ALGONQUIN COLLEGE

CST8507: NATURAL LANGUAGE PROCESSING

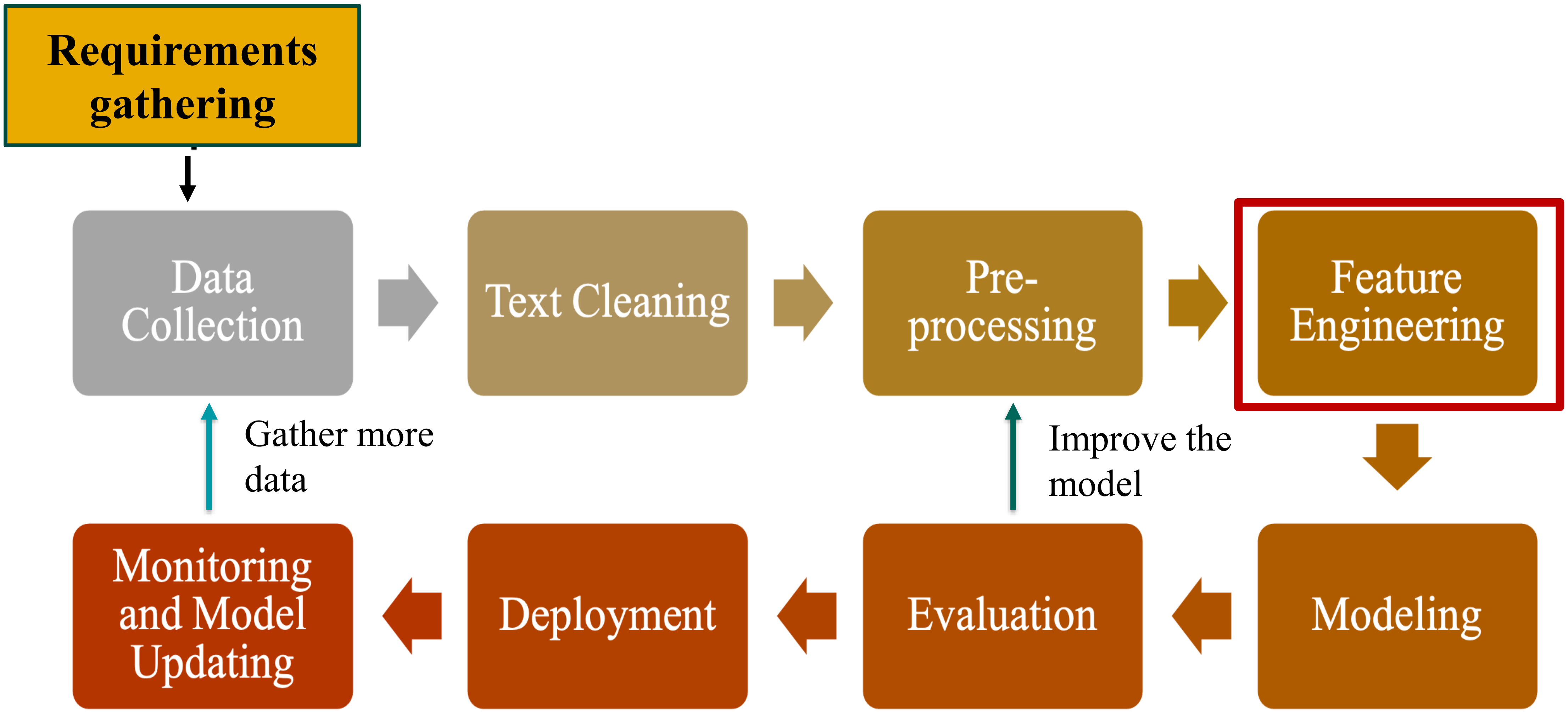
WEEK#4
WORD EMBEDDING

DEVELOPED BY
HALA OWN, PH.D.

Lesson Agenda

- **Prediction based Text representation (word Embedding)**
  - CBOW, Skip-Gram and SGNS
    - Word2Vec
    - FastText
- **Count-Based / Matrix Factorization Methods**
    - Glove

NLP Development Life Cycle



Text Representation Techniques

- **Frequency based Text representation:**
  - One-Hot Encoding
  - Bag of Words
  - Bag of N-Grams
  - TF-IDF
- **Prediction based Text representation (word Embedding)**
- **Universal Text Representations**

Comparing Feature Representations For Audio, Image And Text

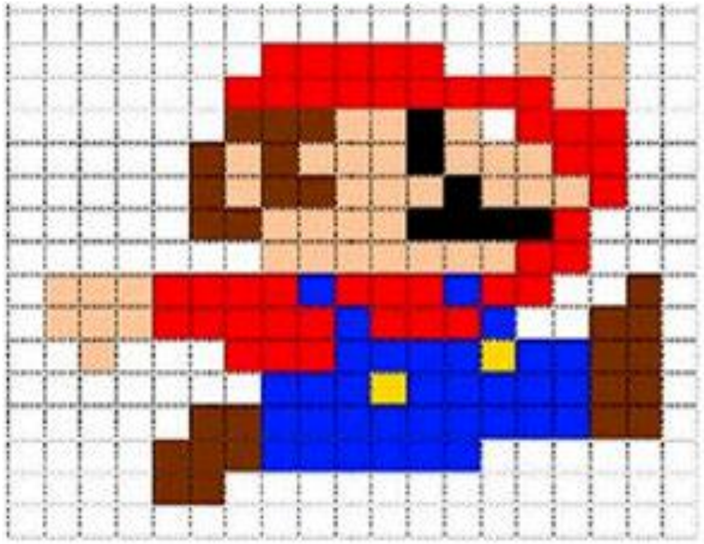
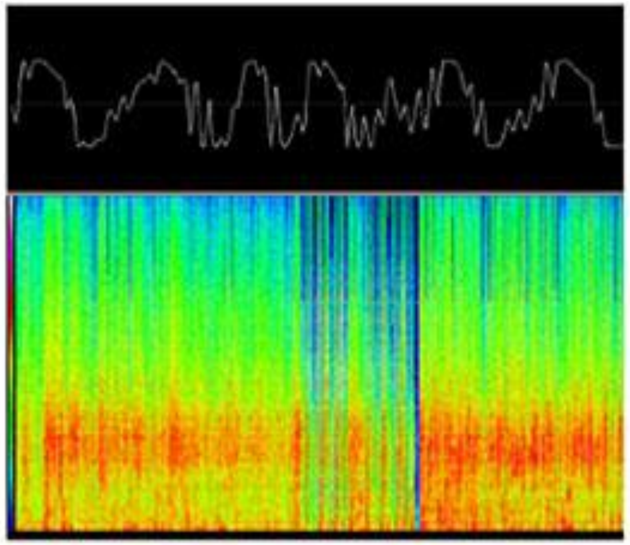


IMAGE PIXELS (DENSE)



AUDIO SPECTROGRAM (DENSE)

bacon	beans	beautiful	blue	breakfast	brown	dog	eggs	fox
0	0	1	1	0	0	0	0	0

WORD VECTORS (SPARSE)

Frequency based Text representation: Limitation

- High-dimensional representation
- sparse
- OOV words
- Lack of semantic meaning

