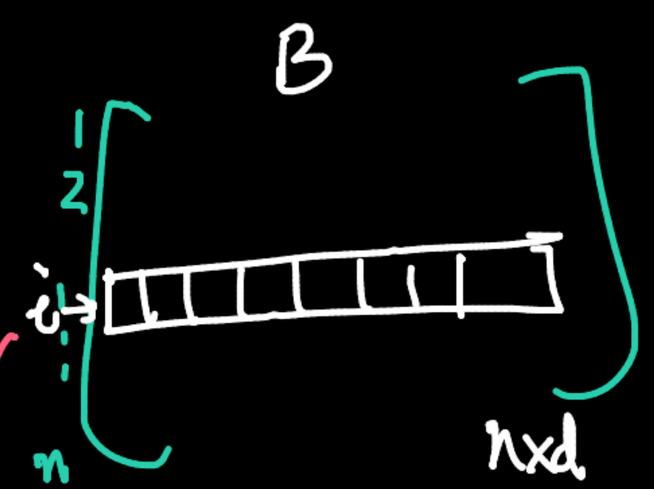
partially taking
Case of the
~~seq~~ context
info ...

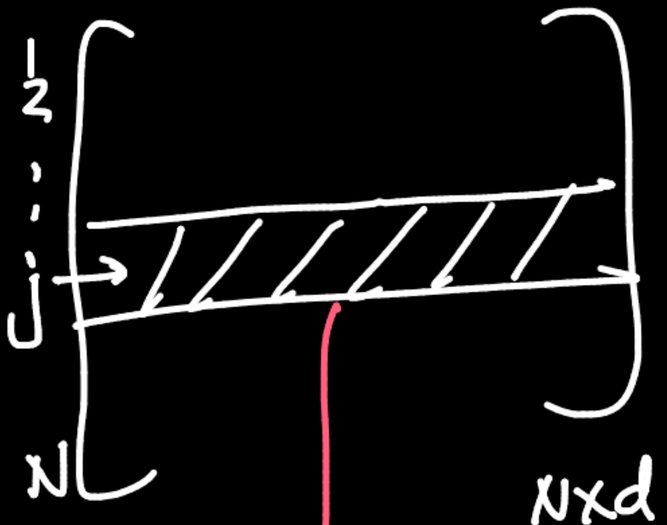
$\Delta W_{n \times N}$

V. Sparse ⟶ MF

$C_{N \times N}$

$$\tilde{A}_{n \times N} = B_{n \times d} \cdot C^{T}_{d \times N}$$





loss :

$$\sum_{\substack{i,j \\ A_{ij} \neq 0}} \left(A_{ij} - B_i^T C_j \right)^2 + \cdots$$

vec. rep of d_i

vec. rep of w_j

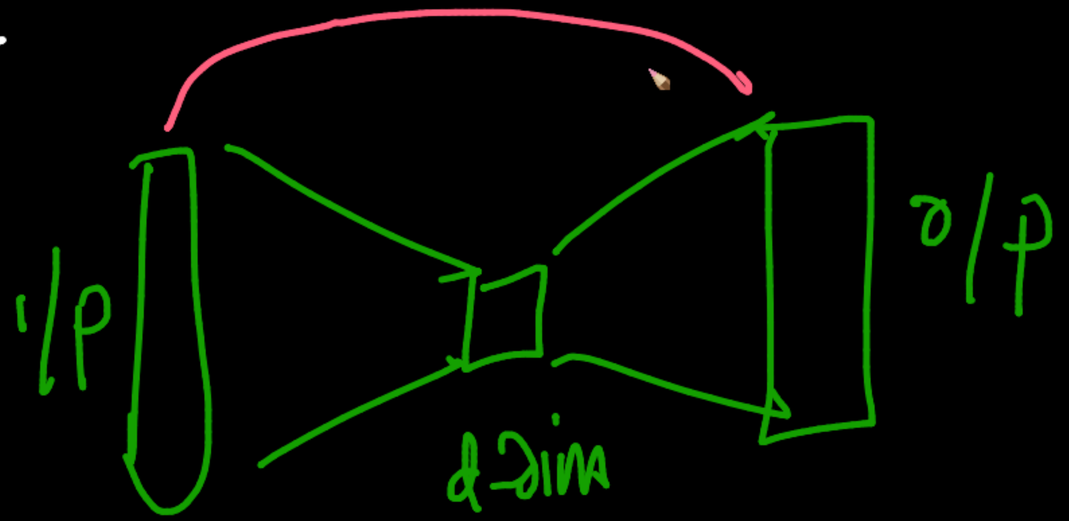
MF /SUD/ ----

Q

MF → Classical ML

? → deep-learning (NN)

