Word Embeddings

HSS 510: NLP for HSS

Taegyoon Kim

Apr 24, 2024

Agenda

Things to be covered

- Word representations
- Word embeddings
- Word2Vec SGNS
- Other models: GloVe, FastText
- Evaluating performance
- Pre-trained vs. self-trained
- Bias reflected in embeddings

Document Representation

We have mostly dealt with document representation

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

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.”

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 words in a multi-dimensional space
- The simplest approach: one-hot encoding
  - A vector with one dimension per unique word (i.e., type) 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

- Semantics
  - Similarity: one-hot(`author`) $\perp$ one-hot(`writer`)
  - Think about the rationale behind lemmatization/stemming: `author` vs. `authors`
- 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., $\pm$ 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/short* vectors representing word meanings in a multi(low)-dimensional space (d = 50–1000)
  - Word embeddings $\subset$ word vectors
- Words are “embedded” into a common low-dimensional space
- 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?

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

Word Embeddings

Why useful?

- A measure of word meaning

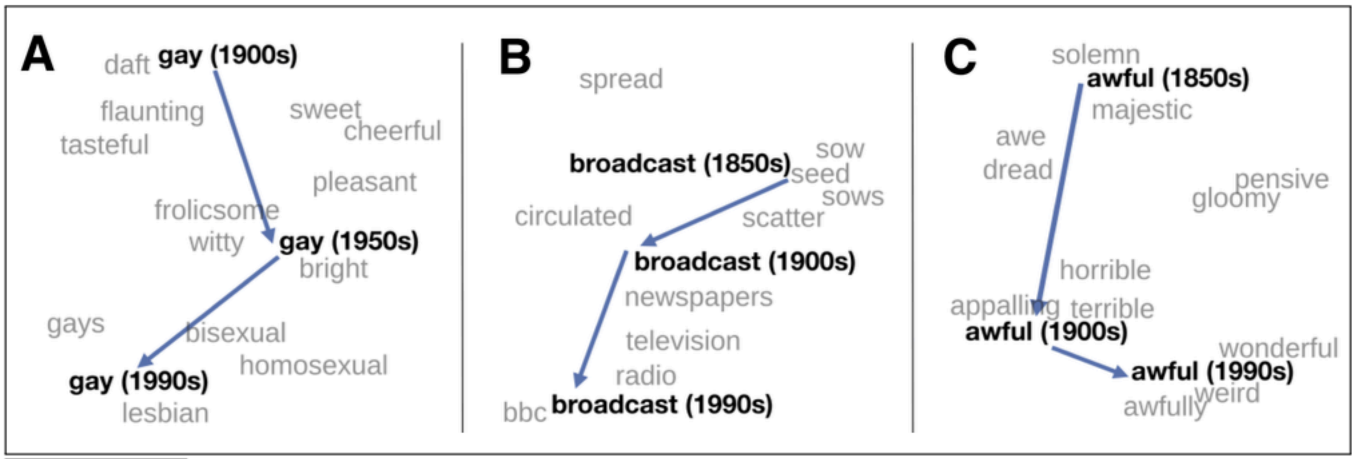
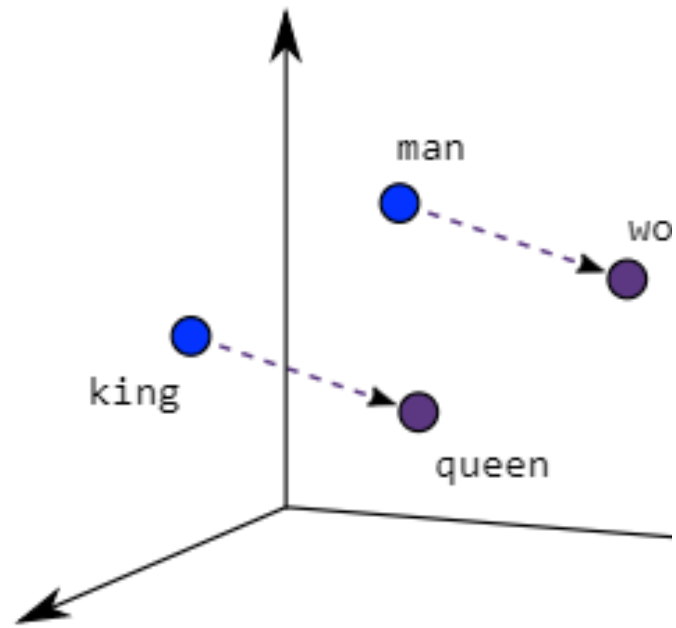