Poor Performance On majority of NLP Tasks

Relation between Word Senses: Word Similarity

- Cat is not a synonym of dog, but cats and dogs are certainly similar words

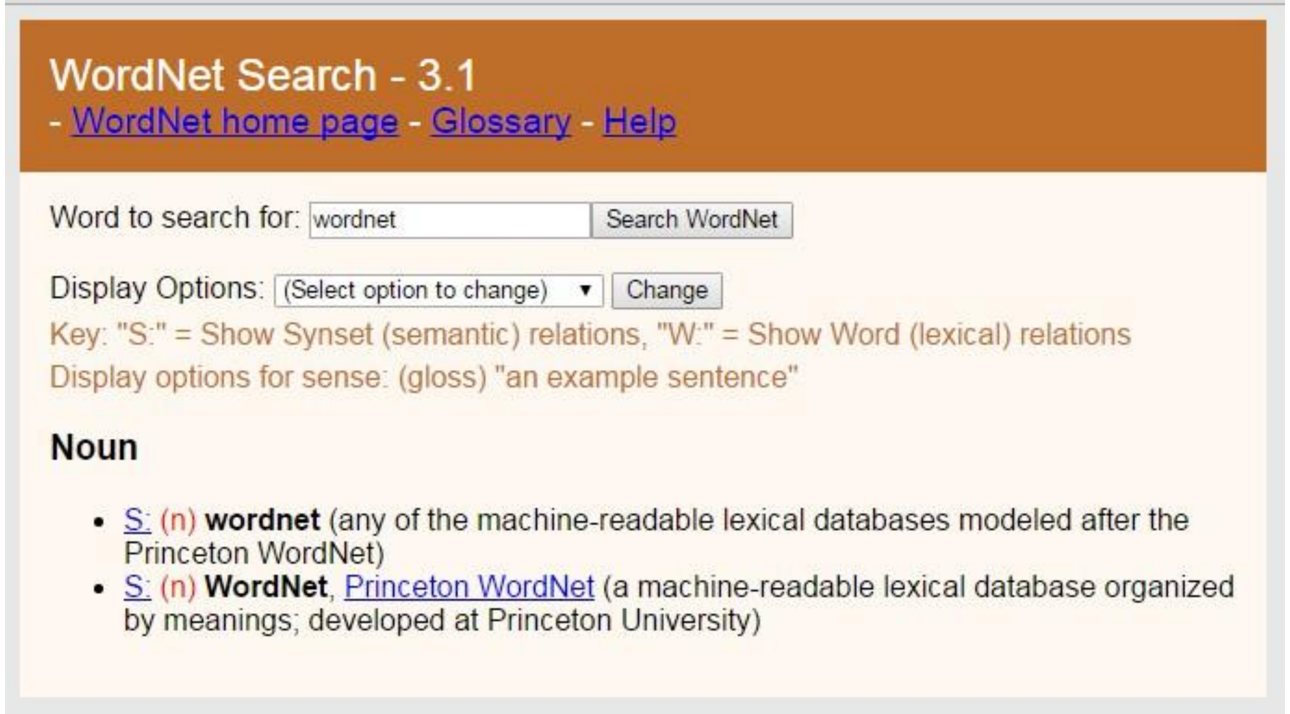
- **A semantic field** is a set of words which cover a particular semantic domain
  - Restaurants: waiter, menu, plate, food, chef
  - Houses: door, roof, kitchen, family, bed

One way of getting values for word similarity is to ask humans to judge how similar one word is to another

ALGONQUIN COLLEGE

WordNet

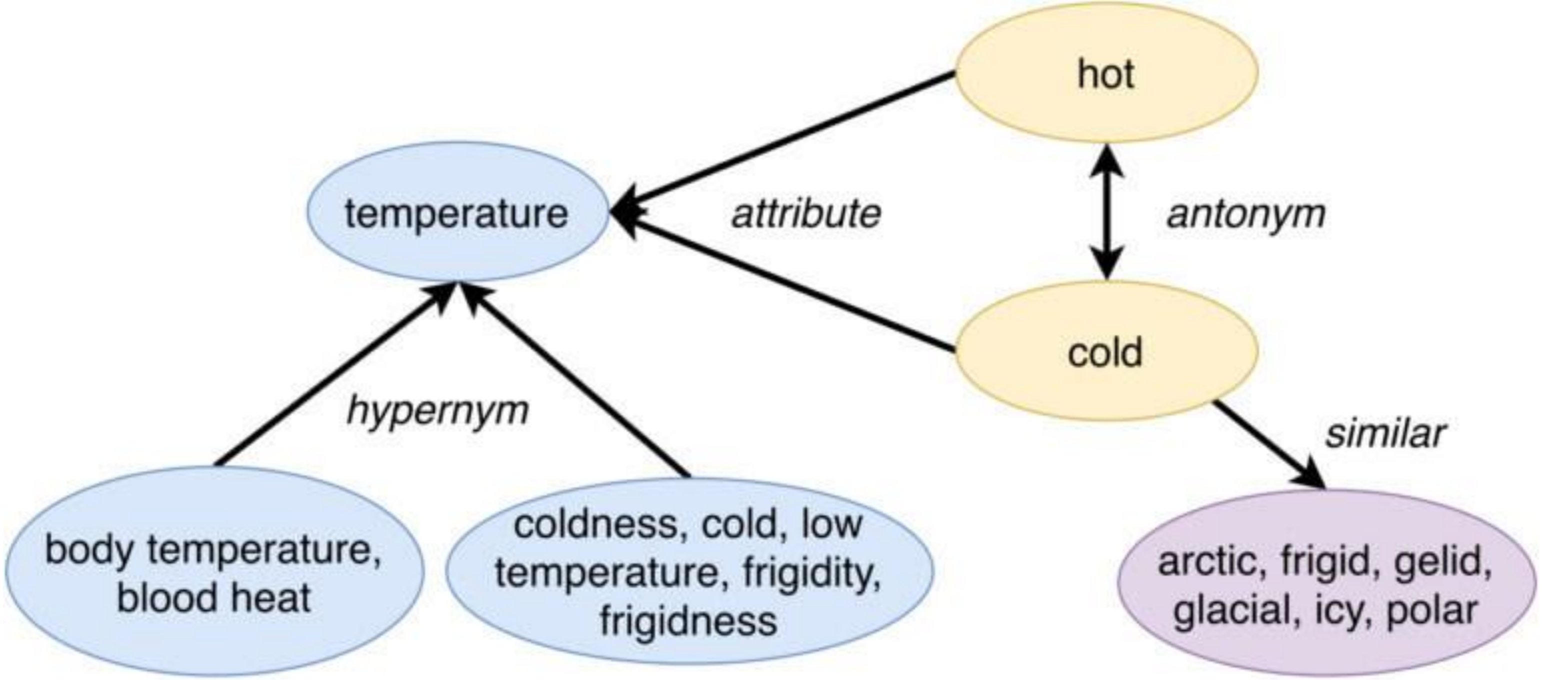
a large lexical database of English words, where words are grouped into sets of synonyms called **synsets**. These synsets are connected by various semantic relationships, such as synonymy, hypernymy, and hyponymy,etc.



https://wordnet.princeton.edu/

WordNet…

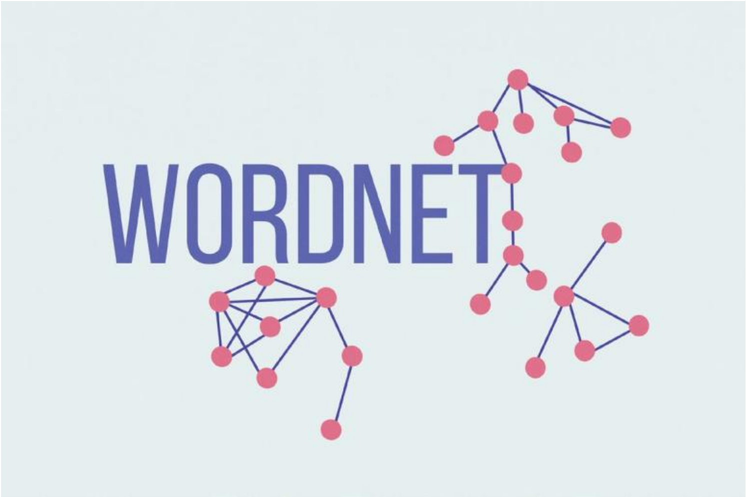
Database of lexical relations for English



