Vector Semantics and Embeddings

Makrai Márton

Számítógépes nyelvészet @ tlp 2022 tavasz

Vector semantics and embeddings

- szabályalapú NLP és gépi tanulás
- ch 6 in Speech and Language Processing. Daniel Jurafsky & James H. Martin. 2021
- the distributional hypothesis

Vector semantics and embeddings

- szabályalapú NLP és gépi tanulás
- ch 6 in Speech and Language Processing. Daniel Jurafsky & James H. Martin. 2021
- the distributional hypothesis
  - Words that occur in similar contexts tend to have similar meanings.
  - first formulated by linguists (Joos, 1950; Harris, 1954; Firth, 1957)
- vector semantics instantiates this linguistic hypothesis
  - learning representations of the meaning of words, called embeddings, directly from their distributions in texts
- used in every computational semantic application
- static embeddings (2013, 2014)
- dynamic or contextualized embeddings like BERT (2018, 2019, következő óra)

Representation learning

Representation learning

- self-supervised ways to learn representations of the input,
- instead of creating representations by hand via feature engineering
- an important focus of NLP research (Bengio, Courville, and Vincent, 2013)

Áttekintés

Lexical semantics

- How to represent the meaning of a word?
  - n-gram models: a string of letters, or an index in a vocabulary list
  - not that different from a tradition in philosophy
    - just spelling the word with small capital letters; representing the meaning of “dog” as DOG, or by using an apostrophe (DOG’)
- lexical semantics
  - a model of word meaning
  - similar meanings (*cat* is similar to *dog*),
    antonyms (*cold* is the opposite of *hot*), connotations (*happy* is +, *sad* is -)
  - *buy*, *sell*, and *pay* offer differing perspectives on the same event
  - draw inferences to address meaning-related tasks like question-answering or dialogue

Lemmas and Senses

- how *mouse* is defined in a dictionary
  (simplified from on online dictionary)

  mouse (N)
    - *any of numerous small rodents. . .*
    - *a hand-operated device that controls a cursor. . .*
- lemma, also called the citation form
  - inflected forms like *mice*
  - In many languages the infinitive form is used as the lemma for the verb
    - Spanish *dormir* "to sleep" is the lemma for *duermes* "you sleep"
  - wordforms
- each lemma can have multiple meanings; a word sense
  - polysemy, WSD

Synonymy

- eg *couch/sofa vomit/throw up filbert/hazelnut car/automobile*
- propositional meaning

Synonymy

- eg *couch/sofa vomit/throw up filbert/hazelnut car/automobile*
- propositional meaning
  - A more formal definition of synonymy (between words rather than senses)
  - if they are substitutable for one another in any sentence without changing the truth conditions of the sentence
- principle of contrast (Girard 1718, Bréal 1897, Clark 1987)

Synonymy

- eg *couch/sofa vomit/throw up filbert/hazelnut car/automobile*
- propositional meaning
  - A more formal definition of synonymy (between words rather than senses)
  - if they are substitutable for one another in any sentence without changing the truth conditions of the sentence
- principle of contrast (Girard 1718, Bréal 1897, Clark 1987)
  - no two words are absolutely identical in meaning
    - difference in linguistic form is always associated with diff in meaning
  - For example, the word H_2O is used in scientific contexts
  - In practice, the word synonym is therefore used to describe a relationship of approximate or rough synonymy

Word Similarity

- words don't have many synonyms,
  most words do have lots of similar words
- *cat* is not a synonym of *dog*, but certainly similar words
- shift from talking about relations between word senses (eg synonymy) to relations between words (like similarity)
  - no commitment to a particular representation of word senses
- in larger semantic tasks
  - how similar the meaning of two phrases or sentences are
  - question answering, paraphrasing, and summarization
  - getting values for word similarity $\leftarrow$ ask humans to judge similarity
  - datasets from such experiments
    - SimLex-999 dataset (Hill, Reichart, and Korhonen, 2015)
      values on a scale from 0 to 10

Word Relatedness

- related in ways other than similarity
- word relatedness (Budanitsky and Hirst, 2006)
  - called word association in psychology
- eg *coffee* and *cup*

Word Relatedness

- related in ways other than similarity
- word relatedness (Budanitsky and Hirst, 2006)
  - called word association in psychology
- eg *coffee* and *cup*
  - practically no shared features (e.g. *coffee* is a plant)
  - associated by co-participating in the everyday event of drinking
- Similarly *scalpel* and *surgeon* [*szike* és *sebész*]
- semantic fields
  - One common kind of relatedness: if w_1 and w_2 belong to the same $\sim$
  - $:=$ a set of words which cover a particular semantic domain
  - structured relations with each other
  - eg the semantic field of hospitals *(surgeon, scalpel, nurse, anesthetic)*
  - related to topic models, like Latent Dirichlet Allocation, LDA,
    - topic models apply unsupervised learning on large sets of texts to induce sets of associated words from text
    - for discovering topical structure in documents
- more relations between senses like hypernymy or IS-A, antonymy

Word Relatedness

- related in ways other than similarity
- word relatedness (Budanitsky and Hirst, 2006)
  - called word association in psychology
- eg *coffee* and *cup*
  - practically no shared features (e.g. *coffee* is a plant)
  - associated by co-participating in the everyday event of drinking
- Similarly *scalpel* and *surgeon* [*szike* és *sebész*]
- semantic fields
  - One common kind of relatedness: if w_1 and w_2 belong to the same $\sim$
  - $:=$ a set of words which cover a particular semantic domain
  - structured relations with each other
  - eg the semantic field of hospitals *(surgeon, scalpel, nurse, anesthetic)*
  - related to topic models, like Latent Dirichlet Allocation, LDA,
    - topic models apply unsupervised learning on large sets of texts to induce sets of associated words from text
    - for discovering topical structure in documents
- more relations between senses like hypernymy or IS-A, antonymy (opposites) and meronymy

Word Relatedness

- related in ways other than similarity
- word relatedness (Budanitsky and Hirst, 2006)
  - called word association in psychology
- eg *coffee* and *cup*
  - practically no shared features (e.g. *coffee* is a plant)
  - associated by co-participating in the everyday event of drinking
- Similarly *scalpel* and *surgeon* [*szike* és *sebész*]
- semantic fields
  - One common kind of relatedness: if w_1 and w_2 belong to the same $\sim$
  - $:=$ a set of words which cover a particular semantic domain
  - structured relations with each other
  - eg the semantic field of hospitals *(surgeon, scalpel, nurse, anesthetic)*
  - related to topic models, like Latent Dirichlet Allocation, LDA,
    - topic models apply unsupervised learning on large sets of texts to induce sets of associated words from text
    - for discovering topical structure in documents
- more relations between senses like hypernymy or IS-A, antonymy (opposites) and meronymy (part-whole relations)