Figure 6.17 A t-SNE visualization of the semantic change of 3 words in English using word2vec vectors. The modern sense of each word, and the grey context words, are computed from the most recent (modern) time-point embedding space. Earlier points are computed from earlier historical embedding spaces. 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 ($\Longleftrightarrow$ one-hot encodings)
    - E.g., $\overrightarrow{e_{author}} \propto \overrightarrow{e_{writer}}$
  - Similarity in relations ("vector arithmetic")
    - E.g., `king` - `man` + `woman` ≈ `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.
    - Dictionaries combined with word embeddings (Garten et al. 2018; Osnabrugge et al. 2021)
        - E.g., keywords for “anger”
        - The centroid of embeddings for terms signalling “anger”
        - The centroid of the embeddings of the words in a document

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., $\pm$ 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 “-”
  - L: the size of the context window
  - K: the proportion of positive (or context) to (randomly selected) negative examples (recommended K: 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 $\Pr(+|w, c)$
  - $\Pr(+|w, c) = \sigma(\overrightarrow{e_w} \cdot \overrightarrow{e_c})$
- 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

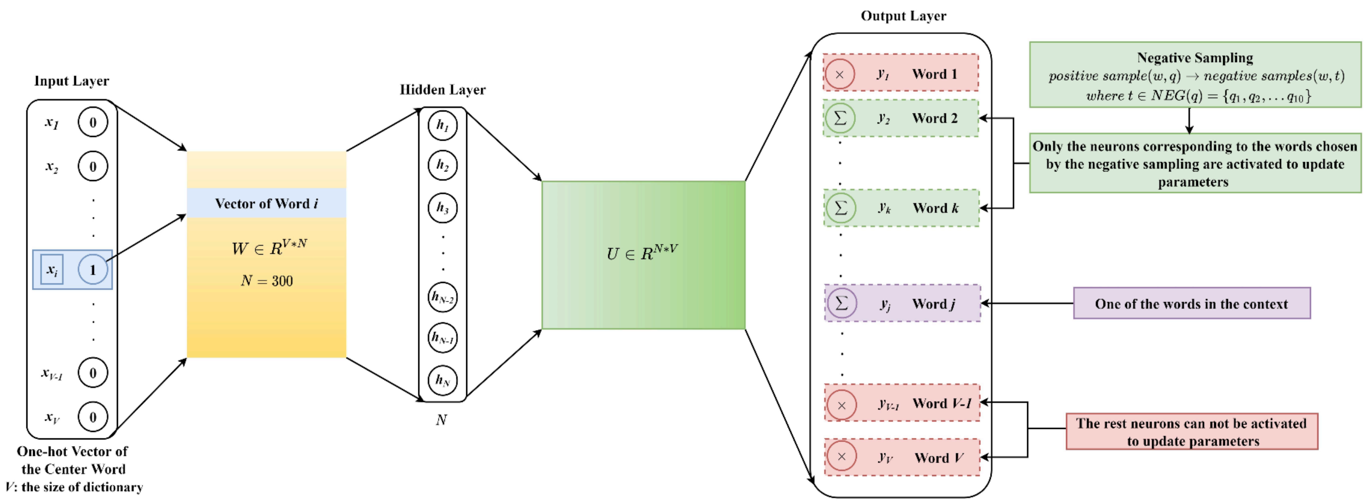
- Optimization: minimize the cross-entropy loss function using (stochastic) gradient descent

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

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

Other Approaches

There are many different approaches how one could obtain 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 n0grams, with boundary symbols $<$ and $>$
    - E.g., $\vec{e_{apple}} = \vec{e_{<ap}} + \vec{e_{app}} + \vec{e_{ppl}} + \vec{e_{ple}} + \vec{e_{le>}} + \vec{e_{<apple>}}$
  - Deals with OOV (out of vocabulary), rare words, and typos (e.g., `appple`) efficiently

Evaluating Performance

How to evaluate word embeddings?