✓ Auto-enc



Word → Vec

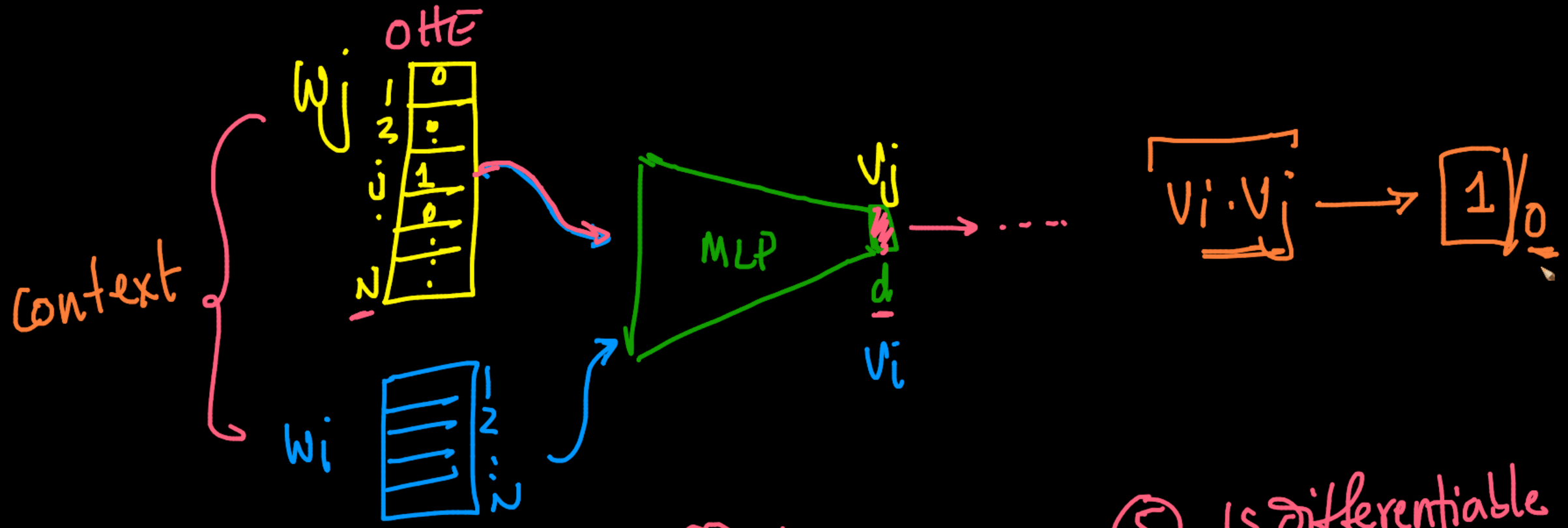
Context → doc-word occurance

→ word-word occurence in a context/window

data

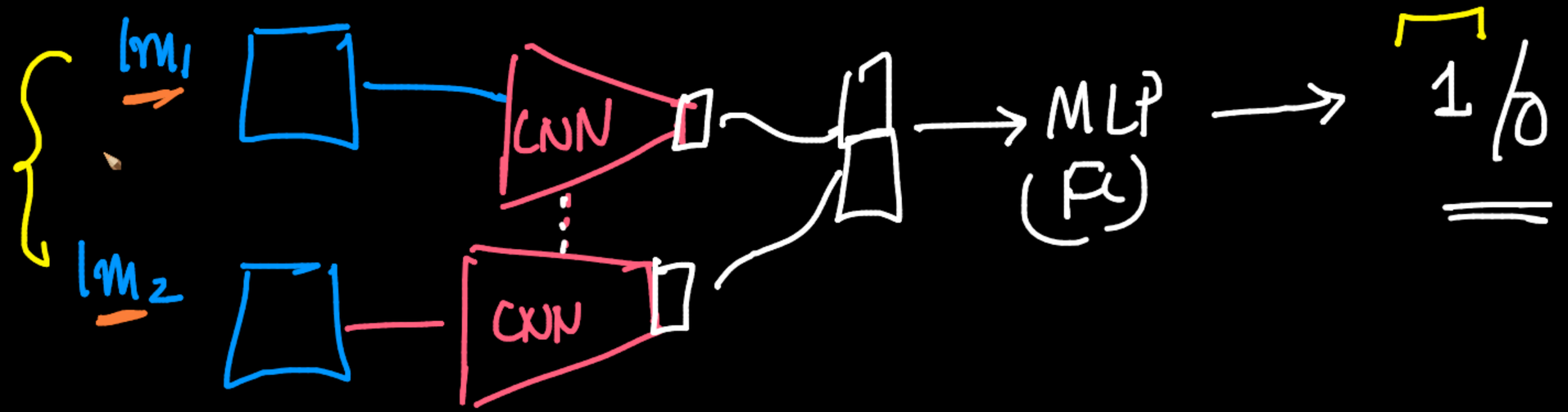
w_j → N-dim OHE vector

Voc Size : N

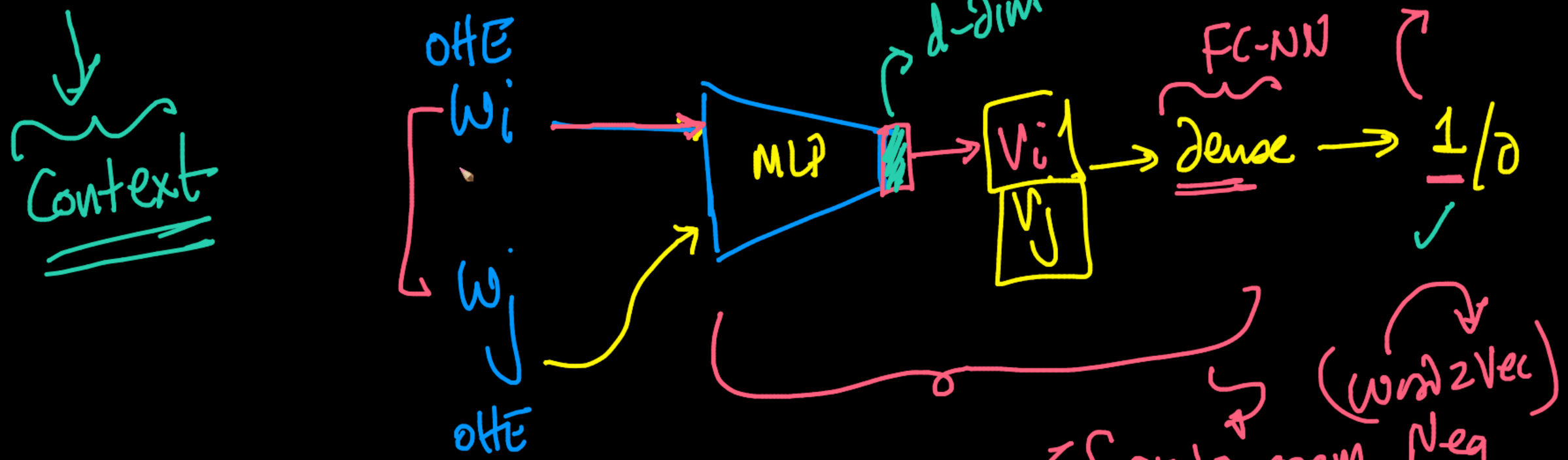


1. input
2. output
3. Loss
4. Architecture
5. is differentiable

Siamese n/w:



Corpus → w_i-w_j cooccurrence matrix (1/0)

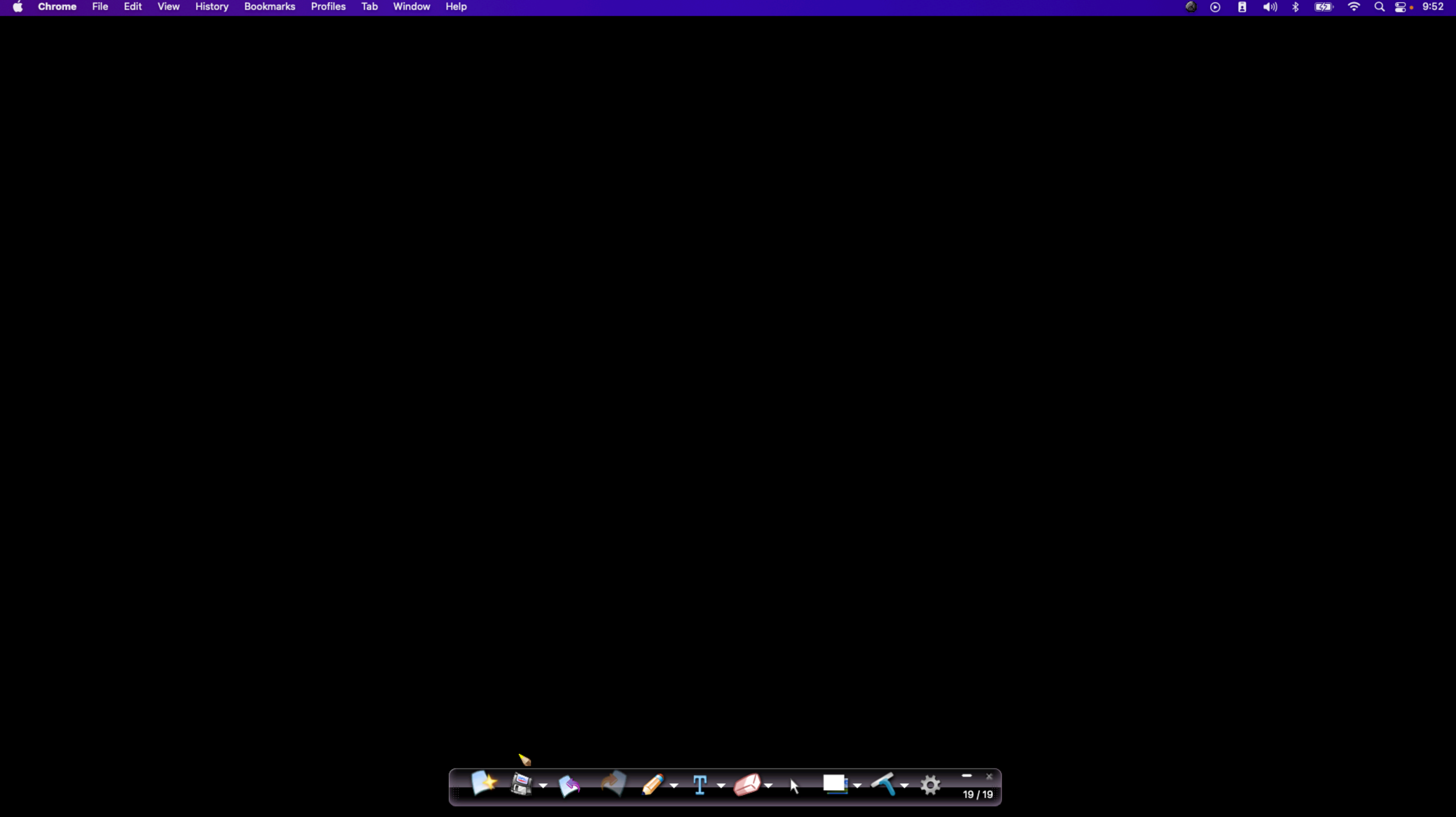


History:

(2012-13)

Word2Vec

- CBOW ✓
- Skip gram ✓
- Skip gram + Negative sampling ✓


Chrome File Edit View History Bookmarks Profiles Tab Window Help
9:52
19 / 19

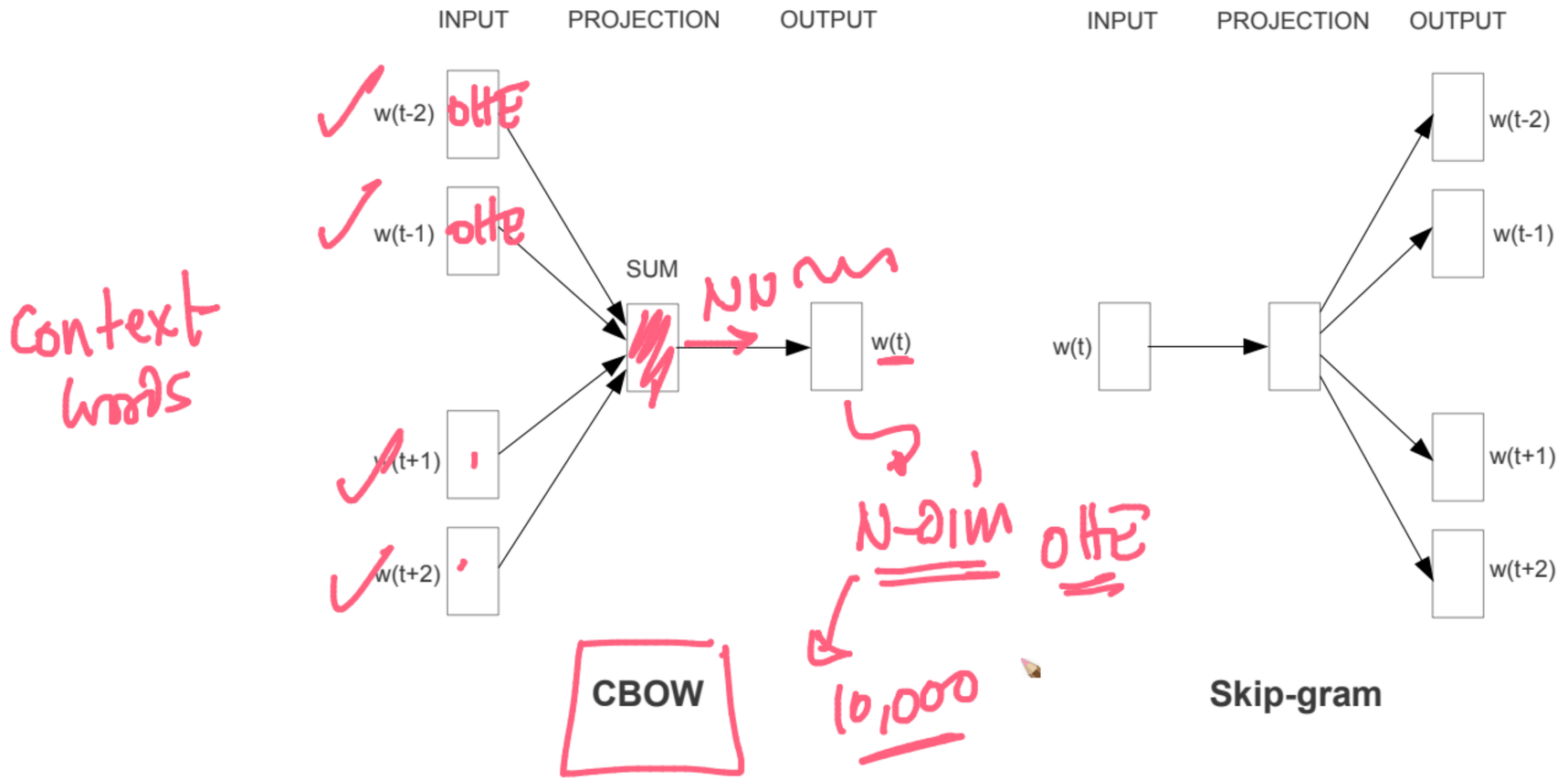


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

R words from ... as to do $R \times 2$

- Finally, the network parameters **W_1 and W_2 are adjusted** using gradient descent.

Is CBoW used in practice?

- When the context window > 1, CBoW **overfits on frequent words** as the input is an average of one-hot encoded vectors of the context word.
- CBoW does **not produce good representations** for **rare words** in the corpus.

We can over-come these dis-advantages of CBoW by using an alternative architecture of Word2Vec called the Skip-Gram.

What is Skip-Gram?

- Skip-Gram is simply an **inversion of the CBoW** architecture.
- Takes in **input** as **center/context word** and **predict** its **surrounding words**.