- *buy*
- *sell*
- *pay*

Semantic Frames and Roles

- Closely related to semantic fields
- A semantic frame is a set of words that denote perspectives or participants in a particular type of event
- eg A commercial transaction
  - a kind of event in which one entity trades money to another entity in return for some good or service, after which
    the good changes hands or perhaps the service is performed
  - encoded lexically by verbs like
    - *buy* (event from the perspect of buyer)
    - *sell* (from the perspective of the seller)
    - *pay* (focusing on the monetary aspect)
    - nouns like *buyer*
- Frames have semantic roles (like *buyer, seller, goods, money*)
- makes it possible for a system to know paraphrases
  eg *Sam bought the book from Ling* ≈ *Ling sold the book to Sam*
  - Sam has the role of the buyer in the frame and Ling the seller
  - important for question answering, and can help in machine translation

Connotation, affective meanings

- connotation
  - the term has different meanings in different fields,
  - here we use it to mean
    the aspects of a word's meaning that are related to a writer or reader's emotions, sentiment, opinions, or evaluations
- positive connotations *(happy)*, negative connotations *(sad)*

copy, replica, reproduction

- sentiment
- in tasks like sentiment analysis, stance detection, and applications of NLP to politics and consumer reviews

Connotation, affective meanings

- connotation
  - the term has different meanings in different fields,
  - here we use it to mean
    the aspects of a word's meaning that are related to a writer or reader's emotions, sentiment, opinions, or evaluations
- positive connotations *(happy)*, negative connotations *(sad)*
- Even words whose meanings are similar can vary in connotation;
  - eg *fake, knockoff, forgery* $\leftrightarrow$ *copy, replica, reproduction*

- sentiment
- in tasks like sentiment analysis, stance detection, and applications of NLP to politics and consumer reviews

Connotation, affective meanings

- connotation
  - the term has different meanings in different fields,
  - here we use it to mean
    the aspects of a word's meaning that are related to a writer or reader's emotions, sentiment, opinions, or evaluations
- positive connotations *(happy)*, negative connotations *(sad)*
- Even words whose meanings are similar can vary in connotation;
  - eg *fake, knockoff, forgery* ↔ *copy, replica, reproduction*
  - *innocent* (positive) vs *naive* (negative)
- sentiment
- in tasks like sentiment analysis, stance detection, and applications of NLP to politics and consumer reviews

Three dimensions of affective meaning

Osgood, Suci, and Tannenbaum (1957)

- words vary along three dimensions of affective meaning

Three dimensions of affective meaning

Osgood, Suci, and Tannenbaum (1957)

- words vary along three dimensions of affective meaning
    - valence: the pleasantness of the stimulus
    - arousal: the intensity of emotion provoked by the stimulus
    - dominance: the degree of control exerted by the stimulus
- Each word is thus represented by three numbers
- Osgood, Suci, and Tannenbaum (1957) noticed that in using these 3 numbers to represent the meaning the model was representing each word as a point in a three-dimensional space,

Áttekintés

Vector semantics I

- history in the 1950s: two big ideas converged
  - Osgood, Suci, and Tannenbaum (1957)'s idea:
    use a point in three-dimensional space to repr connotation
  - linguists like Joos (1950), Harris (1954), and Firth (1957):
    - define the meaning of a word by its distribution in language use, meaning its neighboring words or grammatical environments
    - two words that occur in very similar distributions (whose neighboring words are similar) have similar meanings
- vector semantics
  - represent a word as a point in a multidimensional semantic space
  - space is derived from the distributions of word neighbors
- Vectors for representing words are called embeddings
  - more strictly applied only to dense vectors like word2vec, rather than sparse tf-idf or PPMI vectors
  - etim: from its mathematical sense as a mapping from one space or structure to another, although the meaning has shifted

Vector semantics II

- The fine-grained model of word similarity of vector sem is powerful in
  - pre-neural sentiment classifiers depend on the same words appearing in the training and test sets
  - But by representing words as embeddings, classifiers can assign sentiment as long as it sees some words with similar meanings
  - can be learned automatically from text without supervision
- we'll introduce the two most commonly used models
  - tf-idf model, an important baseline, the meaning of a word is defined by a simple function of the counts of nearby words
    - very long vectors that are sparse, ie mostly zeros
  - word2vec model family: short, dense vectors that have useful semantic properties
  - cosine, the standard way to use embeddings to compute semantic similarity, between two words, two sentences, or two documents
  - applications like question answering, summarization, or aut essay grading

Vectors and documents I

- Vector or distributional models of meaning are generally based on a co-occurrence matrix, a way of representing how often words co-occur
- the term-document matrix and the term-term matrix
- term-document matrix
  - row represents a word in the vocabulary and
    each column represents a document from some collection of documents
  - Each cell in this matrix represents the number of times word occurs in doc
  - first defined as part of the vector space model of information retrieval (Salton, 1971). In this model, a document is represented as a count vector

Dimensions

- linear algebra, a vector is a

Dimensions

- linear algebra, a vector is a list or array of numbers
- dimensions
  - in real term-document matrices, the vectors representing each document would have dimensionality |V|, the vocabulary size
  - meaningful dimensions on which documents vary
  - we can compare each dimension
  - vector for a document as a point in |V|-dimensional space
  - vocabulary sizes are generally in the tens of thousands
  - # documents can be enormous (think about all the pages on the web)
- hard to visualize, Fig. 6.4 shows a visualization in two dimensions
- D columns (one for each document in the collection)

Information retrieval

- originally for finding similar documents:
  document information retrieval
- Two documents that are similar will tend to have similar words $\leftrightarrow$ column vectors will tend to be similar
- Information retrieval (IR)
  - the task of finding the document that best matches a query
- also represent a query by a vector, also of length $|V|$
- compare two vectors to find how similar they are
- the tf-idf term weighting, and the cosine similarity metric
- efficient ways to store and manipulate these vectors by exploiting sparsity

Words as vectors: document dimensions

- associating each word with a word vector, a row vector
- similar words have similar vectors
  because they tend to occur in similar docs