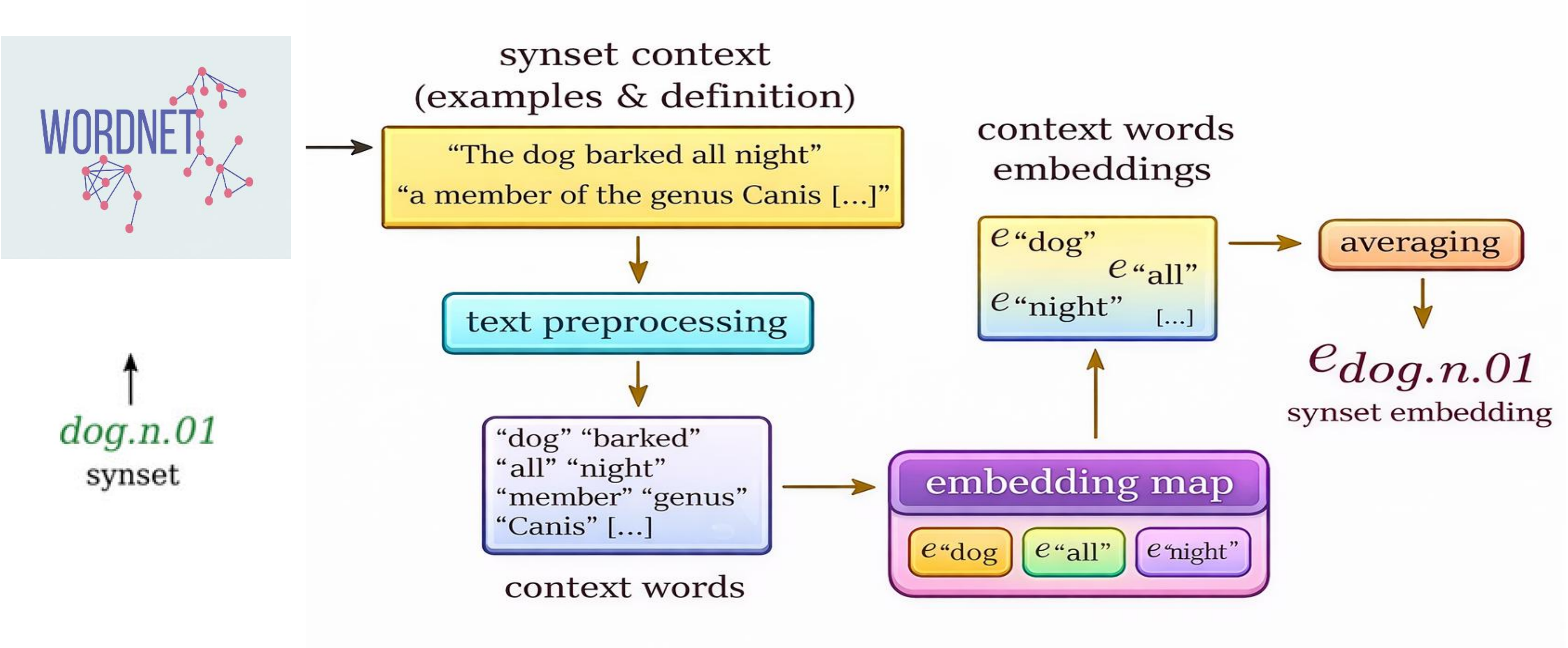
https://www.scaler.com/topics/nlp/wordnet-in-nlp/

WordNet – Important Concepts

- **Synset**: A set of synonyms that share a common meaning.
- **Hypernym**: A general term that encompasses more specific terms (e.g., "animal" is a hypernym of "dog").
- **Hyponym**: A specific term within a broader category (e.g., "dog" is a hyponym of "animal").
- **Meronym**: A term that denotes a part of something (e.g., "wheel" is a meronym of "car").
- **Holonym**: A term that denotes a whole of which the meronym is a part (e.g., "car" is a holonym of "wheel").
- **Antonym**: Words that have opposite meanings (e.g., "hot" and "cold").
- **Troponym**: A verb that denotes a specific manner of doing something (e.g., "run" is a troponym of "move").
- **Entailment**: A relationship where one verb implies another (e.g., "snore" entails "sleep").



WordNet Applications in NLP Tasks: Semantic Test presentation



The context of each synset is tokenized into words, with each word mapped to a vector representation via the learned embedding matrix. The synset vector is the centroid produced by averaging all context word embeddings.

Reference: Text classification with semantically enriched word embeddings
Published online by Cambridge University Press: **06 April 2020**

WordNet Applications in NLP Tasks: Query Expansion

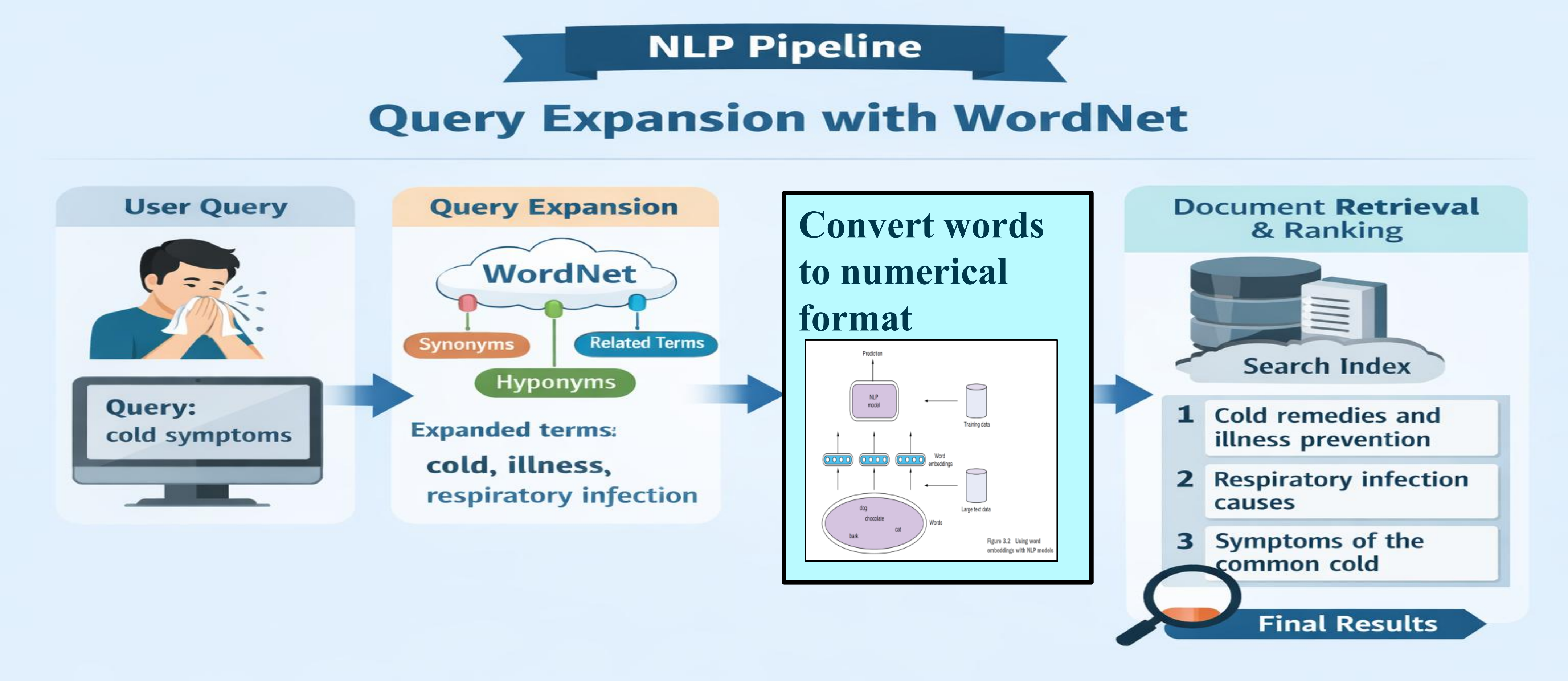


Image generated by ChatGPT with some updates

WordNet: Limitations

- Limited Coverage and Static Nature
- Not Computational
- Domain Specificity
- Language Limitation
- Manual Curation Challenges

Key Terms

- ***Distributional similarity***: the meaning of a word can be understood from the context
- ***Distributional hypothesis***: words that occur in similar contexts have similar meanings.

Vector Semantics(Word Embedding)

Ccomputational model that learn the **linguistic units** (words, phrases, or documents) representations based on distributional properties of these units in a large corpus.

- Representation **linguistic units** as **vectors in a multi-dimensional space**.
- Encoding semantic information using mathematical vectors.
- Standard way to represent word meaning in NLP.

ALGONQUIN COLLEGE

Word Embedding

Representation of words as vectors of numbers in a high-dimensional space. It captures semantic and contextual information about the word.