Architecture of Skip-Gram

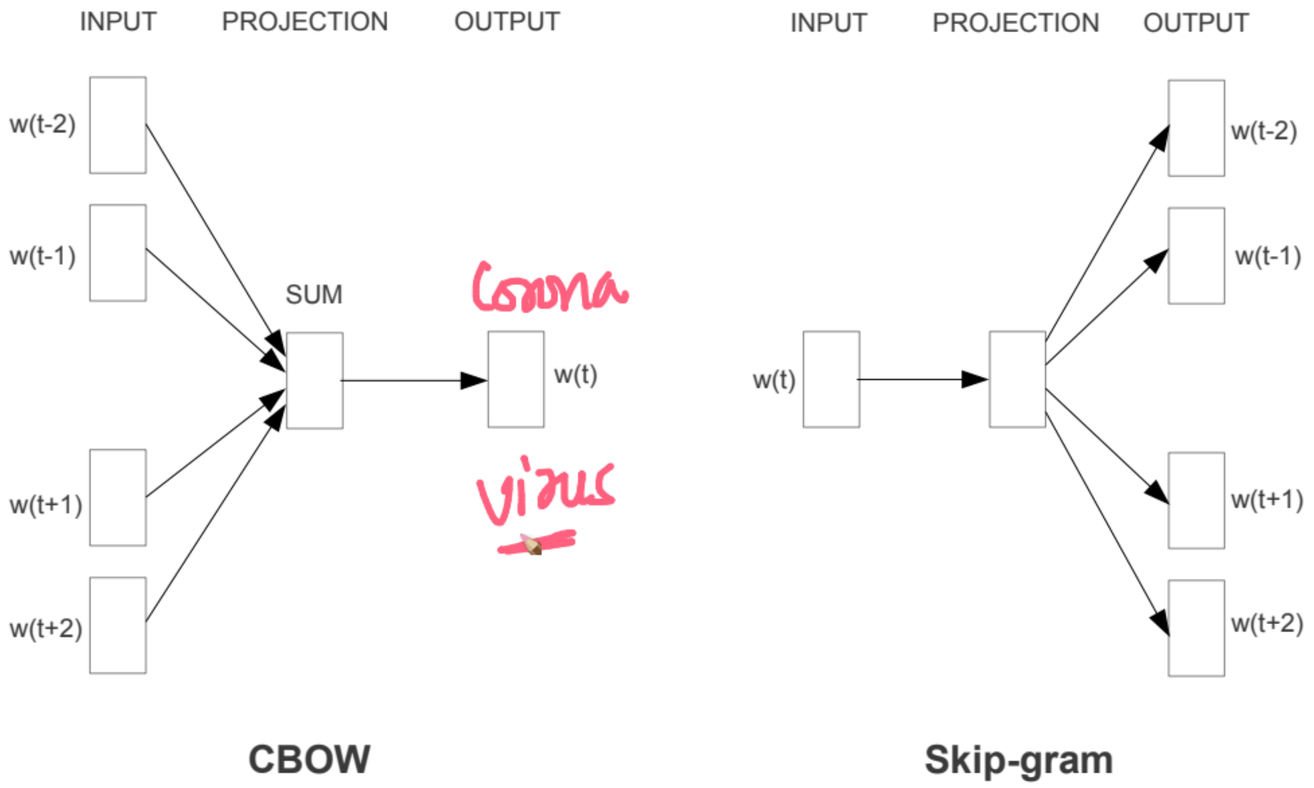


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

R words from t[illegible] as