Words as vectors: word dimensions

- term-term matrix, aka word-word matrix or the term-context matrix
    - the columns are labeled by words rather than documents
    - dimensionality $|V| \times |V|$ and
    - each cell records the number of times the row (target) word and the column (context) word co-occur in some context in the training corpus
    - The context could be the
        - document, most common
        - smaller contexts, generally a window around eg 4 words to the left and 4 words to the right
- *cherry* and *strawberry* are more similar to each other than they are to *digital*
- |V|, the dimensionality of the vector, the size of the vocab, often between 10,000 and 50,000 words (using the most frequent words)
    - keeping words after about the most frequent 50,000 is not helpful
- sparse vector representations
    - there are efficient algorithms for storing and computing with sparse matrices

Áttekintés

Similarly measure

- takes two vectors of the same dimensionality
  - either both with words as dimensions, hence of length |V|
  - or both with documents as dimensions, of length |D| and
- gives a measure of their similarity

Cosine similarity

- By far the most common similarity metric is the cosine of the angle
- based on the dot product operator from linear algebra
    - aka inner product
- like most measures for vector similarity used in NLP
- dot product $(v, w) = v \cdot w =$
- similarity metric
    - high just when the two vectors have large values in the same dimensions.
    - vectors that have non-zeros in different dimensions: orthogonal vectors have a dot product of 0, representing their strong dissim

Cosine similarity

- By far the most common similarity metric is the cosine of the angle
- based on the dot product operator from linear algebra
    - aka inner product
- like most measures for vector similarity used in NLP
- dot product $(v, w) = v \cdot w = \sum_{i=1}^{N} v_i w_i$
- similarity metric
    - high just when the two vectors have large values in the same dimensions.
    - vectors that have non-zeros in different dimensions: orthogonal vectors have a dot product of 0, representing their strong dissim

Long vectors

- raw dot product favors long vectors
- vector length is defined as $|v| = \sqrt{\sum v_i^2}$
- More frequent words have longer vectors $\rightarrow$ raw dot product higher
- we want a similarity metric regardless of freq
- normalize for the vector length by
  dividing the dot product by the lengths
- the same as the cosine of the angle between the two vectors, following from

$$a \cdot b = |a||b|cos\theta \tag{1}$$

$$\frac{a \cdot b}{|a||b|} = cos\theta \tag{2}$$

- For some applications we pre-normalize each vector
  - For unit vectors, the dot product is the same as the cosine

Range

- The cosine value ranges
  - 1 for vectors pointing in the same direction, through
    0 for orthogonal vectors,
    -1 for vectors pointing in opposite directions
  - But since raw frequency values are non-negative, the cosine for these vectors ranges 0–1
- When two vectors are more similar, the cosine is larger but the angle is small

Áttekintés

tf-idf és PPMI

- we're not going to get good discrimination from words like *the, it*
  - they occur frequently with all sorts of words and aren't informative
- paradox
  - Words that occur nearby frequently (maybe pie nearby cherry) are more important than words that only appear once or twice. Yet
  - words that are too frequent—ubiquitous, like *the* or *good*— are unimportant
  - balance these two conflicting constraints?
- two common solutions to this problem
  - tf-idf weighting, usually used when the dimensions are documents
  - PPMI algorithm (usually used when the dimensions are words)

tf-idf

- The tf-idf weighting (the '-' here is a hyphen, not a minus sign)
  - the product of two terms
- term frequency (Luhn, 1957):
  the frequency of the word t in the document d
  - raw count as the term frequency: $tf_{t,d} = \quad count(t, d)$
- to give a higher weight to words that occur only in a few documents
  - these words are useful for discriminating those documents from the rest
  - document frequency df_t of a term t is
    the number of documents it occurs in
  - inverse document frequency or *idf* term weight (Sparck Jones, 1972)
  - the fraction N/df_t, where N is the total number of documents
- log(), Because of the large number of documents in many collections
- $w_{t,d} = tf_{t,d} \cdot idf_t$

tf-idf

- The tf-idf weighting (the ‘-’ here is a hyphen, not a minus sign)
  - the product of two terms
- term frequency (Luhn, 1957):
  the frequency of the word t in the document d
  - raw count as the term frequency: $tf_{t,d} = \log_{10} count(t,d)$
  - $\log_{10}$ of the frequency
    - we can't take the log of 0 *Rightarrow* we normally add 1 to the count
- to give a higher weight to words that occur only in a few documents
  - these words are useful for discriminating those documents from the rest
  - document frequency df_t of a term t is
    the number of documents it occurs in
  - inverse document frequency or *idf* term weight (Sparck Jones, 1972)
  - the fraction N/df_t, where N is the total number of documents
- log(), Because of the large number of documents in many collections
- $w_{t,d} = tf_{t,d} \cdot idf_t$

tf-idf: applications, documents

baseline

tf-idf: applications, documents

- applications of the tf-idf weighting of co-occurrence matrices
    - information retrieval
    - a great baseline, the simple thing to try first
- what counts as a document
    - It's usually clear: in Shakespeare we would use a play
    - encyclopedia articles like Wikipedia, the document is a Wikipedia page
    - newspaper articles, the document is a single article
    - Occasionally your corpus might not have document divisions
        - break up the corpus into documents yourself for computing idf

PMI I

- PPMI (positive pointwise mutual information)
- used for term-term-matrices
  - the vector dimensions correspond to words
- intuition: how much more the two words co-occur in our corpus than we would have a priori expected them to appear by chance
- Pointwise mutual information (Fano, 1961)
  - one of the most important concepts in NLP
  - how often two events x and y occur, compared with what we would expect if they were independent: $I(x,y) = \log_2 P(x,y)/P(x)P(y)$
- PMI between a target word w and a context word c (Church and Hanks 1989, Church and Hanks (1990))

$$PMI(w,c) = \log_2 \frac{P(w,c)}{P(w)P(c)}$$

- we compute probability by using the maximum likelihood estimate

PMI II

- PMI values range from negative to positive infinity
  - negative PMI values tend to be unreliable
  - To distinguish whether two words whose individual probability is each 10^{-6} occur together less often than chance, we would need…
  - not clear whether it's even possible to evaluate such scores of ‘unrelatedness’ with human judgments
  - $\rightarrow$ Positive PMI (called PPMI) replaces all negative PMI values with zero (Church and Hanks 1989, Dagan+ 1993, Niwa and Nitta (1994))
  - Positive PMI also cleanly solves the problem of what to do with log(0)

