

Word Embeddings

Summer Methods Workshop

Taegyeon Kim | DHCSS, KAIST

Jul 31, 2025

Agenda

Things to be covered

- Text representation
- Word2Vec SGNS
- Other models: GloVe, FastText
- Contextual embeddings

Simple Document Representation

We have mostly dealt with document representation

- Document-term matrix (DTM/DFM)
 - Count matrix
 - TF-IDF matrix
- Rows represent documents
- Columns represent words (or types)

Simple Document Representation

An example corpus

- Doc 1: “The clever fox cleverly jumps over the lazy dog, showcasing its cleverness.”
- Doc 2: “Magic and mysteries mingle in the wizard’s daily musings, revealing mysteries unknown.”
- Doc 3: “Sunny days bring sunshine and sunsets, making sunny parks the best for sunny strolls.”

Simple Document Representation

An example DFM

Index	clever	jumps	lazy	dog	mysteries	...
Doc 1	3	1	1	1	0	...
Doc 2	0	0	0	0	2	...
Doc 3	0	0	0	0	0	...

Word Representation

How do we represent *words*?

- Vector semantics: a method that represents texts in a multi-dimensional space
- The simplest approach: one-hot encoding
 - A vector with one dimension per unique word in the vocabulary
 - Records 1 for that word and 0 for all the others
 - E.g., **author** = (0, 0, 0, 0, 1, ..., 0, 0) (the dimension size is **|V|**)

Word Representation

Limitations of one-hot encoding

- Semantics
 - Similarity: $\text{one-hot}(\text{author}) \perp \text{one-hot}(\text{writer})$
- Computation
 - Sparsity (mostly 0s in huge dimensional space: $|V|$)

Word Representation

Term-document matrix (TDM)

- Rows represents words, and columns represent documents
- Similar words have similar vectors because they tend to occur in similar documents (documents are the context)
- E.g., four words in four Shakespeare plays ([JM] Chp. 6)

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Figure 6.5 The term-document matrix for four words in four Shakespeare plays. The red boxes show that each word is represented as a row vector of length four.

Word Representation

Term-term matrix (TTM)

- Dimension: $|V| \times |V|$
- Each cell records the number of times the row word and the column word co-occur in some context
- Contexts are often a window around the word (e.g., ± 5)

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Figure 6.6 Co-occurrence vectors for four words in the Wikipedia corpus, showing six of the dimensions (hand-picked for pedagogical purposes). The vector for *digital* is outlined in red. Note that a real vector would have vastly more dimensions and thus be much sparser.

Word Embeddings

What are word embeddings?

- *Dense* vectors representing word meanings in a multi(low)-dimensional space
 - Word embeddings \subset word vectors
 - Traditional embeddings: $d = 50-1000$
 - Neural embeddings: $d = 500-$ (e.g., GPT-3: 12,288)
- Words are “embedded” into a low-dimensional space

Word Embeddings

What are word embeddings? (cont'd)

- Distributional hypothesis ([Joos 1950](#); [Harris 1954](#))
 - Word that occur in similar contexts tend to have similar meanings
 - “You shall know a word by the company it keeps” ([Firth 1957](#))
- E.g., oculists & eye-doctor: eyes, examine, diagnose, patient, etc.

Word Embeddings

If we have seen

- “... spinach sauteed with garlic over rice ...”
- “... chard stems and leaves are delicious ...”
- “... collard greens and other salty leafy greens ...”

We can guess what **ongchoi** is

- **ongchoi** is delicious sauteed with garlic
- **ongchoi** is superb over rice
- **ongchoi** leaves with salty sauces



Word Embeddings

Why useful?

- Direct object of interest (to study word usage and meaning)
- Downstream tasks: feature representations
 - Part of speech tagging
 - Named entity recognition
 - Text classification
 - Etc.

Word Embeddings

Why useful?