- Extrinsic validation (straightforward)
  - Performance on a downstream NLP task (PoS tagging, NER, etc.)
- Intrinsic validation
  - Whether the embeddings are able to capture similarities between words
  - Computer science as well as social sciences (e.g., Rodriguez and Spirling (2022))

Various 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”)

Pre-trained vs. Self-trained

A few circumstances that require self-training

- Temporal changes in language
  - Known as diachronic/dynamic embeddings (e.g., Kim and Jeon 2023)
- Group-specific language (e.g., Democrats vs. Republicans)
- Domain-specific language (e.g., Case2Vec)
- Low resource languages

Pre-trained vs. Self-trained

Which one captures word similarity better?

- Experiment by Rodriguez and Spirling (2022)
  - Provide crowd workers (human annotators) on MTurk 10 political words and ask them to produce a set of ten nearest neighbors ("human")
  - They then use these same words to generate machine nearest neighbors by finding the most cosine similar vector using word embeddings ("local" or pre-trained "GloVe")
  - They then have a separate set of humans (human judge) look at a prompt word and two possible nearest neighbors ("human" vs. "local" vs. pre-trained "GloVe")

Pre-trained vs. Self-trained

Findings from Rodriguez and Spirling (2022)

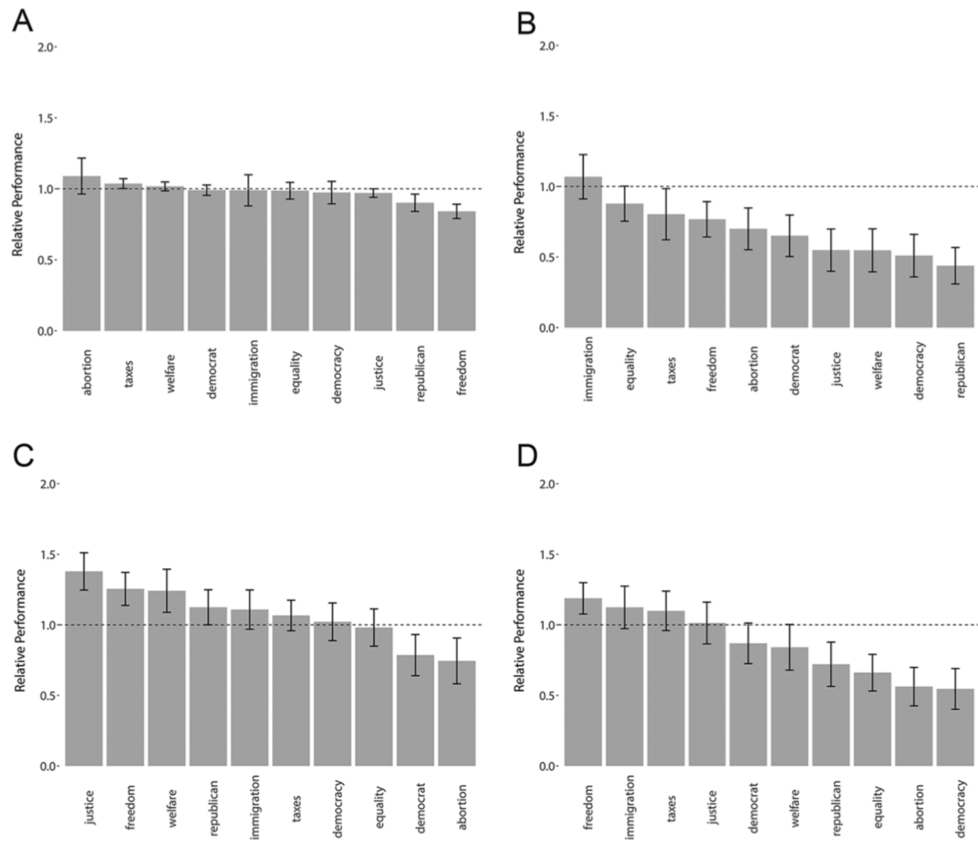


Figure 2. Human preferences: Turing assessment. *A*, Candidate: local 48–300; baseline: local 6–300. *B*, Candidate: local 6–300; baseline: human. *C*, Candidate: GloVe; baseline: local 6–300. *D*, Candidate: GloVe; baseline: human.