$$PPMI(w, c) = \max(\log_2 P(w, c)/P(w)P(c), 0)$$

- a co-occurrence matrix can be turned into a PPMI matrix

$$p_i j, p_i *, p_* j$$

PMI III

- problem: PMI is biased toward infrequent events; very rare words
  - One way to reduce this bias toward low frequency events: $P_\alpha(c)$ that raises the probability of the context word to the power of α:

$$PPMI_\alpha(w,c) = \max(\log_2 P(w,c)/P(w)P_\alpha(c), 0) \tag{3}$$

$$P_\alpha(c) = count(c)^\alpha / \sum_{c'} count(c')^\alpha \tag{4}$$

  - Levy et al. (2015): a setting of $\alpha = 0.75$ improved performance of on a wide range of tasks (drawing on a similar weighting used for skipgram)
- This works because raising the count to $\alpha = 0.75$ increases the probability assigned to rare contexts, and hence lowers their PMI ($P_\alpha(c) > P(c)$ when c is rare, see 6.8.2)
- Another possible solution is Laplace smoothing
  - Before computing PMI, a small constant k (values of 0.1-3 are common) is added to the counts,
    shrinking (discounting) all the non-zero values
  - The larger the k, the more the non-zero counts are discounted

Summary: the vector semantics model
(aka the tf-idf model or the PPMI mode)

- repr target word as a vector with dimensions corresponding to
  - the documents in a large collection (the term-document matrix) or
  - the counts of words in some neighboring window (the term-term matrix)
- The values in each dimension are counts, weighted by tf-idf (for term-document matrices) or PPMI (for term-term mxs)
- the vectors are sparse (since most values are zero)
- similarity between two words x and y by taking the cosine
  - high cosine, high similarity

Applications of “count-based” vector models

- used eg for deciding if two documents are similar
  - represent a document by taking the vectors of all the words in the document, and computing the centroid of all those vectors
  - The centroid has the minimum sum of squared distances to the vectors
  - applications: information retrieval, plagiarism detection, news recommender systems, and even digital humanities tasks like comparing different versions of a text
- word-level tasks
  - finding word paraphrases, tracking changes in word meaning, or automatically discovering meanings of words in different corpora
  - eg find the 10 most similar words to any target word w by computing the cos

Áttekintés

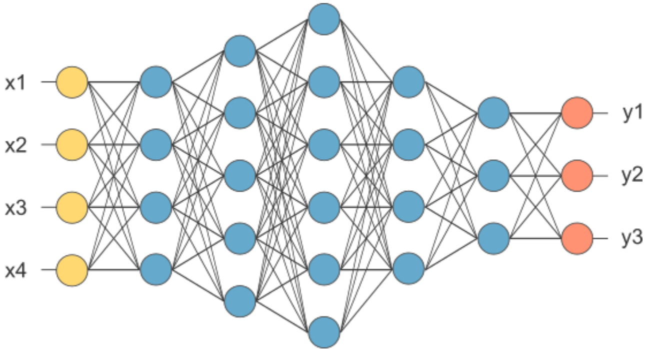
Word2vec I

- embeddings, short dense vectors
  - number of dimensions *d* ranging from 50-1000
  - These *d* dimensions don't have a clear interpretation
  - dense: real-valued numbers that can be negative
  - work better in every NLP task than sparse vectors
  - we don't completely understand all the reasons for this
    - requires our classifiers to learn far fewer weights
    - helps with generalization and avoiding overfitting
- skip-gram with negative sampling aka SGNS
  - one of two algorithms in a software package called word2vec
  - the algorithm is loosely referred to as word2vec (Mikolov+ 2013a,b)
- word2vec methods are fast, efficient to train, and
  easily available online with code and pretrained embeddings

Word2vec II

- static embeddings:
  one fixed embedding for each word in the vocabulary
  - a jövő héten we'll introduce dynamic contextual embeddings like BERT family
    - the vector is different in different contexts
- intuition of word2vec
  - instead of counting how often each word w occurs near, say, *apricot*
  - train a classifier on a binary prediction task: "Is word w likely to show up near *apricot*?"
- we'll take the learned classifier weights as the word embeddings

Artificial neural networks



- cybernetics (1949), connectionism (1974), deep learning (2006)
- Learning features, more and more abstract layers
-
  - computer vision (Krizhevsky and Sutskever, 2012)
  - speech recognition (Hinton et al., 2012)
- fast learning on the graphics card
- like in the brain?

Self-supervision, deep language models, intuition

- self-supervision
  - use running text as implicitly supervised train data for such a classifier as gold ‘correct answer’ to the question “Is word c likely to show up near *apricot*?”
  - first proposed in the task of neural language modeling
  - Bengio et al. (2003) and Collobert et al. (2011)
    a neural language model (a neural network that learned to predict the next word from prior words) could just use the next word in running text as its supervision signal, and could be used to learn an embedding representation for each word as part of doing this prediction task
- We’ll see deep neural network language models a következő héten
- word2vec is a much simpler model in two ways
  - task (making it binary classification instead of word prediction)
  - architecture (training a logistic regression classifier instead of a multi-layer neural network with hidden layers that demand more sophisticated training algorithms)
- The intuition of skip-gram is:
  - Treat the target word and a neighboring context word as positive examples

The classifier I

- classifier
  - given a tuple (w, c) of a target word w paired with a candidate context word c
  - return the probability that c is a real context word
- $P(-|w, c) = 1 - P(+|w, c)$
- How does the classifier compute the probability P?
  - base this probability on embedding similarity
    - a word is likely to occur near the target if its embedding vector is similar
  - if they have a high dot product (cosine is just a normalized dot product)
  - Similarity$(w, c) \approx c \cdot w$
  - not a probability, it's just a number ranging from $-\infty$ to ∞
  - To turn the dot product into a probability
    logistic or sigmoid function $\sigma(x)$, the fundamental core of logistic regression

$$\sigma(x) = 1/1 + exp(-x)$$

The classifier II

- $P(+|w,c) = \sigma(c \cdot w) =$
- we'll also need the total probability of the two possible events to sum to 1
- We thus estimate the probability that word c is not a real context word
- $P(-|w,c) = 1 - P(+|w,c) = \sigma(-c \cdot w) = 1/1 + exp(c \cdot w)$
- simplifying assumption that all context words are independent

$$P(+|w, c_{1:L}) = \prod \sigma(c_i \cdot w) \tag{5}$$

$$\log P(+|w, c_{1:L}) = \log \sum \sigma(c_i \cdot w) \tag{6}$$

- two embeddings for each word
  - the word as a target (aka input embedding)/context (and noise aka output)
  - the target matrix and the context matrix could use different vocabularies, but we'll simplify by assuming one shared vocabulary V
  - ie the parameters we need to learn are two matrices W and C, each containing an embedding for every one of the |V| words