to do $R \times 2$

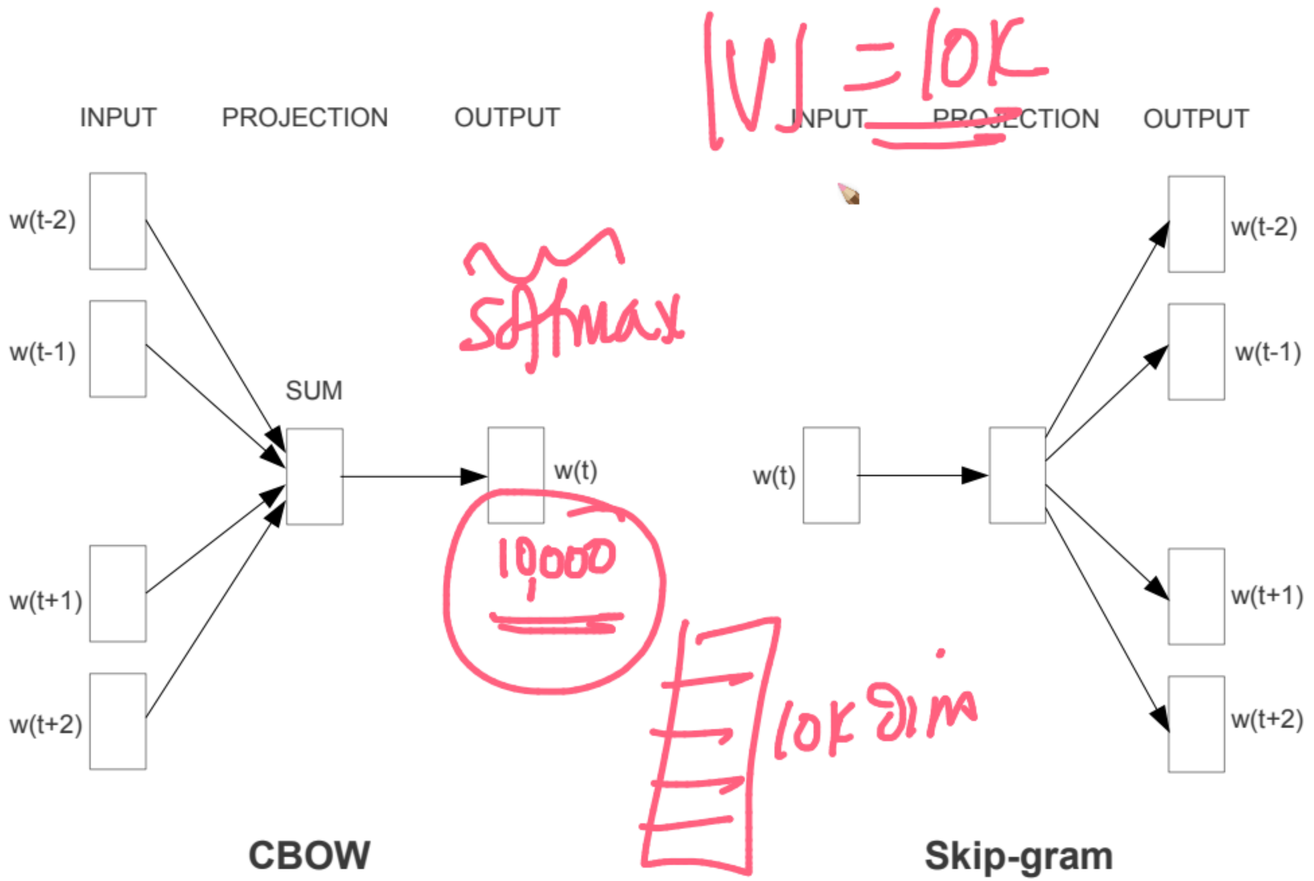
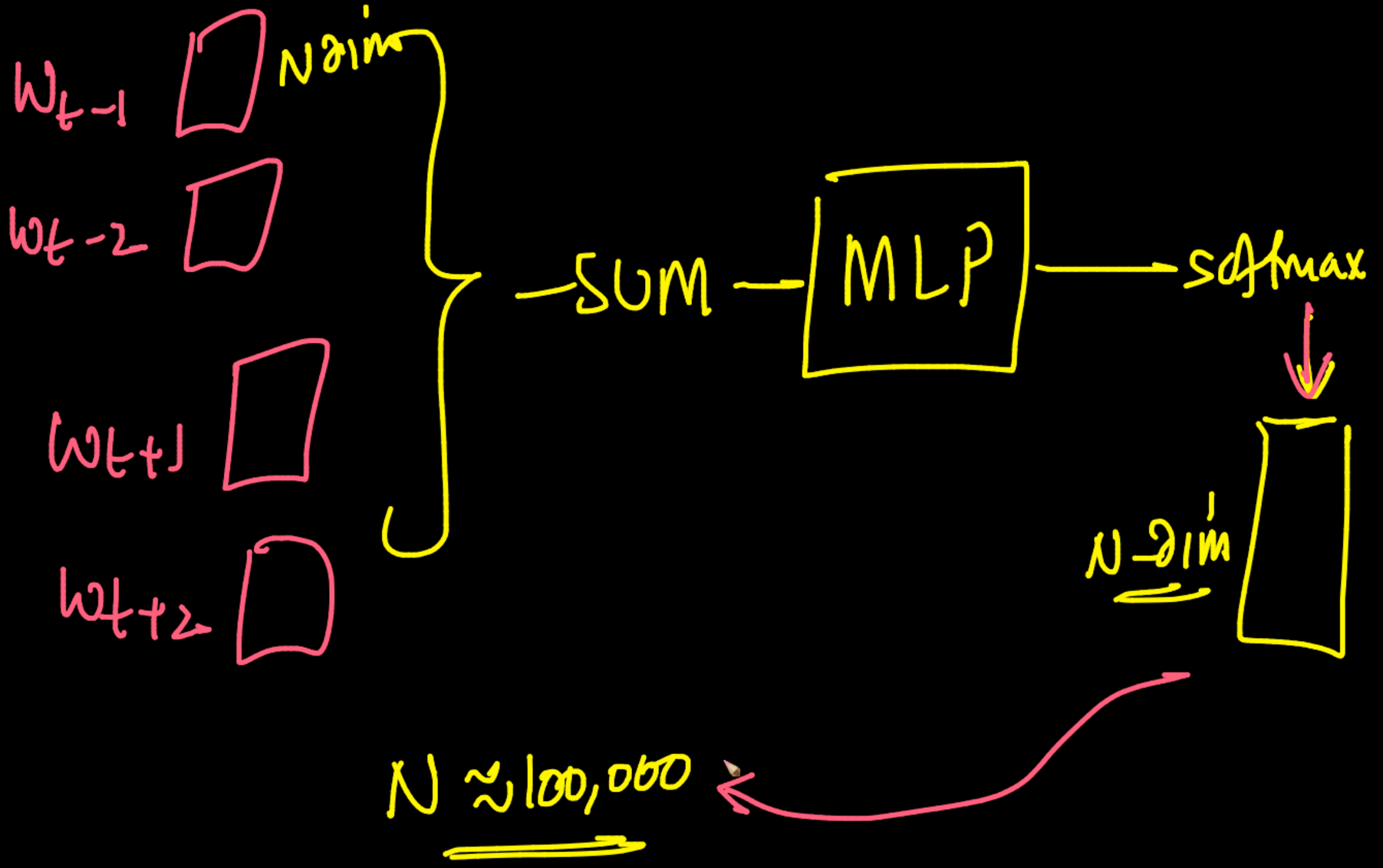
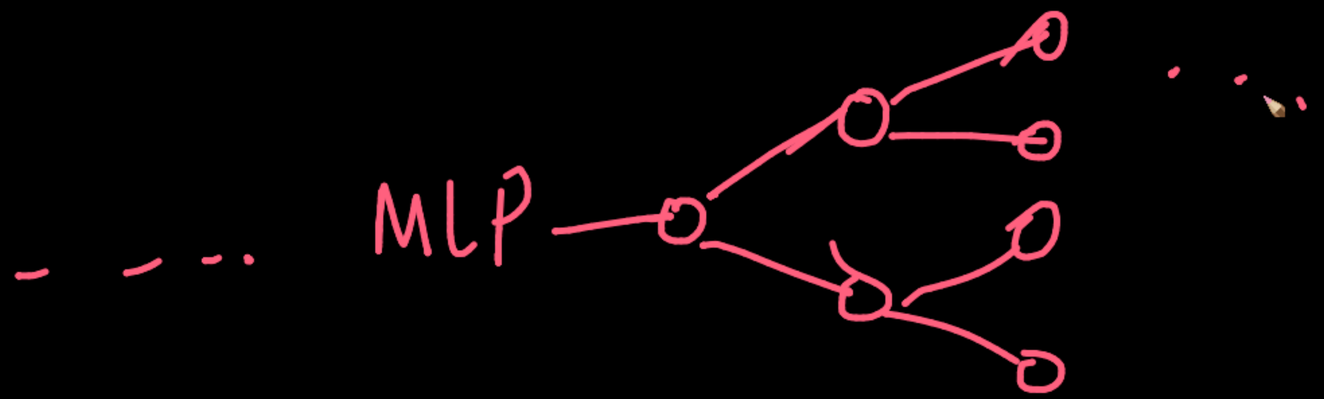