Input: large number of corpus, Vocabulary V, and vector of dimension d

output

$$f: V \to R^d$$

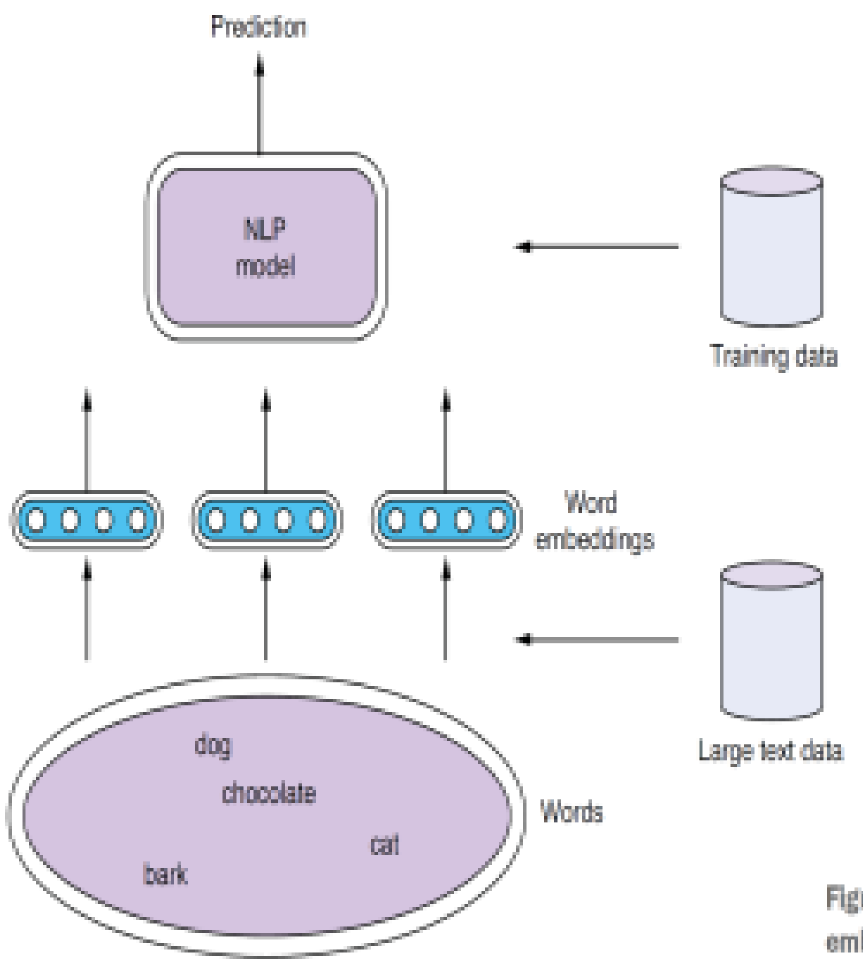
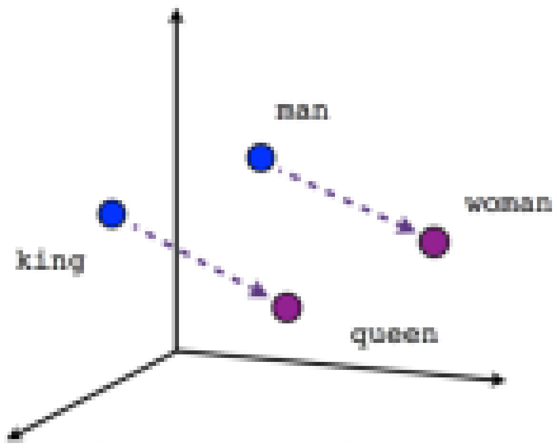


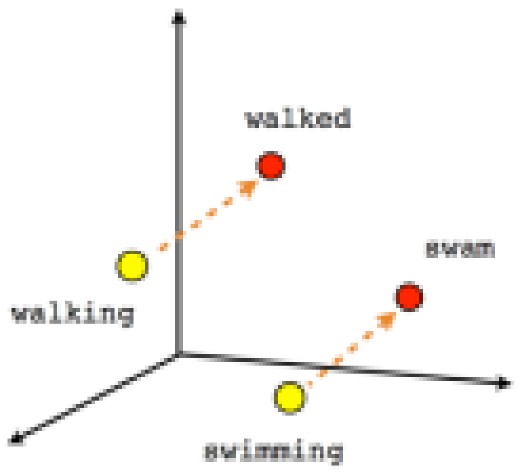
Figure 3.2 Using word embeddings with NLP models

Image source: real world natural langue processing book

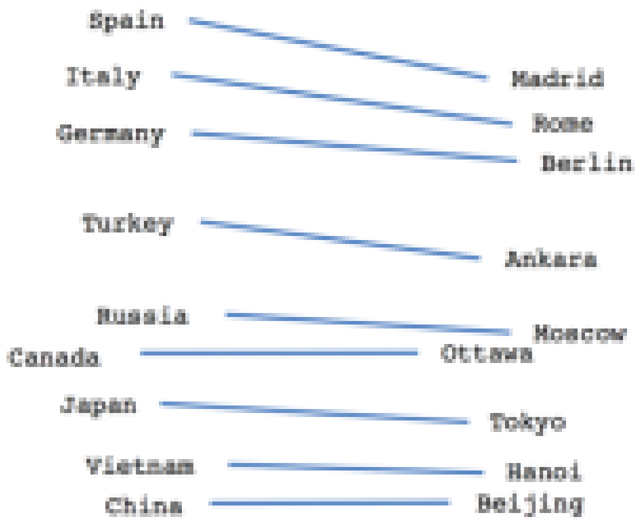
Word Embedding…
Analogy



Male-Female



Verb tense



Country-Capital

vector('*king*') - vector('*man*') + vector('*woman*') ≈ vector('queen')

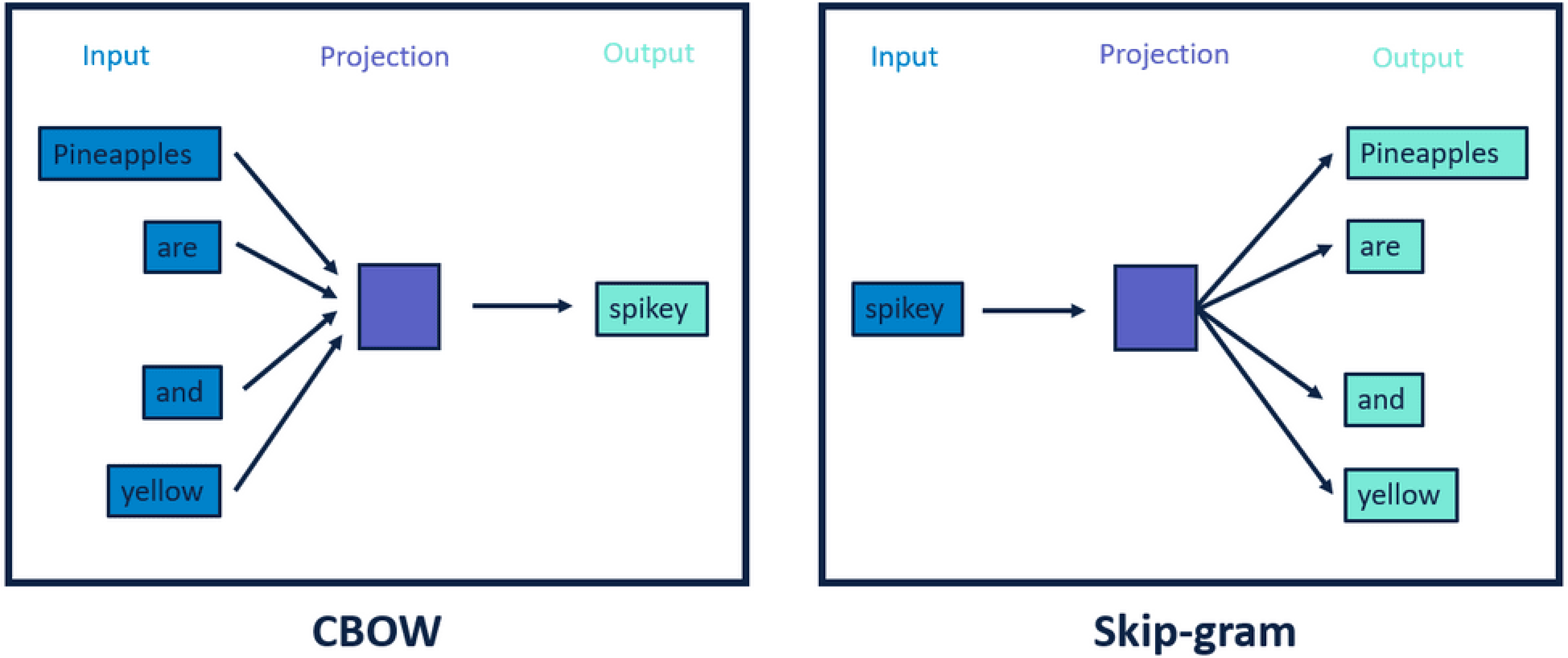
Prediction based Text representation
Big idea: self-supervision, Bengio et al. (2003) and Collobert et al. (2011)