Learning skip-gram embeddings I

- input a corpus of text, and a chosen vocabulary size N
- It begins by assigning a random embedding vector for each of the N words, and
- iteratively shift the embedding of each word w to be
  more like the embeddings of words that occur nearby in texts, and
  less like the embeddings of words that don't occur nearby
- negative examples, skipgram with negative sampling (SGNS)
  - k negative examples for 1 positive examples (ratio set by a parameter k)
  - each consisting of the target w plus a noise word c_{neg}
  - A noise word is a random word from the lexicon, constrained to $\neq w$
  - chosen according to their weighted unigram frequency $p_\alpha(w)$
  - in practice it is common to set $\alpha = .75$

$$P_\alpha(w) = count(w)^\alpha / \sum_{w}^{\prime} count(w')^\alpha$$

Learning skip-gram embeddings II

- weighting gives rare noise words slightly higher probability
- for rare words, $P_\alpha(w) > P(w)$
- eg with two events, $P(a) = .99$ and $P(b) = .01$

$$P_\alpha(a) = .97, P_\alpha(b) = .03$$

- learning algorithm is to adjust those embeddings to
  - Maximize the similarity of the target word, context word pairs (w, c_{pos}) drawn from the positive examples
  - Minimize the similarity of the (w, c_{neg}) pairs from the negative examples
  - If we consider one word/context pair (w, c_{pos}) with its k noise words c_{neg_i}
  - loss function L to be minimized (hence the -)
- We minimize this loss function using stochastic gradient descent (SGD)
  - take the derivative of Eq. 6.34 with respect to the different embeddings

Learning skip-gram embeddings III

- It turns out the derivatives are the following (we leave proof as exercise)

$$\partial LCE/\partial c_{pos} = [\sigma(c_{pos} \cdot w) - 1]w \tag{7}$$

$$\partial LCE/\partial c_{neg} = [\sigma(c_{neg} \cdot w)]w \tag{8}$$

$$\partial LCE/\partial w = [\sigma(c_{pos} \cdot w) - 1]c_{pos} + \sum [\sigma(c_{neg_i} \cdot w)]c_{neg_i} \tag{9}$$

- The update equations going from time step t to $t+1$ in SGD
- two embeddings for each word i: the target embedding w_i and the context
  - It's common to just add them together
  - Alternatively we can throw away the C matrix
- the context window size L affects the performance of skip-gram embeddings
  - experiments often tune the parameter L on a devset

Other kinds of static embeddings I

- fasttext (Bojanowski+ 2017)
  - problems with word2vec
    - deal with unknown words—words that appear in a test corpus
    - A related problem is word sparsity, such as in languages with rich morphology: many forms for each noun and verb
  - subword models
    - representing each word as itself plus a bag of constituent n-grams, with special boundary symbols $<$ and $>$ added to each word
    - embedding is learned for each constituent n-gram
    - the word *where* is represented by the sum of all of the embeddings of its constituent n-grams
  - A fasttext open-source library, including pretrained embeddings for 157 languages, is available at https://fasttext.cc

Other kinds of static embeddings II

- GloVe (Pennington+ 2014), short for Global Vectors
  - capturing global corpus statistics
  - based on ratios of probabilities from the word-word cooccurrence matrix,
    - combining the intuitions of count-based models like PPMI
      while also capturing the linear structures used by methods like word2vec
- elegant mathematical relationship with sparse embeddings like PPMI
  - word2vec can be seen as implicitly optimizing a shifted version of a PPMI mx (Levy and Goldberg, 2014c)

Áttekintés

Visualizing embeddings

- list the most similar words to w
    - sorting the vectors by their cosine to w
- hierarchical clustering algorithm
- project the 100 dimensions of a word down into 2 dimensions
    - projection method eg t-SNE (van der Maaten and Hinton, 2008)

Áttekintés

Different types of similarity or association

- size of the context window used to collect counts
  - relevant to both sparse tf-idf vectors and dense word2vec vectors
  - generally between 1 and 10 words on each side of the target word
  - The choice depends on the goals of the representation
    - Shorter context windows tend to lead to more syntactic representations * the most similar words to a target word w tend to be semantically similar words with the same parts of speech
    - long context windows, the highest cos: topically related but not similar
  - eg Levy and Goldberg (2014a) using skip-gram with a window of
    - ± 2, the nearest neighbors of *Hogwarts* (from the Harry Potter series) were names of other fictional schools: *Sunnydale* or *Evernight*
    - ± 5, the most similar words to *Hogwarts*: *Dumbledore*, *Malfoy*, & *half-blood*
- two kinds of similarity or association (Schütze and Pedersen, 1993)
  - first-order co-occurrence (sometimes called syntagmatic association) if they are typically nearby each other
    - Thus *wrote* is a first-order associate of *book* or *poem*
  - second-order co-occurrence (sometimes called paradigmatic association) if they have similar neighbors

Analogy/Relational Similarity I

- embeddings can capture relational meanings
- the parallelogram model
- In an important early vector space model of cognition,
- Rumelhart and Abrahamson (1973) proposed it
- for solving simple analogy problems of the form
  a is to b as a* is to what?
- *apple:tree::grape:*?, ie, *apple* is to *tree* as, and must fill in *vine*
- the vector from the word apple to the word tree ($= \mathit{tree} - \mathit{apple}$) is added to the vector for *grape*
  - the nearest word to that point is returned
- early sparse vector models of meaning could solve such analogy problems (Turney and Littman, 2005)

Analogy/Relational Similarity II

- more modern attention because of its success with word2vec or GloVe vectors (Mikolov, Yih, and Zweig, 2013; Levy and Goldberg, 2014; Pennington, Socher, and Manning, 2014)
  eg $king - man + woman$ is a vector close to *queen*
  $Paris - France + Italy$ results in a vector that is close to *Rome*
- ie The embedding model extracts representations of relations like MALE-FEMALE, or CAPITAL-CITY-OF, or even COMPARATIVE/SUPERLATIVE
- For a $a : b :: a* : b*$ problem
  - given vectors a, b, and a* and must find b*
  - $\widehat{b*} = \arg\min distance(x, b - a + a*)$
    with some distance function