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

R words from t[illegible] as to do $R \times 2$





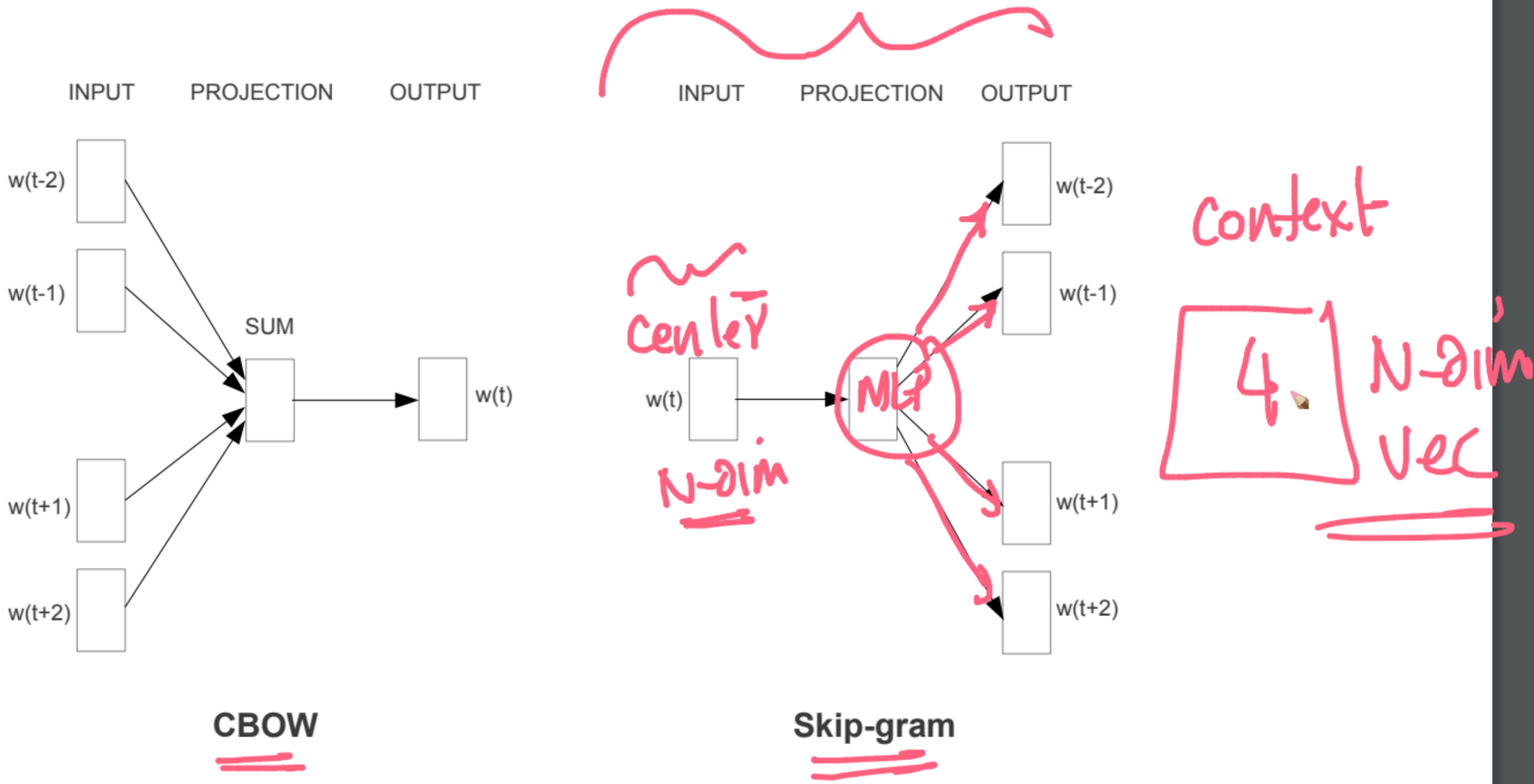


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

R words from t... as to do $R \times 2$

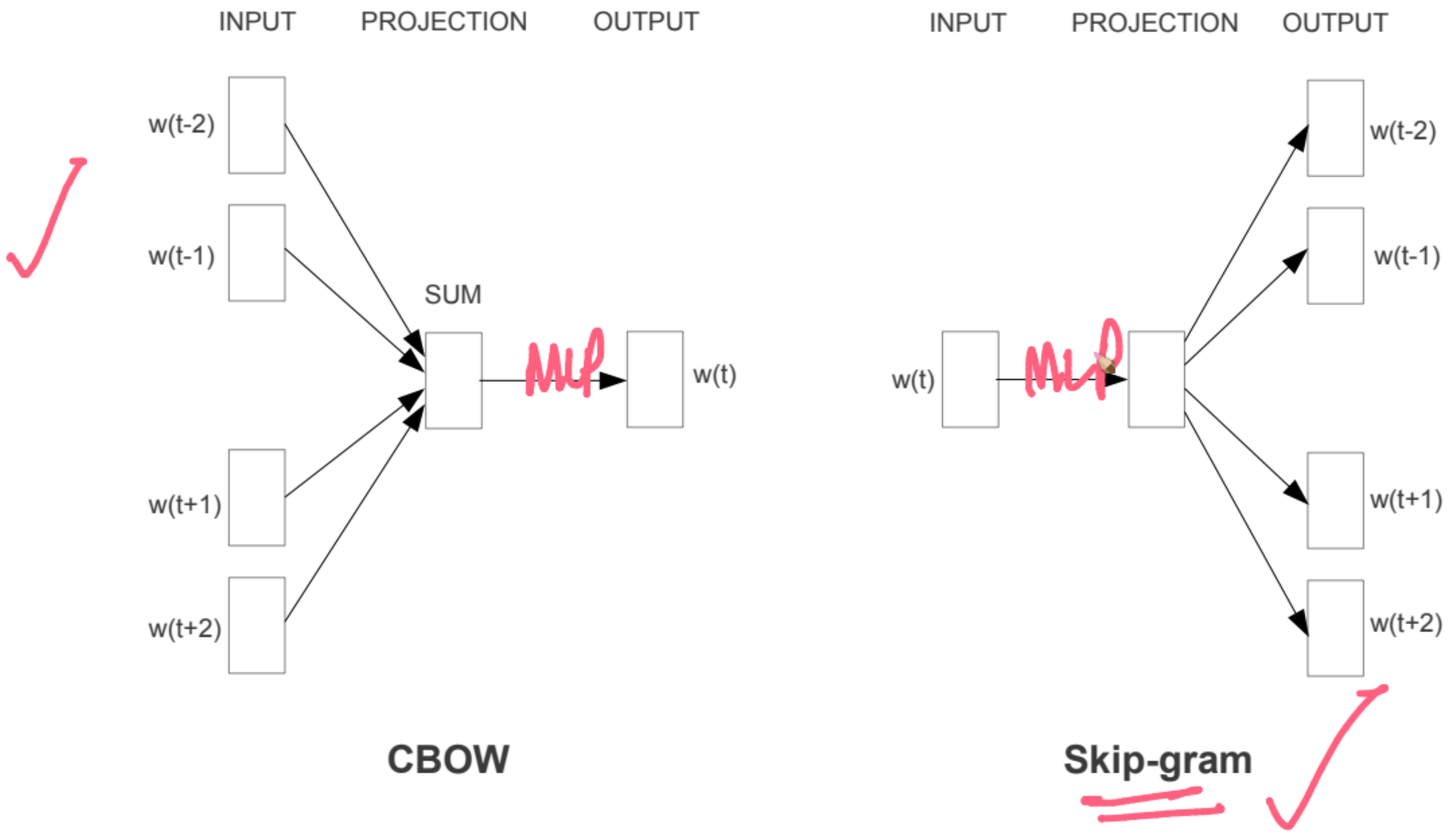


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

R words from t[illegible] as to do $R \times 2$

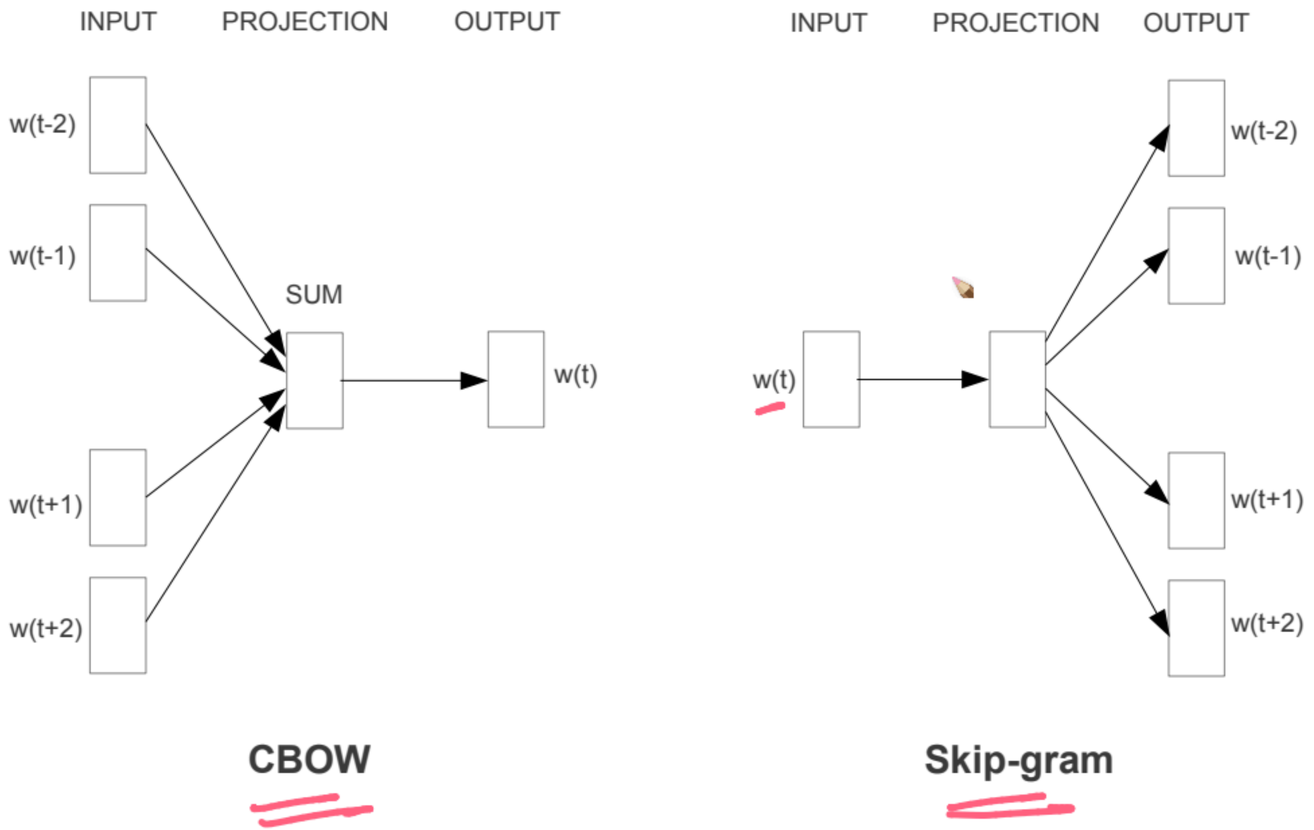


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

R words from t[illegible]s to do $R \times 2$

- Given the objective function, the model parameters **W and W' are fine-tuned** using gradient descent via backpropogation.
- Similar to CBoW, the **weight matrix** W_1 is considered as the **word vector**.