- Popular embedding method
- Very **fast to train**
- Code available on the web
- **Predict** rather than **count**

ALGONQUIN COLLEGE

Prediction Based Text Representation(Word2Vec)

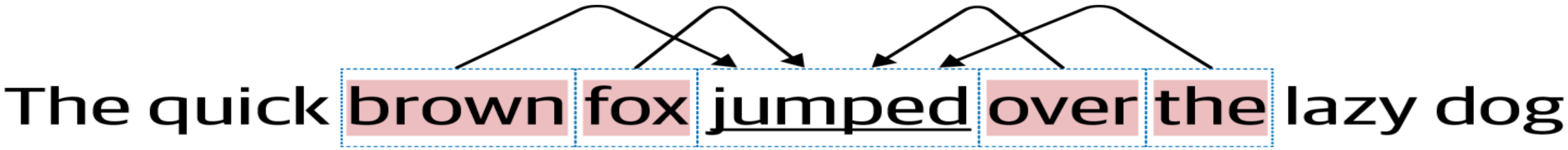
Word Embedding (creating dense vector representations of words)

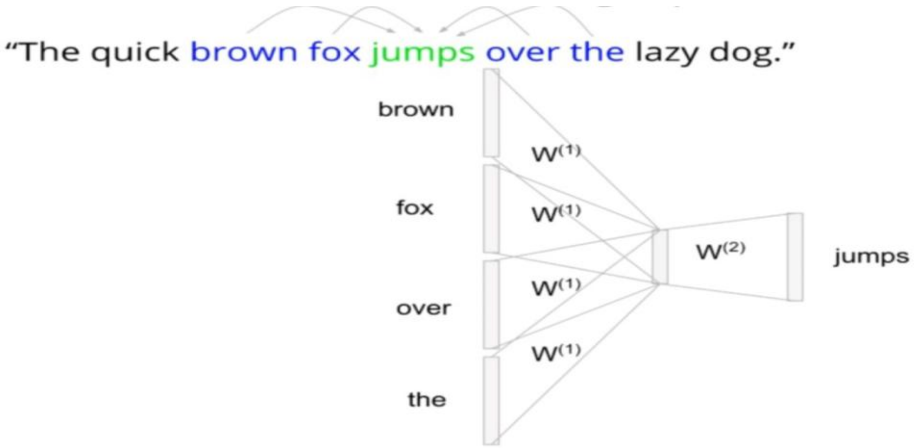


Source: https://dataaspirant.com/word-embedding-techniques-nlp/

CBOW

- Goal: Predict the **middle** word given the **words of the context**





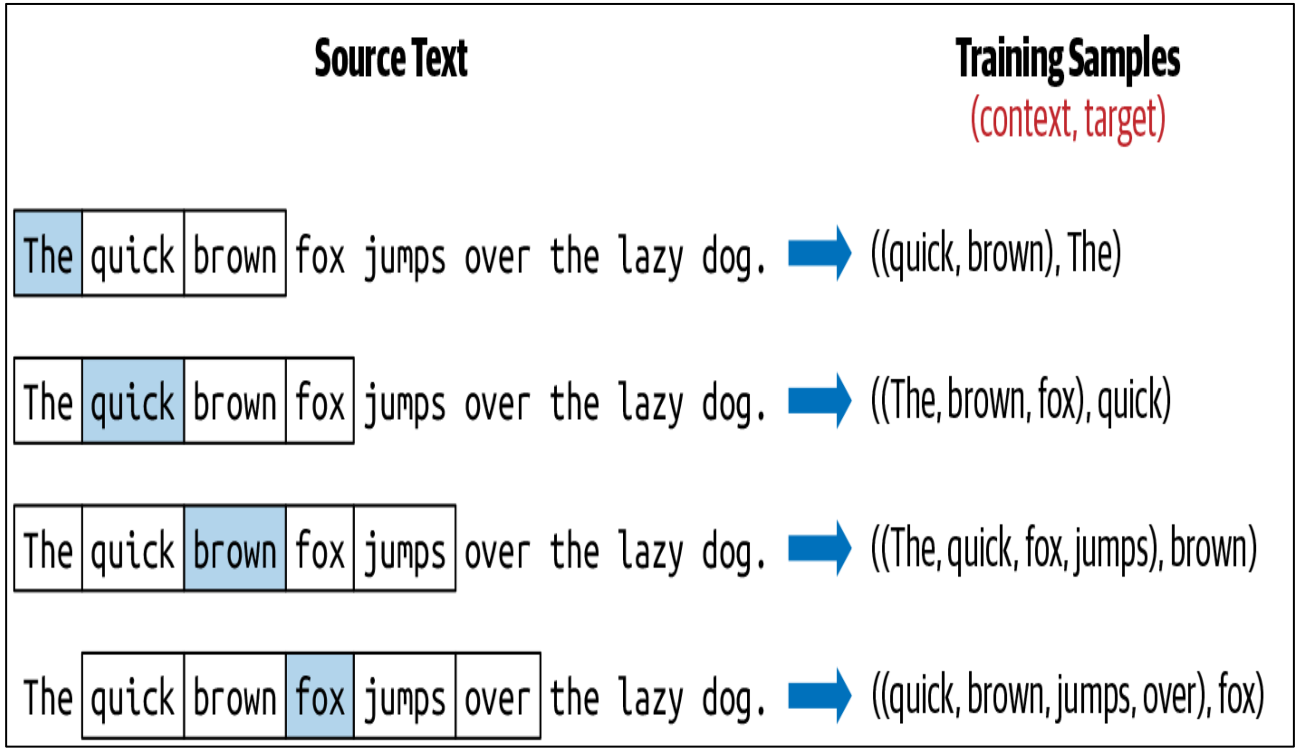
"Context size" could be considered 2 (or 4)

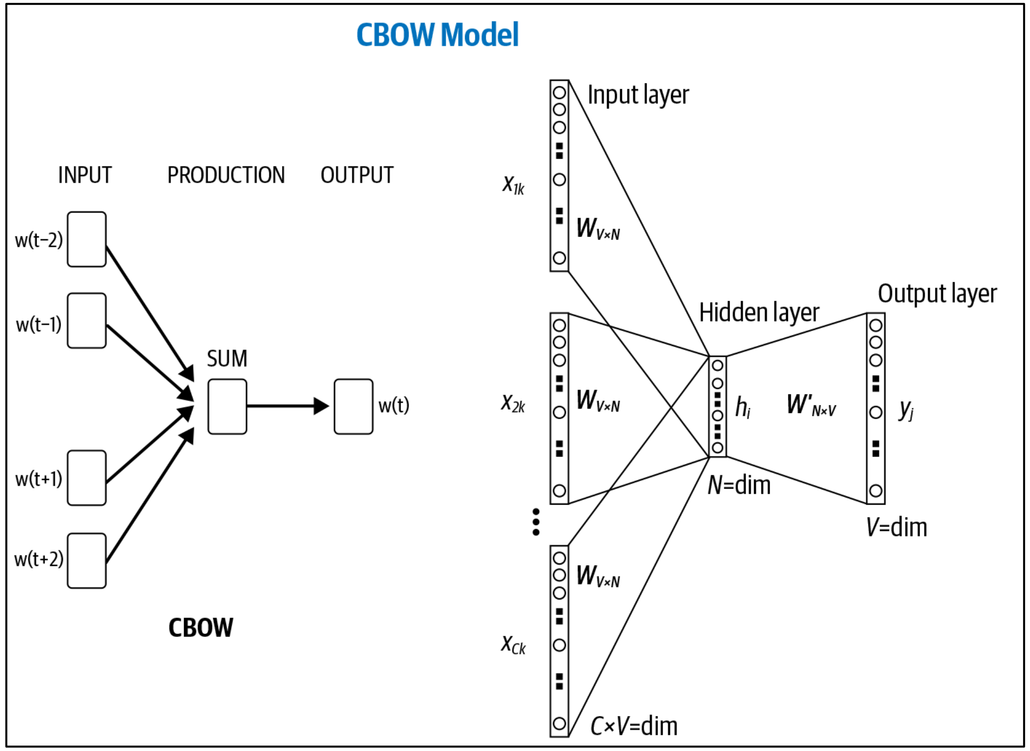
In practice, context size is usually set from 5-10 (on either side)

The input weight is $W^{(1)}$ for all input words (same weight used multiple times)

CBOW..