Analogy: caveats

- the closest value returned by the parallelogram algorithm in word2vec or GloVe embedding spaces is usually not in fact b* but one of the 3 input words or their morphological variants (ie, *cherry:red :: potato:*x returns *potato* or *potatoes* instead of *brown*),
- these must be explicitly excluded
- embedding spaces perform well if the task involves frequent words, small distances, and certain relations (like relating countries with their capitals or verbs/nouns with their inflected forms)
- doesn't work as well for other relations (Linzen (2016) and Gladkova and Drozd (2016), Schluter 2018, Ethayarajh (2019))
- Peterson+ (2020): the parallelogram method is in general too simple to model the human cognitive process of forming such analogies

Embeddings and Historical Semantics

- for studying how meaning changes over time, by computing multiple embedding spaces, each from texts written in a particular time period
- eg Fig. 6.17 shows a visualization of changes in meaning in English words over the last two centuries
    - from historical corpora like Google n-grams (Lin+ 2012) and the Corpus of Historical American English (Davies, 2012)
- A t-SNE visualization of the semantic change of 3 words in English
    - The modern sense of each word, and the grey context words, are computed from most recent (modern) time-point embedding space
    - Earlier points are computed from earlier historical embedding spaces
    - gay, broadcast, aweful

Áttekintés

Bias and embeddings

- embeddings can roughly model relational similarity
- gender stereotypes. Bolukbasi+ (2016)
    - $-man + computer\ programmer + woman = homemaker$
    - *father* is to *doctor* as *mother* is to *nurse*
    - allocational harm (Crawford (2017) and Blodgett+ (2020))
        - a system allocates resources (jobs or credit) unfairly to different groups
        - For example algorithms that use embeddings for hiring potential programmers or doctors might thus incorrectly downweight documents with women's names
- embeddings don't just reflect the statistics of their input, but also amplify bias
    - gendered terms become more gendered in embedding space than they were in the input text statistics (Zhao+ 2017, Ethayarajh+ 2019b, Jia+ 2020),
    - biases are more exaggerated than in actual labor employment statistics (Garg+ 2018)

Bias: Embeddings also encode implicit associations I

- a property of human reasoning
- The Implicit Association Test (Greenwald+ 1998)
    - measures people's associations between concepts (like *flowers* or *insects*) and attributes (like *pleasantness* and *unpleasantness*) by measuring differences in the latency with which they label words in the various categories
- push a green button for *flowers* (daisy, iris, lilac) and *pleasant words* (love, laughter, pleasure) and a
  red button for *insects* (flea, spider, mosquito) and *unpleasant words* (abuse, hatred, ugly) they are
  faster than in an incongruous condition where they push a red button for *flowers* and *unpleasant words* and a green button for *insects* and *pleasant words*

Bias: Embeddings also encode implicit associations II

- $\rightarrow$ people in the United States associate
  - African-American names with unpleasant words (more than European-American)
  - male names more with mathematics and female names with the arts
  - old people's names with unpleasant words (Greenwald+ 1998, Nosek+ 2002a, Nosek+ 2002b)
- Caliskan+ (2017) replicated all these findings of implicit associations using GloVe vectors and cosine similarity instead of human latencies
  - eg African-American names like *Leroy* and *Shaniqua* had a higher GloVe cosine with unpleasant words while European-American names (*Brad*, *Greg*, *Courtney*) had a higher cosine with pleasant words
  - representational harm (Crawford 2017, Blodgett+ 2020)
  - system demeaning [megaláz?] or even ignoring some social groups

Bias: remove these kinds of biases

- transformation of the embedding space that removes gender stereotypes but preserves definitional gender (Bolukbasi+ 2016, Zhao+ 2017)
- other training procedure (Zhao+ 2018)
- may reduce bias, they do not eliminate it (Gonen and Goldberg, 2019)

Bias: Historical embeddings

to measure biases in the past

- Garg+ (2018) occupations and ethnicities or genders
  - eg women's names versus men's to occupations like *librarian* or *carpenter* across the 20th century
  - the cosines correlate with the empirical historical percentages of women or ethnic groups in those occupations
- also replicated old surveys of ethnic stereotypes
  - the tendency of experimental participants in 1933 to associate adjs like *industrious* or *superstitious* [babonás] with, e.g., Chinese ethnicity,
  - correlates with the cosine between Chinese last names and those adjectives using embeddings trained on 1930s text
- document historical gender biases
  - eg embeddings for adjectives related to competence (*smart*, *wise*, *thoughtful*, *resourceful*) had a higher cosine with male than female words, and showed that this bias has been slowly decreasing since 1960

Áttekintés

Evaluating vector models I

- The most important evaluation metric: extrinsic evaluation
  - whether using vectors improves performance over some other model
- useful to have intrinsic evaluations
  - most common metric is to test their performance on similarity
    - the correlation between an algorithm's word similarity scores and word similarity ratings assigned by humans
  - WordSim-353 (Finkelstein et al., 2002) is a commonly used set of ratings from 0 to 10 for 353 noun pairs; for example *(plane, car)*
  - SimLex-999 (Hill, Reichart, and Korhonen, 2015) is a more difficult dataset that
    - similarity (cup, mug) rather than relatedness (cup, coffee)
    - both concrete and abstract adjective, noun and verb pairs

Evaluating vector models II

- TOEFL dataset is a set of 80 questions, each consisting of a target word with 4 additional word choices
    - choose which is the correct synonym, as in the example:
    - eg *levied* is closest in meaning to: *imposed, believed, requested, correlated* (Landauer and Dumais, 1997)
- All of these datasets present words without context

- intrinsic similarity tasks that include context
    - The Stanford Contextual Word Similarity (SCWS) dataset
        - (Huang et al., 2012)
        - human judgments on 2,003 pairs of words in their sentential context
    - Word-in-Context (WiC) dataset
        - (Pilehvar and Camacho-Collados, 2019)
        - target words in two sentential contexts that are either in the same or different senses
    - The semantic textual similarity task
        - (Agirre et al., 2012; Agirre et al., 2015)
        - evaluates sentence-level similarity algorithms
        - a set of pairs of sentences, each pair with human-labeled similarity score

Evaluating vector models III

- analogy task
  - A number of sets of tuples have been created for this task
  - (Mikolov+ 2013a, Mikolov+ 2013c, Gladkova and Drozd (2016)),
    - morphology (city:cities::child:children),
    - lexicographic relations (leg:table::spout::teapot) and
    - encyclopedia relations (Beijing:China::Dublin:Ireland)
  - some drawing from the SemEval-2012 Task 2 dataset of 79 different relations (Jurgens+ 2012)
- variability: All embedding algorithms suffer from it
  - randomness in the initialization and the random negative sampling,
  - individual documents in a collection may strongly impact the result (Tian+ 2016, Hellrich and Hahn 2016, Antoniak and Mimno 2018)
  - best practice to train multiple embeddings with bootstrap sampling over documents and average the results (Antoniak and Mimno, 2018)