Negative Sampling for Skipgram training:

Refer: https://jalammar.github.io/illustrated-word2vec/

Skipgram

shalt	not	make	a	machine

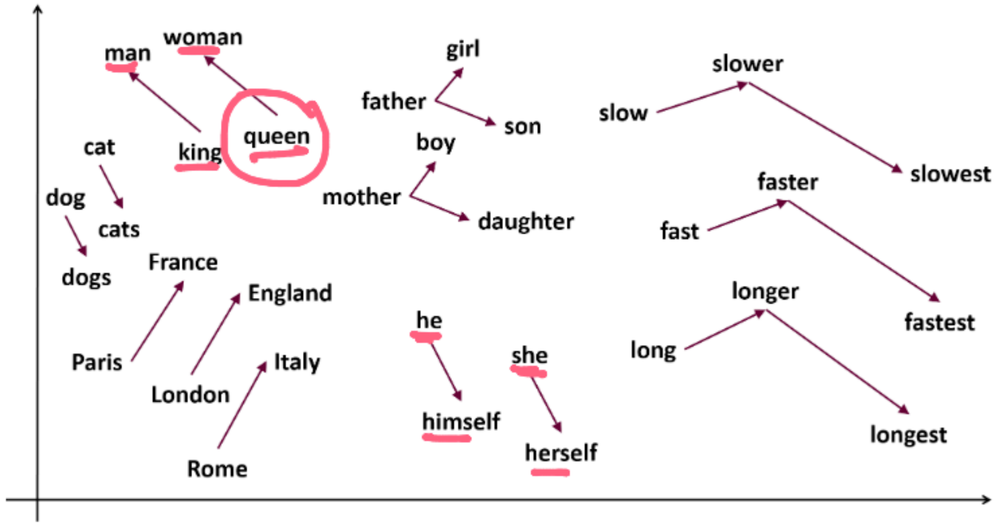
input	output
make	shalt
make	not
make	a
make	machine

Negative Sampling

input word	output word	target
make	shalt	1
make	aaron	0
make	taco	0

Now that we've understood how Word2Vec can help generate meaningful and contextual word

NLP with gensim (word2vec)



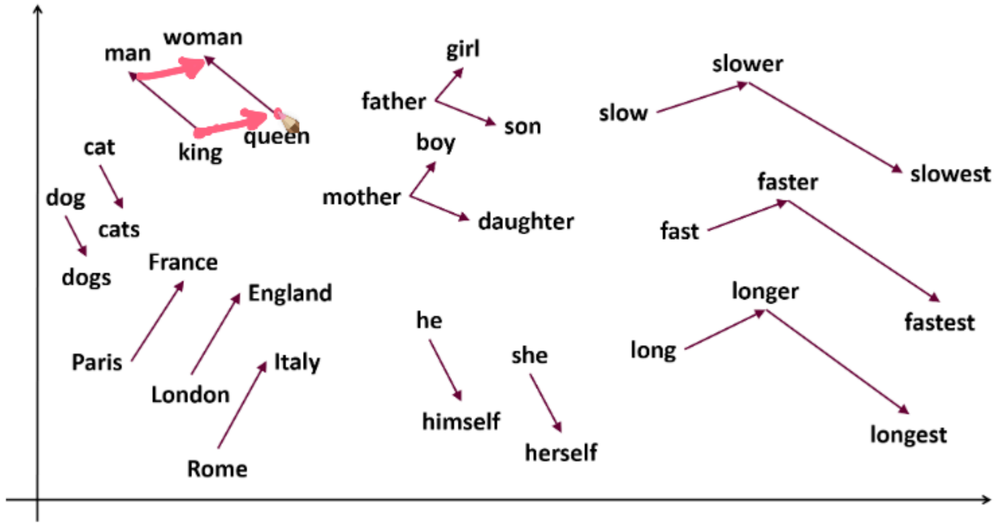
NLP (Natural Language Processing) is a fast developing field of research in recent years, especially by Google, which depends on NLP technologies for managing its vast repositories of text contents.

In this study unit we will lay a simple introduction to this field through the use of the excellent **gensim** Python package of Radim Rehurek's and his excellent **word2vec Tutorial**:
https://rare-technologies.com/word2vec-tutorial.

We have followed Radim's code with some supplements and more examples, and adapted it to an **IMDB** movie reviews dataset from Cornell university:
https://www.cs.cornell.edu/people/pabo/movie-review-data

You may downlaod this dataset more conveniently from here:
http://www.samyzaf.com/ML/nlp/aclImdb.zip

NLP with gensim (word2vec)



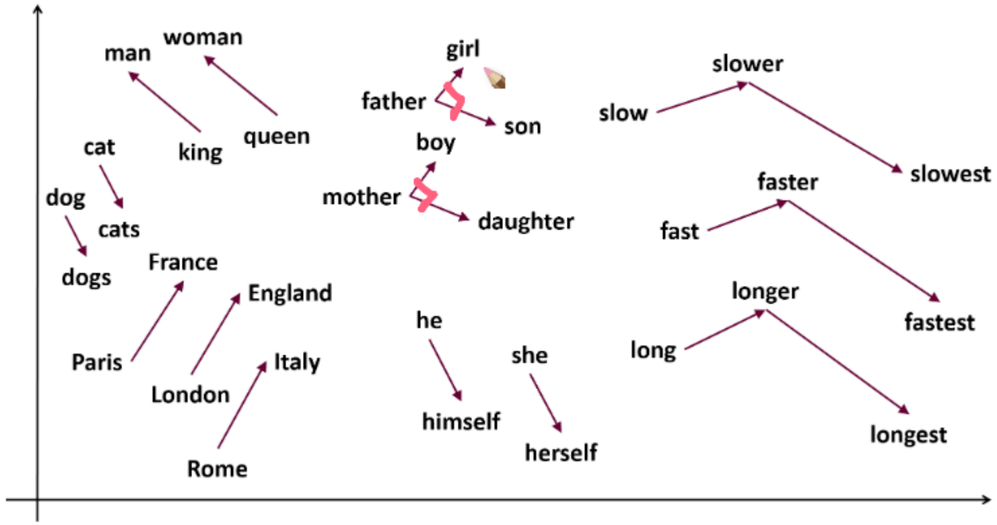
NLP (Natural Language Processing) is a fast developing field of research in recent years, especially by Google, which depends on NLP technologies for managing its vast repositories of text contents.

In this study unit we will lay a simple introduction to this field through the use of the excellent **gensim** Python package of Radim Rehurek's and his excellent **word2vec Tutorial**:
https://rare-technologies.com/word2vec-tutorial.