- A measure of word meaning

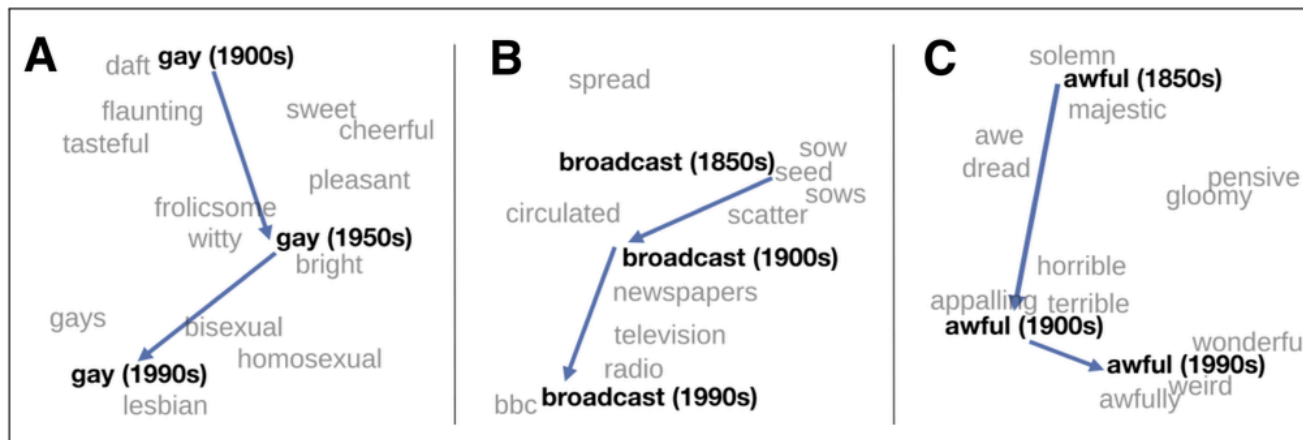
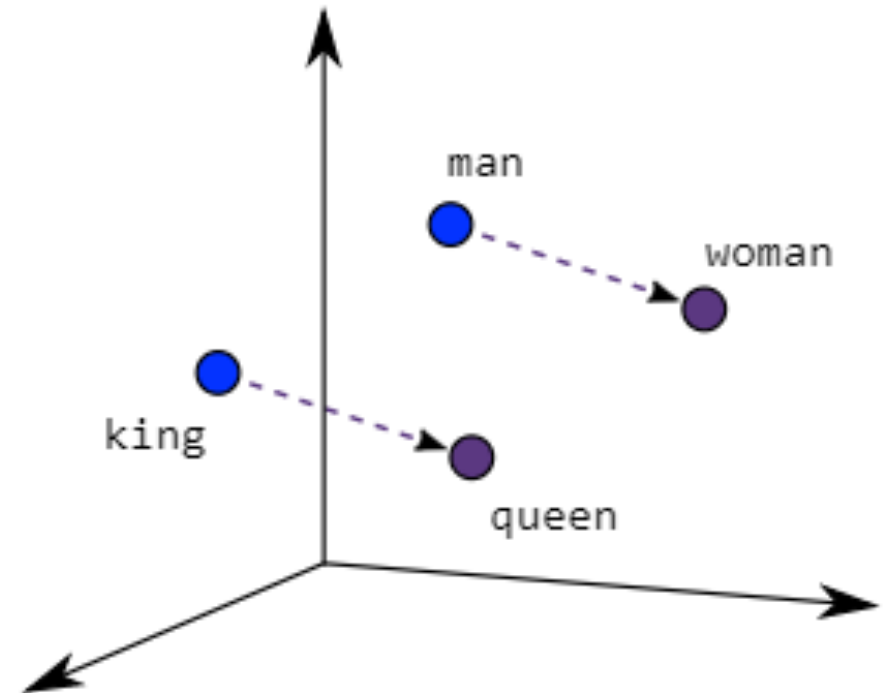


Figure 6.17 A t-SNE visualization of the semantic change of 3 words in English using word2vec vectors. The modern sense of each word, and the grey context words, are computed from the most recent (modern) time-point embedding space. Earlier points are computed from earlier historical embedding spaces. The visualizations show the changes in the word *gay* from meanings related to “cheerful” or “frolicsome” to referring to homosexuality, the development of the modern “transmission” sense of *broadcast* from its original sense of sowing seeds, and the pejoration of the word *awful* as it shifted from meaning “full of awe” to meaning “terrible or appalling” (Hamilton et al., 2016).

Word Embeddings

Why useful?