Window size=2





https://medium.com/co-learning-lounge/nlp-word-embedding-tfidf-bert-word2vec-d7f04340af7f

CBOW: Simple Example

Context

C1	this	0	1	0	0	0	0	0	0	0	0
C2	corpus	0	0	0	0	1	0	0	0	0	0
C3	context	0	0	0	0	0	0	0	0	1	0

Input-Hidden Weight

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16
17	18	19	20
21	22	23	24
25	26	27	28
29	30	31	32
33	34	35	36
37	38	39	40

Hidden Activation

5	6	7	8
17	18	19	20
33	34	35	36

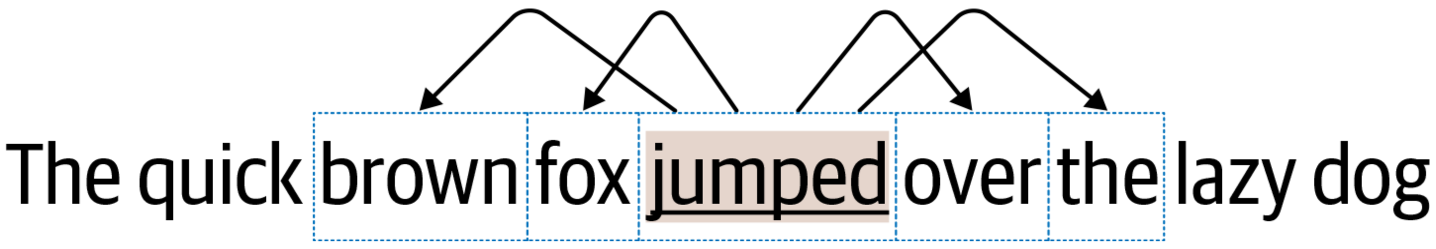
Average hidden Activation

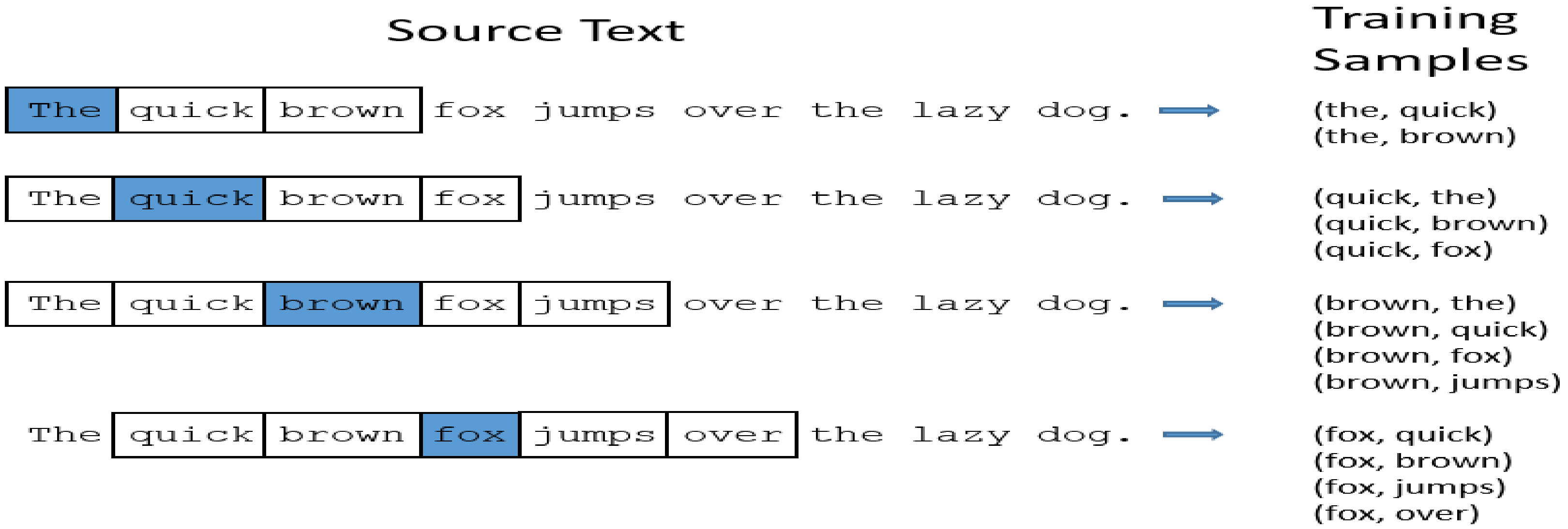
18.33333333	19.33333333	20.33333333	21.33333333

ALGONQUIN COLLEGE

Skip-gram

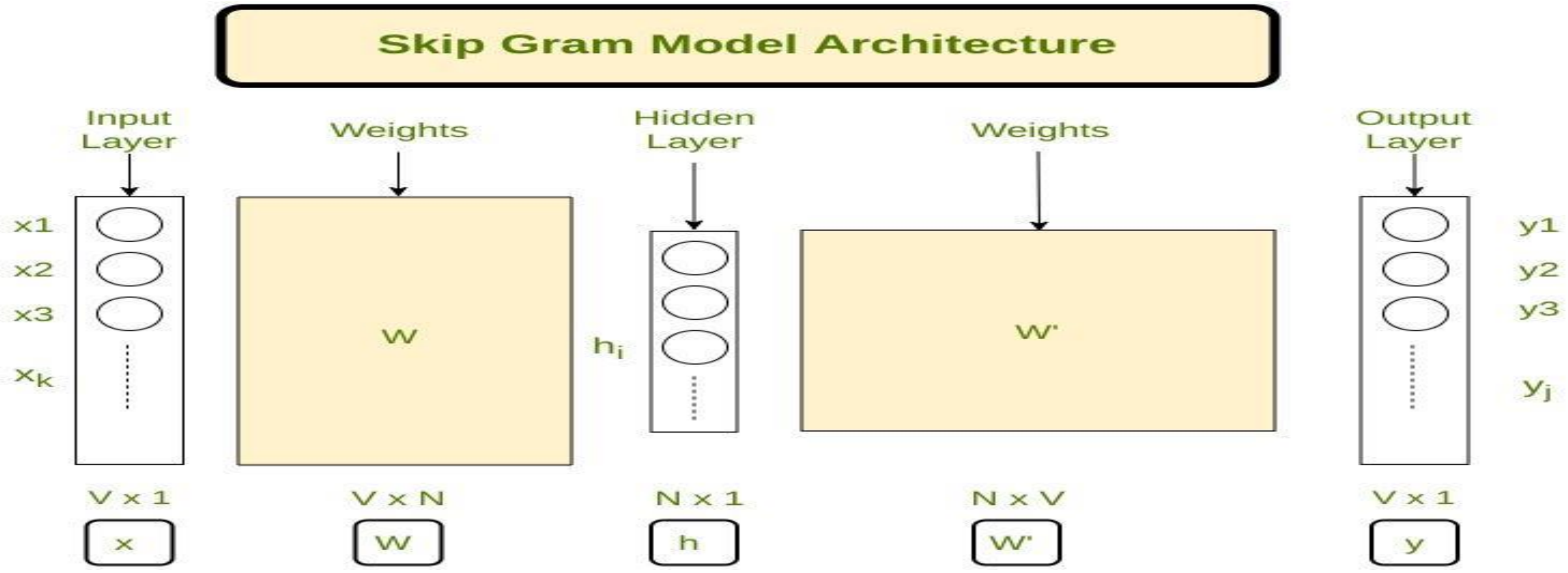
- Goal: Predict the context words given the middle word





Skip-gram…

the training objective is to minimize the summed prediction error across all context words in the output layer.