We have followed Radim's code with some supplements and more examples, and adapted it to an **IMDB** movie reviews dataset from Cornell university:
https://www.cs.cornell.edu/people/pabo/movie-review-data

You may downlaod this dataset more conveniently from here:
http://www.samyzaf.com/ML/nlp/aclImdb.zip

NLP with gensim (word2vec)



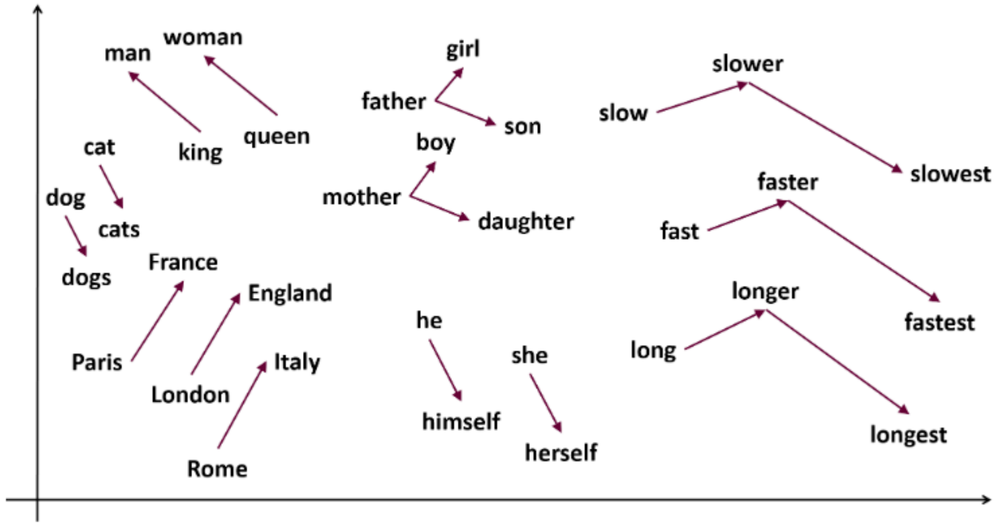
NLP (Natural Language Processing) is a fast developing field of research in recent years, especially by Google, which depends on NLP technologies for managing its vast repositories of text contents.

In this study unit we will lay a simple introduction to this field through the use of the excellent **gensim** Python package of Radim Rehurek's and his excellent **word2vec Tutorial**:
https://rare-technologies.com/word2vec-tutorial.

We have followed Radim's code with some supplements and more examples, and adapted it to an **IMDB** movie reviews dataset from Cornell university:
https://www.cs.cornell.edu/people/pabo/movie-review-data

You may downlaod this dataset more conveniently from here:
http://www.samyzaf.com/ML/nlp/aclImdb.zip

NLP with gensim (word2vec)



NLP (Natural Language Processing) is a fast developing field of research in recent years, especially by Google, which depends on NLP technologies for managing its vast repositories of text contents.

In this study unit we will lay a simple introduction to this field through the use of the excellent **gensim** Python package of Radim Rehurek's and his excellent **word2vec Tutorial**:
https://rare-technologies.com/word2vec-tutorial.

We have followed Radim's code with some supplements and more examples, and adapted it to an **IMDB** movie reviews dataset from Cornell university:
https://www.cs.cornell.edu/people/pabo/movie-review-data

You may downlaod this dataset more conveniently from here:
http://www.samyzaf.com/ML/nlp/aclImdb.zip

(Q)

$w_j \rightarrow$ 100-dim

Vector $d_i \rightarrow w_3\ w_6\ w_{12}\ w_{15}\ w_{26}$

Task: Query: $w_6\ w_4\ w_5 \rightarrow Vec_Q$

↓

docs Most similar

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:28

doc-vec

1. avg
2. ✓ Centroid
3. weighted avg → TFIDF → ✓

✓ d_1: w_3 w_6 w_4=of

$\downarrow$ $\downarrow$ $\downarrow$

$Tfidf_3 v_3 + Tfidf_6 v_6 + v_4 \cdot Tfidf_4$

Remains

```
[ ]         for item in cosine_list[:num]:
                papers_list.append((item[0], item[1], item[2]))
            return papers_list
```

Method to use the base function and retrieve top matching documents

```
[ ] def query(query, top_matches=10):
        model_to_use = model
        df_covid_to_use = df_covid
        return rank_docs(model_to_use, query, df_covid_to_use, top_matches)
```

```
[ ] query('origin of corona virus')
```

```
[('Role of Nonstructural Proteins in the Pathogenesis of SARS-CoV-2',
  '10.1002/jmv.25858',
  2.381886614774109),
 ('Re-emergence of a genetic outlier strain of equine arteritis virus: Impact on phylogeny',
  '10.1016/j.virusres.2014.12.009',
  2.3744738115509545),
 ('Emerging and Neglected Viruses of Zoonotic Importance in Croatia',
  '10.3390/pathogens10010073',
  2.370784610584278),
 ('The global emergence of severe acute respiratory syndrome coronavirus 2 in human',
  '10.1007/s13337-020-00613-y',
  2.369730359573394),
```

```
[ ]   banyangvirus in Northern bats from Germany',
        '10.1038/s41598-020-58466-w',
        2.357634998523873),
       ('Update of the current knowledge on genetics, evolution, immunopathogenesis, and transmission for coronavirus
      disease 19 (COVID-19)',
        '10.7150/ijbs.48812',
        2.3548538950775884)]
```

Conclusion

- **Continous text representations** are used to **capture syntactic and semantic similarity** that, **discrete text representations dont** capture.
- **SVD** based methods **suffer from scaling** and in practice, iterative method like Word2Vec is used.
- **Word2Vec** models the data as multi-class classification problem and in doing so, **learns the word representations**.
- Two architectures of Word2Vec - (i) **CBoW** and (ii) **Skip-Gram**.
- Training methods available - (i) **Negative Sampling** and (ii) **Hierarchical Softmax**
- **Gensim** library is used for building Word2Vec from scratch.
- Finally, we **built** a **search engine** based on **semantic similarity between query and abstracts** of the publications.

Post Read

```
[ ]   banyangvirus in Northern bats from Germany',
        '10.1038/s41598-020-58466-w',
        2.357634998523873),
       ('Update of the current knowledge on genetics, evolution, immunopathogenesis, and transmission for coronavirus
      disease 19 (COVID-19)',
        '10.7150/ijbs.48812',
        2.3548538950775884)]
```

Conclusion

- **Continous text representations** are used to **capture syntactic and semantic similarity** that, **discrete text representations dont** capture.
- **SVD** based methods **suffer from scaling** and in practice, iterative method like Word2Vec is used.
- **Word2Vec** models the data as multi-class classification problem and in doing so, **learns the word representations**.
- Two architectures of Word2Vec - (i) **CBoW** and (ii) **Skip-Gram**.
- Training methods available - (i) **Negative Sampling** and (ii) **Hierarchical Softmax**
- **Gensim** library is used for building Word2Vec from scratch.
- Finally, we **built** a **search engine** based on **semantic similarity between query and abstracts** of the publications.

Post Read

(Google)
✓ Word2Vec (2013) → very small-context/window

→ ✓ Glove (2014) → Classical MF (doc-word) + w2Vec A_{nxN}
(stanford)

→ fastext (2016-17)
✓ (FB)

→ non-vocabulary words
facebook

With over 50,000 publications and research papers on the corona-virus family till date, it has become difficult to search across and get useful insights for medical practitioners.

As a *DataScientist* at Google, you are tasked with solving this problem with the help of Machine Learning.

Efficient Estimation of Word Representations in Vector Space

GloVe: Global Vectors for Word Representation

How can we solve for this problem?

- Can we match keywords from user queries that are present in abstract?
- If we do keyword matching, will we be able to understand the user's intent? For e.g: 'origin' and 'discovery'
- Should we consider the context of the words, then?

Let us build a search engine using Word Embeddings

Datset