- Encoding similarity
 - For similar words, their embeddings point in similar directions ($\vec{e}_{\text{author}} \rightarrow \vec{e}_{\text{writer}}$)
 - E.g., $\vec{e}_{\text{author}} \rightarrow \vec{e}_{\text{writer}}$
 - Similarity in relations (“vector arithmetic”)
 - E.g., $\text{king} - \text{man} + \text{woman} \approx \text{queen}$ (Mikolov et al. 2013)



Word Embeddings

Why useful?

- Automatic generalization
 - Information retrieval
 - E.g., identifying academic papers about literacy in the digital age
 - Seed keywords: **digital literacy**, **information literacy**, etc.
 - Identifying similar words using word embeddings: **e-literacy**, **technology proficiency**, etc.

Word Embeddings

Why useful?

- Automatic generalization (cont'd)
 - Dictionaries combined with word embeddings ([Garten et al. 2018](#))
 - [Osnabrugge et al. \(2021\)](#): measuring emotive rhetoric from political speech
$$\text{AvgSim}_{\text{emotive}}(w) = \frac{1}{N_{\text{emotive}}} \sum_{i=1}^{N_{\text{emotive}}} \cos(\vec{w}, \vec{e}_i)$$
$$\text{AvgSim}_{\text{neural}}(w) = \frac{1}{N_{\text{neural}}} \sum_{i=1}^{N_{\text{neural}}} \cos(\vec{w}, \vec{e}_i)$$
$$\text{Emotiveness}(w) = \text{AvgSim}_{\text{emotive}}(w) - \text{AvgSim}_{\text{neural}}(w)$$

Estimating Word Embeddings

Word2Vec ([Mikolov et al. 2013a](#); [Mikolov et al. 2013b](#))

- Skipgram and CBOW (Continuous Bag Of Words)
 - Skipgram: given a target word, predict the context words (e.g., ± 5)
 - CBOW: given the context words, predicts the target word
- SGNS (skip-gram with negative sampling)
 - Given a pair of a target word and another word c , what is the probability of c being the actual context word c_{pos} ?

Estimating Word Embeddings

Word2Vec SGNS

- Self-supervision: “+” if in context, otherwise “-”
 - $\backslash(L\backslash$): the size of the context window
 - $\backslash(K\backslash$): the proportion of positive (or context) to (randomly selected) negative examples (recommended $\backslash(K\backslash$): 2–5 for big, 5–20 for small data)

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

c1 c2 w c3 c4

positive examples +

w	c _{pos}
apricot	tablespoon
apricot	of
apricot	jam
apricot	a

negative examples -

w	c _{neg}	w	c _{neg}
apricot	aardvark	apricot	seven
apricot	my	apricot	forever
apricot	where	apricot	dear
apricot	coaxial	apricot	if

Estimating Word Embeddings

Word2Vec SGNS

- Task
 - Train a binary classifier that computes $\Pr(+ | (w, c))$
 - $\Pr(+ | (w, c)) = \frac{1}{\sigma(\vec{e}_w \cdot \vec{e}_c)}$
- Goal
 - Maximize the similarity of the target-context pairs (w, c_{pos})
 - Minimize the similarity of the target-non-context pairs (w, c_{neg})