Áttekintés

Idea from research in the 1950s in 3 fields I

- ling: meaning is related to the distribution of words in context
  - widespread in linguistic theory of the 1950s, among
  - distributionalists like Zellig Harris, Martin Joos, and J. R. Firth, and
  - semioticians like Thomas Sebeok
  - As Joos (1950) put it, the linguist's "meaning" of a morpheme. . . is by definition the set of conditional probabilities of its occurrence in contx
- psychology: the meaning of a word might be modeled as a point in a multidimensional semantic space
  - from psychologists like Osgood
  - people assigning values along scales like happy/sad or hard/soft
  - Osgood+ (1957) proposed that
  - meaning of a word modeled as a point in a multidim Euclidea space
  - similarity of meaning between two words modeled as the distance

Idea from research in the 1950s in 3 fields II

- comp sci: mechanical indexing
  - in the 1950s and early 1960s
  - now known as information retrieval
  - the vector space model for info retrieval
    - (Salton 1971, Sparck Jones 1986)
  - new ways to define the meaning of words in terms of vectors
    - (Switzer 1965)
  - refined methods for word similarity based on measures of statistical association between words like mutual information (Giuliano, 1965) and idf (Sparck Jones, 1972)
  - the meaning of documents could be represented in the same spaces as words

Wittgenstein

philosophical underpinning of the distributional way of thinking

- late writings of the philosopher Wittgenstein
- W was skeptical of the possibility of building a completely formal theory of meaning definitions for each word
- “the meaning of a word is its use in the language” (Wittgenstein, 1953, PI 43)
- instead of using some logical language to define each word, or drawing on denotations or truth values
- we should define a word by how it is used by people in speaking and understanding day to day
- prefiguring the movement toward embodied and experiential ling/NLP model (Glenberg & Robertson 2000, Lake & Murphy 2021, Bisk+ 2020, Bender & Koller 2020)

Defining words by a vector of discrete features

- More distantly related
- roots at least as far back as Descartes and Leibniz (Wierzbicka 1992, Wierzbicka 1996)
- Hjelmslev, (1943/1969)
- early models of generative grammar (Katz and Fodor, 1963)
- representing meaning with semantic features, symbols that represent some sort of primitive meaning
- eg hen, rooster, or chick

  hen +female, +chicken, +adult
  rooster -female, +chicken, +adult
  chick +chicken, -adult

- some attempt to show that certain dimensions of embedding models contribute some specific compositional aspect of meaning like these early semantic features

Latent Semantic Analysis

- Dense vectors to model word meaning
  - and the term embedding
- latent semantic indexing (LSI) model (Deerwester+ 1988)
- recast as LSA (latent semantic analysis, Deerwester+ 1990)
- In LSA singular value decomposition—SVD— is applied to a term-document mx
  - weighted by log frequency and normalized by entropy
  - the first 300 dimensions are used as the LSA embedding
  - Singular Value Decomposition (SVD) is a method for finding the most important dimensions of a data set, those dimensions along which the data varies the most

LSA applied

- as a cognitive model (Landauer and Dumais, 1997)
- spell checking (Jones and Martin, 1997)
- language modeling (Bellegarda 1997, Coccaro and Jurafsky 1998, Bellegarda 2000)
- morphology induction (Schone & Jurafsky 2000, Schone & Jurafsky 2001b)
- multiword expressions (MWEs, Schone and Jurafsky, 2001a)
- essay grading (Rehder+ 1998)
- Related models were simultaneously developed and applied to word sense disambiguation by Schütze (1992)
- in a probabilistic classifier, in the logistic regression document router of Schütze+ (1995)

Alternatives to LSA

- SVD on the term-term matrix
  - as a model of meaning for NLP was proposed soon after LSA by Schütze (1992)
  - (97-dimensional) embeddings produced by SVD to the task of word sense disambiguation, analyzed the resulting semantic space, and also suggested possible techniques like dropping high-order dimensions
    - See Schütze (1997)
- alternative matrix models followed on from the early SVD work
  - Probabilistic Latent Semantic Indexing (PLSI, Hofmann, 1999),
  - Latent Dirichlet Allocation (LDA, Blei+ 2003), and
  - Non-negative Matrix Factorization (NMF, Lee and Seung, 1999)
- first use of the word "embedding" in Landauer+ (1997)
  - in a variant of its mathematical meaning
    - as a mapping from one space or mathematical structure to another
  - described the mapping from the space of sparse count vectors to the latent
  - metonymically shifted to mean the resulting dense vector in the latent space

Neural Language Models I

- develop embeddings as part of the task of word prediction
  - Bengio+ (2003, 2006)
- embeddings could be used to represent word meanings for many tasks
  - Collobert & Weston (2007), Collobert & Weston (2008), & Collobert+ (2011)
- Turian+ (2010) compared the value of different kinds of embeddings for different NLP tasks
- Mikolov+ (2011): recurrent neural nets could be used as LMs
- simplifying the hidden layer of these neural net language models to create the skip-gram (and also CBOW) algorithms was proposed by Mikolov+ (2013a)
- negative sampling training algorithm (Mikolov+ 2013b)
- surveys of static embeddings and their parameterizations
  - (Bullinaria and Levy 2007, Bullinaria and Levy 2012, Lapesa and Evert 2014, Kiela and Clark 2014, Levy+ 2015)

Bibliographical notes continued

- for a deeper understanding of the role of vectors in information retrieval,
  - See Manning+ (2008)
  - how to compare queries with documents, more details on tf-idf, and issues of scaling to very large datasets
- Kim (2019): a clear and comprehensive tutorial on word2vec
- Cruse (2004): introductory linguistic text on lexical semantics

Bibliography I

Agirre, Eneko et al. (2012). “SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity”. In: *First Joint Conference on Lexical and Computational Semantics (*SEM)*. Montréal, Canada: Association for Computational Linguistics, pp. 385–393 (cit. on p. 78).

Agirre, Eneko et al. (2015). “SemEval-2015 Task 2: Semantic Textual Similarity, English, Spanish and Pilot on Interpretability”. In: *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Denver, CO, U.S.A. (cit. on p. 78).