Skip-gram: Example

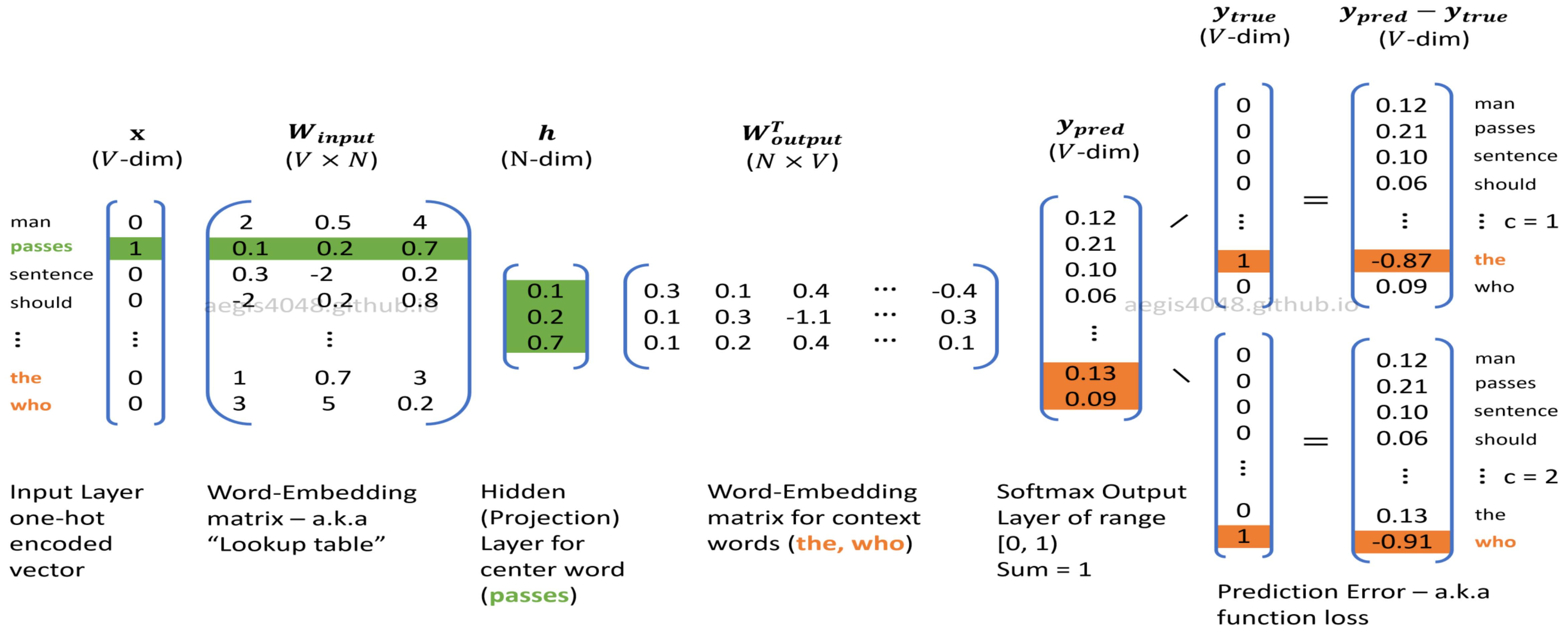


Image source: https://aegis4048.github.io/demystifying_neural_network_in_skip_gram_language_modeling

Skip-gram Prediction:Example

the cat sat on the mat

w_t = the → CLASSIFIER →

w_{t-2} = <start_{-2}>
w_{t-1} = <start_{-1}>
w_{t+1} = cat
w_{t+2} = sat

context size = 2

Skip-gram Prediction :Example…

the cat sat on the mat

w_t = cat → CLASSIFIER →

w_{t-2} = <start$_{-1}$>
w_{t-1} = the
w_{t+1} = sat
w_{t+2} = on

context size = 2

Skip-gram Prediction: :Example…

the cat sat on the mat

w_t = sat → **CLASSIFIER** →
w_{t-2} = the
w_{t-1} = cat
w_{t+1} = on
w_{t+2} = the

context size = 2

Skip-gram Prediction :Example…

the cat sat on the mat

w_t = on → CLASSIFIER →

w_{t-2} = cat
w_{t-1} = sat
w_{t+1} = the
w_{t+2} = mat

context size = 2

Skip-gram Prediction :Example…

the cat sat on the mat

w_t = the → **CLASSIFIER** →

w_{t-2} = sat
w_{t-1} = on
w_{t+1} = mat
w_{t+2} = <end$_{+1}$>

context size = 2

Skip-gram Prediction :Example…

the cat sat $\boxed{\text{on the } \underline{\text{mat}}}$

w_t = mat → **CLASSIFIER** →

w_{t-2} = on
w_{t-1} = the
w_{t+1} = <end$_{+1}$>
w_{t+2} = <end$_{+2}$>

context size = 2

Skip-gram vs CBOW

- CBOW is comparatively **faster** to train than skip-gram and better for frequently occurring words
- Skip-gram is slower but works well for **smaller amount of data**
- CBOW is an **easier** classification problem than Skip-gram

Skip-gram Negative Sampling(SGNS): Approach

1. Treat the target word ***t*** and a neighboring context word ***c*** as **positive examples**.
2. Randomly sample other words in the lexicon to get negative examples
3. Use **logistic regression** to train a classifier to distinguish those two cases
4. Use the learned weights as the embeddings

SGNS : how to learn vectors

- Given the set of positive and negative training instances, and an initial set of embedding vectors
- The goal of learning is to adjust those word vectors such that we:
  - **Maximize** the similarity of the target word, context word pairs (w , c_{pos}) drawn from the positive data
  - **Minimize** the similarity of the (w , c_{neg}) pairs drawn from the negative data.

SGNS : how to learn vectors…

- Training sentence:

... lemon, a tablespoon of apricot jam a pinch ...

c1 c2 t c3 c4

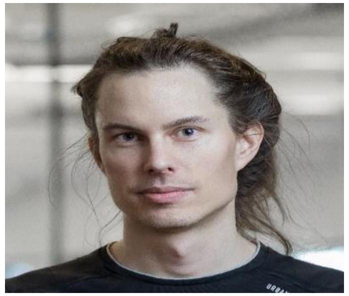
positive examples +

t	c
apricot	tablespoon
apricot	of
apricot	preserves
apricot	or

negative examples -

t	c	t	c
apricot	aardvark	apricot	twelve
apricot	puddle	apricot	hello
apricot	where	apricot	dear
apricot	coaxial	apricot	forever

Pretrained Word Embeddings Models



- **Word2vec** (Mikolov et al.) 2013
- https://code.google.com/archive/p/word2vec/
- **Fasttext** http://www.fasttext.cc/ 2016
- **Glove** (Pennington, Socher, Manning) 2014
- http://nlp.stanford.edu/projects/glove/



Google’s Word2Vec

- Gensim package :Google’s *pre-trained* Word2Vec model in Python.
- This model is trained on the vocabulary of **3 million** words and phrases from100 billion words of the Google News dataset.
- The vector length for each word is **50,100,300**.

Google’s Word2Vec

- Install genism library

```
conda install -c conda-forge genism
```

- Genism word2vec Model Training

```
model = Word2Vec(text, min_count=1,vector_size= 50, window =5,
sg = 1, negative=5)
```

Demo

- inclassCode

The GloVe (Global Vector for word presentation)

- Unsupervised learning model that can be used to obtain dense word vectors.
- invented in Stanford by Pennington et al. https://nlp.stanford.edu/projects/glove/
- https://github.com/stanfordnlp/GloVe

Glove Algorithm

Step—1: Collect a large datasetof words and their co-occurrences. This can be a text corpus, such as documents or web pages.

↓

Step—2: Preprocess the dataset by tokenizing the text into individual words and filtering out rare or irrelevant words.

↓

Step—3: Construct a co-occurrence matrix that counts the number of times each word appears in the same context as every other word.

↓

Step—4: Use the co-occurrence matrix to compute the word embeddings using the GloVe algorithm. This involves training a model to minimize the error between the word vectors' dot product and the co-occurrence counts' logarithm.

↓

Step—5: Save the resulting word embeddings to a file or use them directly in your model.