Estimating Word Embeddings

Word2Vec SGNS

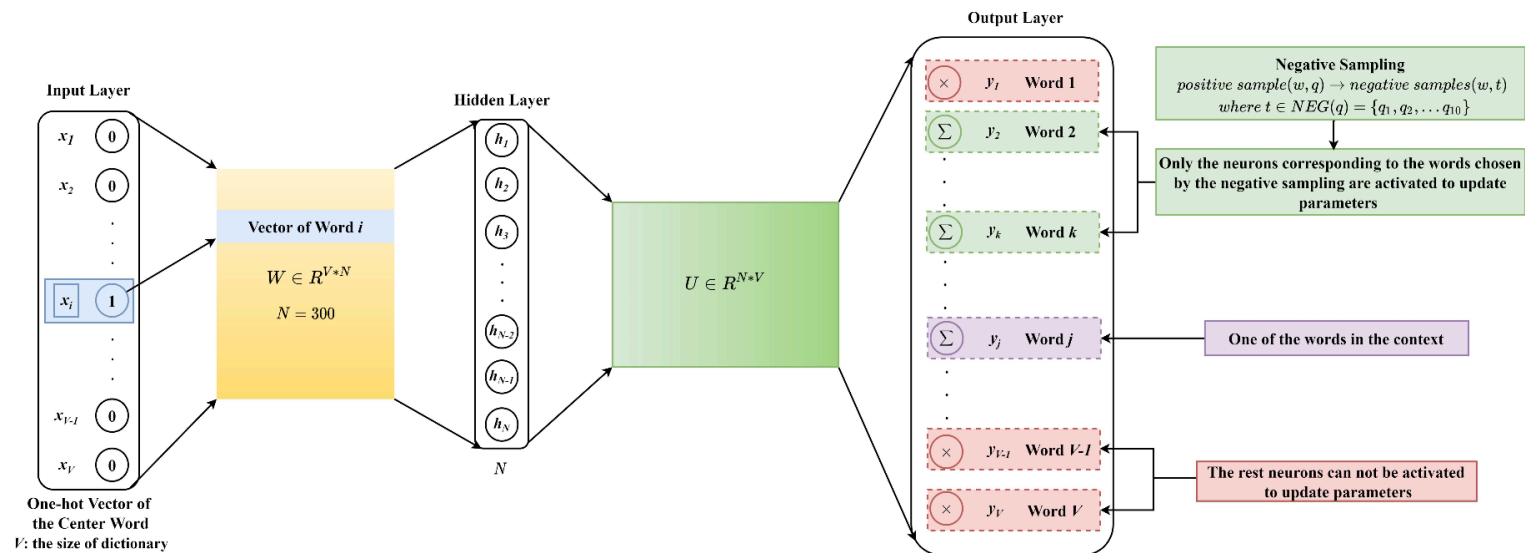
- Optimization: minimize the cross-entropy loss function using (stochastic) gradient descent

$$L_{CE} = -\log[P(+|w, c_{pos}) \prod_{i=1}^k P(-|w, c_{neg_i})]$$

Estimating Word Embeddings

Word2Vec SGNS

- The neural network for SG(NS) (source: [link](#))



Estimating Word Embeddings

Word2Vec SGNS

- Detailed treatments of SG and SGNS
 - SG: [link](#)
 - SGNS: [link](#)

Various Approaches

Different approaches to obtaining word embeddings

- GloVe (Global Vectors for Word Representation) ([Pennington et al. 2014](#))
- FastText ([Bojanowski et al. 2017](#))
 - Subword-level model
 - Each word is represented as itself along with a bag of constituent n-grams, with boundary symbols `<` and `>`
 - E.g., $\vec{e}_{\text{apple}} = \vec{e}_{\text{<ap}} + \vec{e}_{\text{app}} + \vec{e}_{\text{ppl}} + \vec{e}_{\text{ple}} + \vec{e}_{\text{le>}} + \vec{e}_{\text{<apple>}}$
 - Deals with OOV (out of vocabulary), rare words, and typos (e.g., [appple](#)) efficiently

Pre-trained Embeddings

General embeddings

- Word2Vec: [link](#) (“GoogleNews-vectors-negative300.bin.gz”)
- GloVe: [link](#)
- FastText: [link](#)

(A few examples from many) domain-specific embeddings

- Trained on 19th-century British newspapers: [link](#)
- Trained on tweets: [link](#) (“glove.twitter.27B.zip”)