Pre-trained vs. Self-trained

Findings from Rodriguez and Spirling (2022)

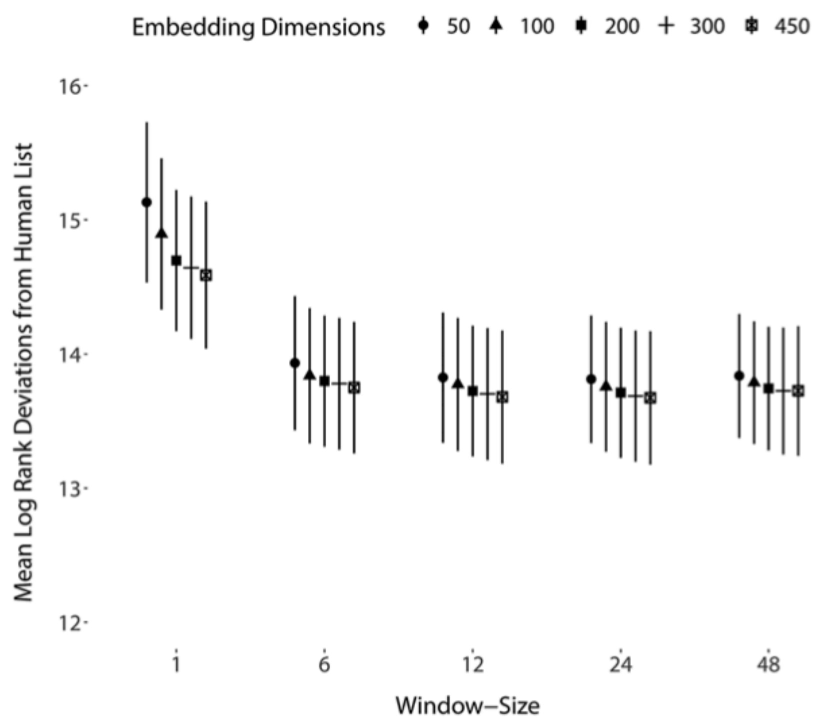


Figure 3. Human preferences: log rank deviations: complex models come closer to "human" assessments, but medium-size models are almost as good as very large ones.

Pre-trained vs. Self-trained

Findings from Rodriguez and Spirling (2022)