Bengio, Yoshua, Aaron Courville, and Pascal Vincent (2013). “Representation Learning: A Review and New Perspectives”. In: *IEEE Trans. PAMI* 35.8, pp. 1798–1828. URL: http://arxiv.org/abs/1206.5538 (cit. on pp. 4, 5).

Bengio, Yoshua et al. (2003). “A Neural Probabilistic Language Model”. In: *Journal of Machine Learning Research* 3, pp. 1137–1155. URL: http://www.jmlr.org/papers/v3/bengio03a.html (cit. on p. 54).

Bojanowski, Piotr, Armand Joulin, and Tomas Mikolov (2016). “Alternative Structures for Character-level RNNs”. In: *International Conference on Learning Representations, Workshop track (ICLR 2016)*. arXiv: 1511.06303 [cs.LG].

Bibliography II

Church, Kenneth W. and Patrick Hanks (1990). “Word association norms, mutual information, and lexicography”. In: *Computational Linguistics* 16.1, pp. 22–29 (cit. on p. 45).

Collobert, R. et al. (2011). “Natural Language Processing (Almost) from Scratch”. In: *Journal of Machine Learning Research (JMLR)* (cit. on p. 54).

Devlin, Jacob et al. (Oct. 11, 2018). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. Version 1. In: *arXiv preprint arXiv:1810.04805*. arXiv: 1810.04805v1 [cs.CL]. URL: http://arxiv.org/abs/1810.04805v1.

Ethayarajh, Kawin (2019). “How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2”. In: *EMNLP* (cit. on p. 68).

Fano, Robert (1961). *Transmission of Information: A Statistical Theory of Communications*. MIT Press (cit. on p. 45).

Finkelstein, Lev et al. (2002). “Placing Search in Context: The Concept Revisited”. In: *ACM Transactions on Information Systems* 20(1), pp. 116–131 (cit. on p. 77).

Firth, John R. (1957). “A synopsis of linguistic theory”. In: *Studies in linguistic analysis*. Blackwell, pp. 1–32 (cit. on pp. 2, 3, 25).

Bibliography III

Gladkova, Anna and Aleksandr Drozd (2016). “Intrinsic Evaluations of Word Embeddings: What Can We Do Better?” In: *Proc. RepEval (this volume)*. Ed. by Omer Levy. ACL (cit. on pp. 68, 79).

Halácsy, Péter et al. (2004). “Creating open language resources for Hungarian”. In: *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*. ELRA, pp. 203–210.

Harris, Zellig S. (1954). “Distributional structure”. In: *Word* 10.23, pp. 146–162 (cit. on pp. 2, 3, 25).

Hashimoto, Kazuma et al. (2017). “A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks”. In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Hill, Felix, Roi Reichart, and Anna Korhonen (2015). “Simlex-999: Evaluating semantic models with (genuine) similarity estimation”. In: *Computational Linguistics* 41.4, pp. 665–695 (cit. on pp. 12, 77).

Hinton, G. et al. (2012). “Deep neural networks for acoustic modeling in speech recognition”. In: *IEEE Signal Processing Magazine* 29, pp. 82–97 (cit. on p. 53).

Bibliography IV

Howard, Jeremy and Sebastian Ruder (2018). “Universal Language Model Fine-tuning for Text Classification”. In: *ACL*.

Huang, Eric et al. (2012). “Improving Word Representations via Global Context and Multiple Word Prototypes”. In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*. Jeju Island, Korea: Association for Computational Linguistics, pp. 873–882 (cit. on p. 78).

Joos, Martin (1950). “Description of language design”. In: *Journal of the Acoustical Society of America* 22, pp. 701–708 (cit. on pp. 2, 3, 25).

Krizhevsky, A. and G. Sutskever I.and Hinton (2012). “ImageNet classification with deep convolutional neural networks”. In: *NIPS’2012* (cit. on p. 53).

Landauer, Thomas K and Susan T Dumais (1997). “A solution to Plato’s problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge.”. In: *Psychological review* 104.2, p. 211 (cit. on pp. 78, 86).

Lazaridou, Angeliki et al. (2013). “Compositionally Derived Representations of Morphologically Complex Words in Distributional Semantics”. In: *ACL (1)*, pp. 1517–1526. URL: http://aclweb.org/anthology/P/P13/P13-1149.pdf.

Bibliography V

Levy, Omer and Yoav Goldberg (2014). “Linguistic Regularities in Sparse and Explicit Word Representations”. In: *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*. Ann Arbor, Michigan: Association for Computational Linguistics, pp. 171–180. URL: http://aclweb.org/anthology/W14-1618 (cit. on p. 67).

Levy, Omer et al. (2015). “Do Supervised Distributional Methods Really Learn Lexical Inference Relations?” In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, pp. 970–976. DOI: 10.3115/v1/N15-1098. URL: https://www.aclweb.org/anthology/N15-1098 (cit. on p. 47).

Linzen, Tal (2016). “Issues in evaluating semantic spaces using word analogies”. In: *RepEval* (cit. on p. 68).

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig (2013). “Linguistic Regularities in Continuous Space Word Representations”. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013)*. Atlanta, Georgia: Association for Computational Linguistics, pp. 746–751 (cit. on p. 67).

Bibliography VI

Mikolov, Tomas et al. (2013). “Distributed Representations of Words and Phrases and their Compositionality”. In: *Advances in Neural Information Processing Systems 26*. Ed. by C.J.C. Burges et al. Curran Associates, Inc., pp. 3111–3119. URL: https://bit.ly/39HikH8.

Niwa, Yoshiki and Yoshihiko Nitta (1994). “Cooccurrence vectors from corpora vs. distance vectors from dictionaries”. In: *Proceedings of the 15th conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, pp. 304–309 (cit. on p. 46).

Oravecz, Csaba, Tamás Váradi, and Bálint Sass (2014). “The Hungarian Gigaword Corpus”. In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*. Reykjavik, Iceland: European Language Resources Association (ELRA). URL: http://www.aclweb.org/anthology/L14-1536.

Osgood, Charles E., George Suci, and Percy Tannenbaum (1957). *The measurement of meaning*. University of Illinois Press (cit. on pp. 22, 23, 25).

Bibliography VII

Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014). “Glove: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: http://www.aclweb.org/anthology/D14-1162 (cit. on p. 67).

Peters, Matthew et al. (2018). “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: 10.18653/v1/N18-1202. URL: http://aclweb.org/anthology/N18-1202.

Vaswani, Ashish et al. (2017). “Attention is All you Need”. In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., pp. 5998–6008. arXiv: 1706.03762 [cs.CL]. URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.