Static vs. Contextual Embeddings

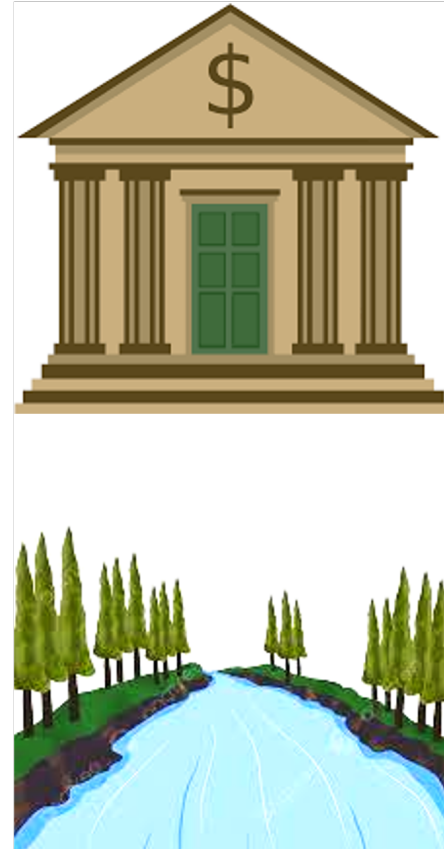
From contexts but not contextual

- Word2Vec, GloVe, and FastText are *static* embeddings
- A word's embedding *is* derived from its context
- Different contexts do *not* lead to different embeddings
- However, a word's meaning differs depending on the context even if a word has the same form

Static vs. Contextual Embeddings

Example

- Embeddings for the same word differ by context
- Sentence 1: *Open a **bank** account*
- Sentence 2: *On the river **bank***



Static vs. Contextual Embeddings

Modern contextual embeddings are based on neural network models with **self-attention**

- The primary goal of self-attention is to compute contextualized representations of each token in the sequence
- Self-attention allows each token in the sequence to ‘attend’ to (or reference) all other parts of the sequence (including self)
- This allows for capturing contextual meanings of tokens
 - E.g., “The **dog** in the yard started to bark because **it** was hungry”

Static vs. Contextual Embeddings

Previous example

- *Open a **bank** account* \rightarrow $(e_{\text{bank}_{s1}})$: [0.3, 0.9, ...]
- *On the river **bank*** \rightarrow $(e_{\text{bank}_{s2}})$: [0.8, 0.1, ...]
- $(e_{\text{bank}_{s1}})$ should be similar to $(e_{\text{account}_{s1}})$, and $(e_{\text{bank}_{s2}})$ should be similar to $(e_{\text{river}_{s2}})$