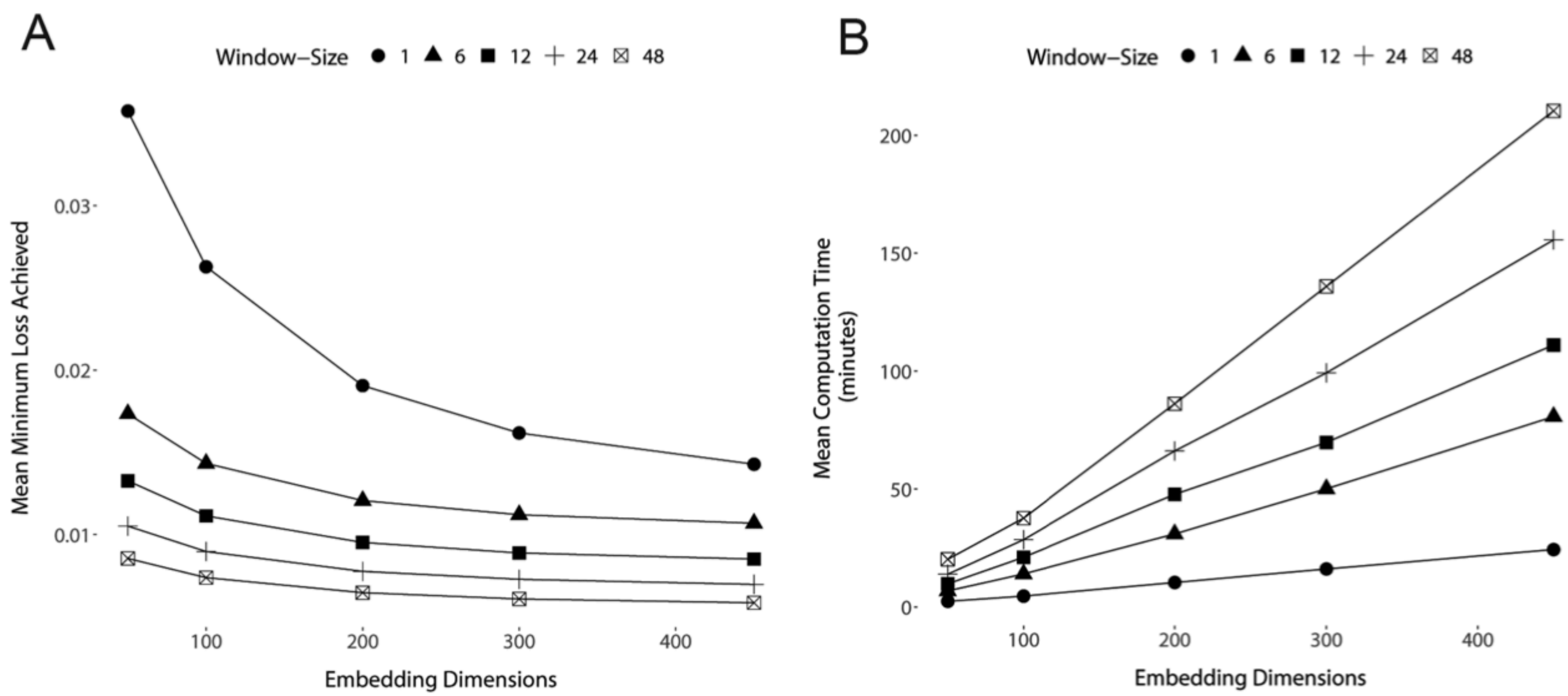


Figure 4. Technical criteria: larger models fit better but take longer to compute. *A*, Mean minimum loss achieved. *B*, Computation time (minutes)

Pre-trained vs. Self-trained

Lessons

- Popular pre-trained word embeddings "perform" at a level close to–or even surpassing–both human annotators and locally fit models with various configurations

Caveat

- A specific meaning of "perform"
- The results are based on a limited set of corpora (political in nature)

Bias Reflected in Human Language

Bolukbasi et al. (2016)

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

Gender stereotype *she-he* 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 *she-he* 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 are 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.

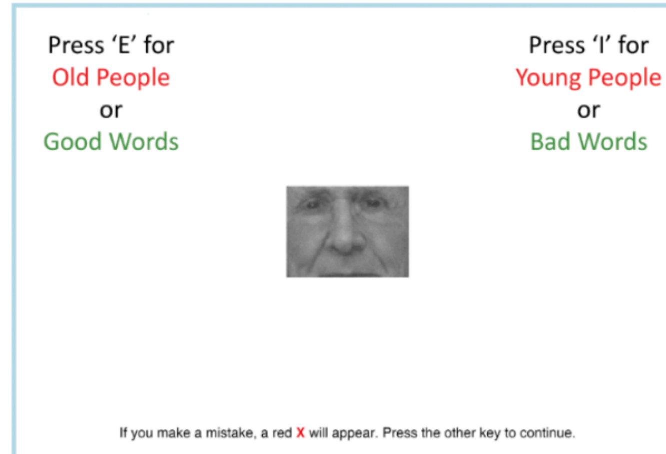
Bias Reflected in Human Language

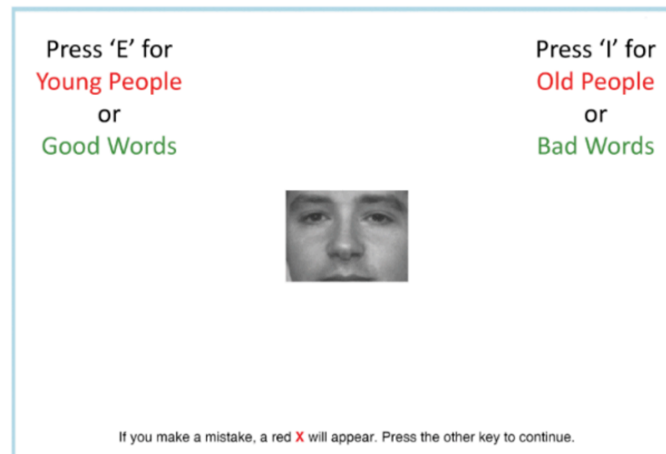
Let's try it in Korean: link

Bias Reflected in Human Language

Caliskan et al. (2017)

- Replicated evidence of bias from IATs (Implicit Association Test) using pre-trained GloVe vectors and cosine similarity
- African American (European-American) names have higher cosine similarity with unpleasant (pleasant) words





Summary

- Word embeddings can be used
  - Not only for downstream NLP tasks
  - But also for studying word usage/meanings
- Popular pre-trained embeddings appear to match (or outperform) locally trained embeddings in terms of capturing word similarity
- Word embeddings reflect bias in various aspects

Guided Coding

Training Word2Vec and FastText in Python