GloVe :Co-Occurrence Matrix

I love Programming. I love Math. I tolerate Biology.

	I	love	Programming	Math	tolerate	Biology	.
I	0	2	0	0	1	0	2
love	2	0	1	1	0	0	0
Programming	0	1	0	0	0	0	1
Math	0	1	0	0	0	0	1
tolerate	1	0	0	0	0	1	0
Biology	0	0	0	0	1	0	1
.	1	0	1	1	0	1	0

Window size = 1

The GloVe (Global Vector for word presentation)

Load the embeddings and use them as fixed word vectors in your application.

- https://nlp.stanford.edu/projects/glove/
- **Download pre-trained word vectors**
- Pre-trained word vectors. This data is made available under the Public Domain Dedication and License v1.0 whose full text can be found at: http://www.opendatacommons.org/licenses/pddl/1.0/.Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, & 300d vectors, 822 MB download): glove.6B.zip
- Common Crawl (42B tokens, 1.9M vocab, uncased, 300d vectors, 1.75 GB download): glove.42B.300d.zip
- Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors, 2.03 GB download): glove.840B.300d.zip
- Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased, 25d, 50d, 100d, & 200d vectors, 1.42 GB download): glove.twitter.27B.zip

glove.6B.300d , glove.6B.200d , glove.6B.100d , glove.6B.50d

Dealing with OOV

- Use a Default Vector
- Fallback to a Similar Word
- Train Your Own Embeddings

Better Solution: The FastText Model

The FastText Model

- The *FastText* model was introduced by Facebook in 2016 as an extension and supposedly improvement of the vanilla Word2Vec model.
- FastText is a framework for learning word representations and performing robust, fast, and accurate text classifications

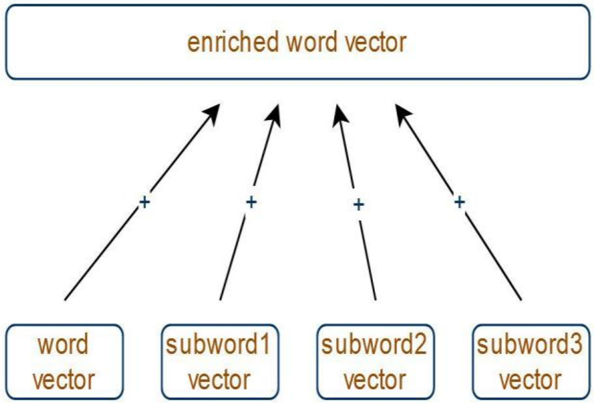
https://fasttext.cc/

FastText n-gram embedding model (Bojanowski et al., 2017): Enriching Word Vectors with Subword Information

ALGONQUIN COLLEGE

The FastText Model…

Sub-word generation



Download English vector:
https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.en.300.vec.gz

Access from Kaggle: https://www.kaggle.com/facebook/fasttext-wikinews

FastText: Subword Generation

For a word, we generate character n-grams of length **3 to 6** present in it

Word	Length(n)	Character n-grams
eating	3	<ea, eat, ati, tin, ing, ng>
eating	4	<eat, eati, atin, ting, ing>
eating	5	<eati, eatin, ating, ting>
eating	6	<eatin, eating, ating>

Two-step vector representation updating

1. First, the embedding for the center word is calculated by taking a **sum of vectors for the character n-grams** and the whole word itself
2. For the actual context words, we directly take their word vector from the embedding table without adding the character n-grams

ALGONQUIN COLLEGE

Advantages of FastText

- Capture fine level more gradual information
- Solve VOO
- Open-source, free, lightweight library.
- Handle text data in various languages .
- Has a simple and intuitive API

Word Embedding: Benefits

- Dimensionality reduction
- Semantic meaning
- Handling Out-of-Vocabulary (**OOV**)
- Transfer learning

Improved Performance On NLP Tasks

Word Embedding - Limitations

Context Insensitivity

Bias

Limited Semantic Adaptation

Dimensionality

Resource Intensive

OOV words

Universal Text Representations

- *contextual word representations*
- Advanced neural language models
- complex architectures involving multiple passes through the text and multiple reads from left to right and right to left to model the context of language
  - ELMo , BERT, ULMFiT

Word Embedding - Evaluation

1. **Intrinsic Evaluation**

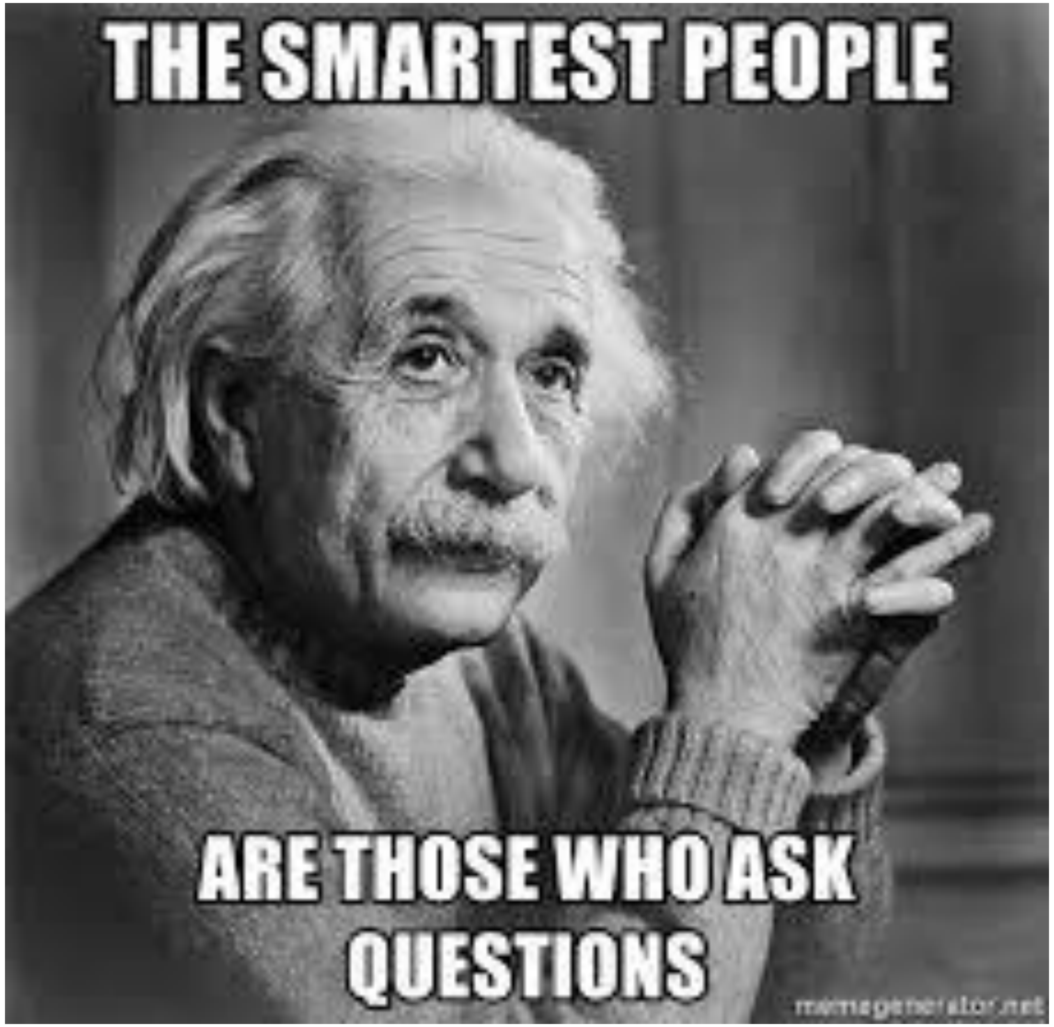
   - Assessing the quality of word embeddings independently of any specific task.
   - They focus on the internal properties of the embeddings. Word Similarity, Analogy Tasks, …

1. **Extrinsic Evaluation**

   - Assessing the quality of word embeddings based on their performance in downstream NLP tasks like: Text classification, NER, etc.

- WordNet and word senses
- Distributed representation
- Word Embedding
  - Word2Vec
  - Glove
  - FastText
- Evaluation of Word Embeddings
- Problems with Word Embedding

Q&A


THE SMARTEST PEOPLE
ARE THOSE WHO ASK
QUESTIONS
memegenerator.net