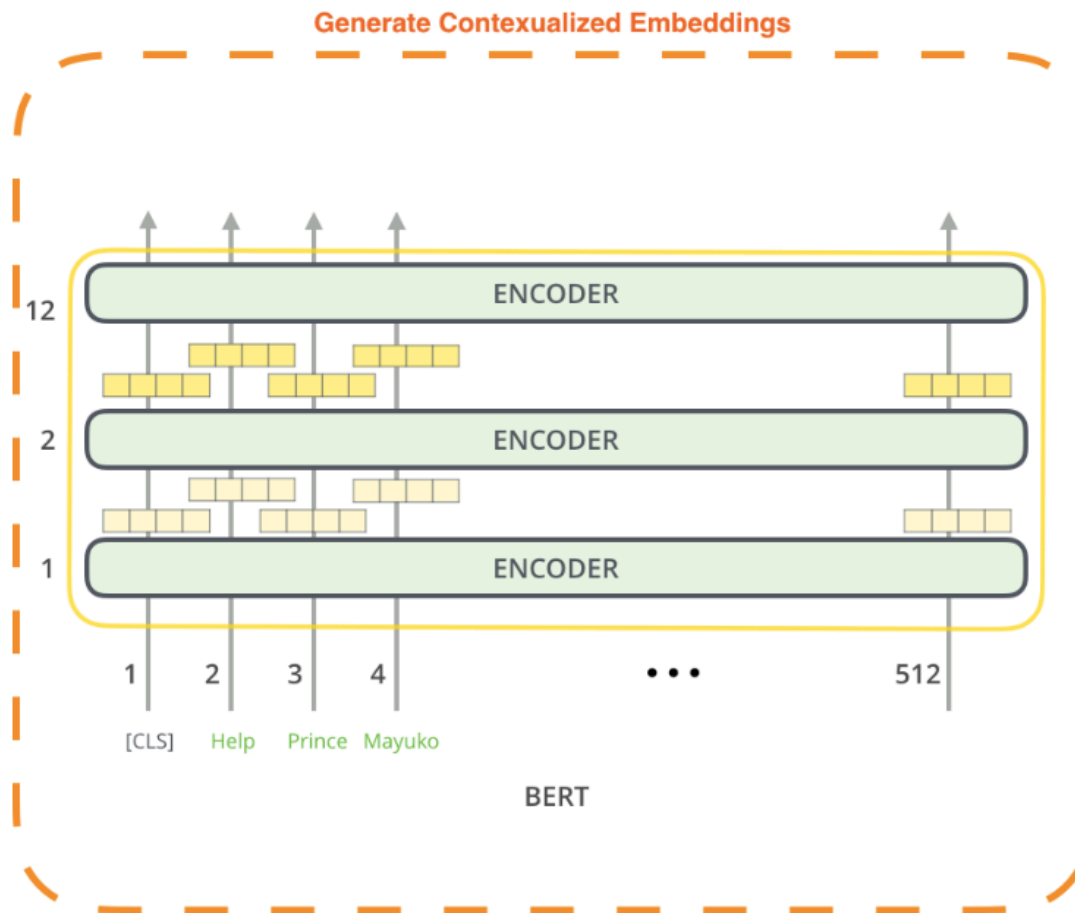
Static vs. Contextual Embeddings

Bi-directional Encoder Representations from Transformers

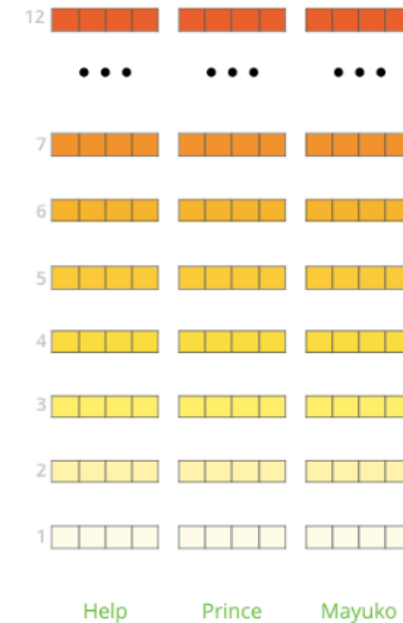
- A form of transformer (a type of neural network architecture with self-attention mechanisms)
- An encoder model (processing an input sequence and transforms it into an embedding)
- (Jointly) trained on huge data sets ([BookCorpus](#) and Wikipedia) for two tasks: masked language modeling & next sentence prediction
- Two versions of the model introduced in [the original paper](#)
 - BERT BASE (12 encoder stacks)
 - BERT LARGE (24 encoder stacks)

Static vs. Contextual Embeddings

Bi-directional Encoder Representations from Transformers



The output of each encoder layer along each token's path can be used as a feature representing that token.



But which one should we use?

Static vs. Contextual Embeddings

What to use?

- If your analysis is about words—especially general or out-of-context meanings—static embeddings are often sufficient and computationally efficient
- If your task involves meaning in context or sentence-level analysis (e.g., classification, topic modeling, semantic similarity), contextual embeddings are typically more appropriate.

Bias Reflected in Human Language

Bolukbasi et al. (2016)

- Pretrained Word2Vec embeddings
 - E.g., ‘computer programmer’ - ‘man’ + ‘woman’ = ‘homemaker’

Gender stereotype <i>she-he</i> analogies.		
sewing-carpentry	register-nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	hairdresser-barber
Gender appropriate <i>she-he</i> analogies.		
queen-king	sister-brother	mother-father
waitress-waiter	ovarian cancer-prostate cancer	convent-monastery

Figure 2: **Analogy examples.** Examples of automatically generated analogies for the pair *she-he* using the procedure described in text. For example, the first analogy is interpreted as *she:sewing :: he:carpentry* in the original w2vNEWS embedding. Each automatically generated analogy is evaluated by 10 crowd-workers to whether or not it reflects gender stereotype. Top: illustrative gender stereotypic analogies automatically generated from w2vNEWS, as rated by at least 5 of the 10 crowd-workers. Bottom: illustrative generated gender-appropriate analogies.

Bias Reflected in Human Language

Let's try it in Korean: [link](#)

Summary

- Word embeddings can be used to study word usage/meanings and as feature representations for downstream NLP tasks
- Static embeddings are lightweight and capture general meaning, while contextual embeddings adapt to specific usage and reflect meaning more accurately
- Word embeddings can reflect bias in various aspects

Guided Coding

- [Word2Vec and FastText in Python](#)
- [Exploring BERT](#)