GloVe: Global Vectors for Word Representation

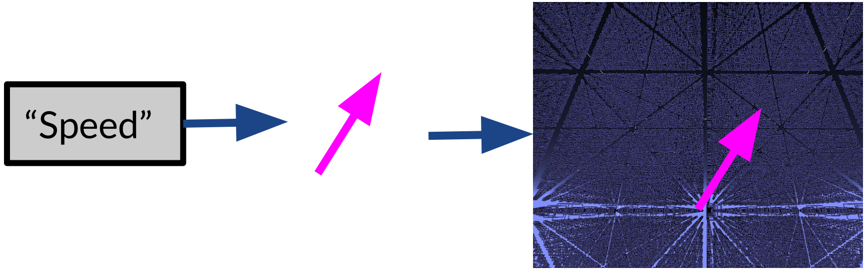
Presented at the
2014 Conference on Empirical Methods in Natural Language Processing.
Authored by
Jeffrey Pennington, Richard Socher, and Christopher D. Manning

Presented by Chang Ju Kim, Tommy Sanford, Ben Mo, Erik Svetlichny, Tim Budding

Getting Up to Speed:

To be able to define words in vector space so that computers can perform complex language tasks using a mathematical framework e.g. solving analogies.

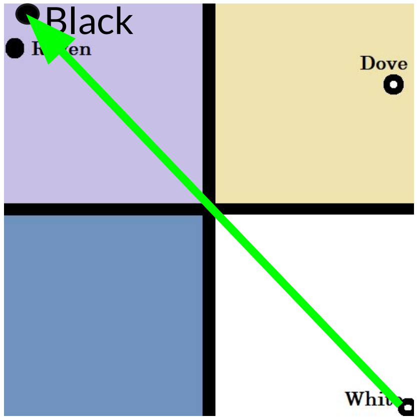
Jargonspeak: want a word vector space with a meaningful substructure.

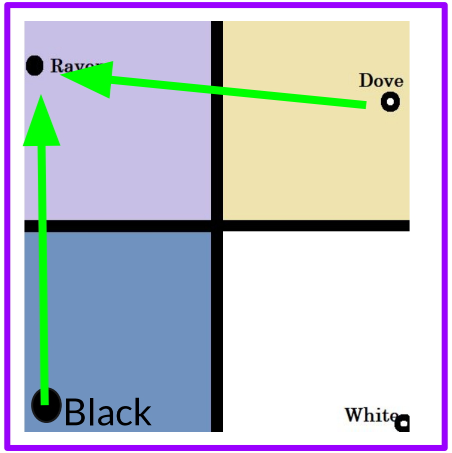


Word → Vector → Vector in Semantic Vector Space

Motivating Question

How can we construct a meaningful vector space using a collection of text documents?





How to evaluate Word Embeddings

Benchmarks tested in the paper

- Word analogy
  - “A is to B as C is to __”
- Word similarity
  - Find how similar two different words are
  - Distance between word vectors should correlate to how similar in meaning the words are
  - Commonly uses cosine similarity
- Named entity recognition
  - Tag entities in text with its type
  - e.g. ConLL-03 - 4 Types: person, location, organization, miscellaneous

How to evaluate Word Embeddings

Other evaluators

- Outlier Detection
  - Find words that do not belong
- Concept Categorization
  - Split a group of words into categories
- Part of Speech Tagging
  - Assign part of speech to different words
- Chunking
  - Group words from a sentence into phrases
- Sentiment Analysis
  - Classify text between positive or negative

Related Work

Two main model categories:

1. Global matrix factorization methods (LSA)
    a. (+) Uses matrices (so statistical information is easily used)
    b. (-) Bad at word analogies (what words occur frequent with others)
2. Local shallow window-based methods (skip-gram)
    a. (+) Great at word analogies (what words occurs frequently with others)
    b. (-) Needs to scan through the documents (doesn't directly use statistical data)

Matrix Factorization: Generating the Matrix

Main objective: decompose large matrices containing statistics about text.

Two methods to get the large matrix:

1. Latent Semantic Analysis (LSA): n x d matrix
   - Rows: words
   - Columns: # of documents in the corpus
   - Entries: # of times word appears in each document
2. Hyperspace Analogue to Language (HAL): n x n matrix
   - Rows: words
   - Columns: words
   - Entries: # of times and how close $word_1$ appears near $word_2$

Getting the matrix allows you to perform computations on it

Problem with the Matrix

Frequent words are disproportionately large in the matrix.

- E.g. *the* and *and* appear frequently with other words.

Thus, to an article about bananas, it will seem like *the* and *and* are closely associated with bananas despite having little to do with their meaning.

Main workarounds include COALS, PPMI, and HPCA

Article 3 of the US constitution. Has only 378 words, but 27 thes and 17 ands

Article III

Section 1

The judicial Power of the United States, shall be vested in one Supreme Court, and in such inferior Courts as the Congress may from time to time ordain and establish. The Judges, both of the supreme and inferior Courts, shall hold their Offices during good Behavior, and shall, at stated Times, receive for their Services, a Compensation, which shall not be diminished during their Continuance in Office.

Section 2

The Judicial Power shall extend to all Cases, in Law and Equity, arising under this Constitution, the Laws of the United States, and Treaties made, or which shall be made, under their Authority;---to all Cases affecting Ambassadors, other public Ministers and Consuls;---to all Cases of admiralty and maritime Jurisdiction;---to Controversies to which the United States shall be a Party;---to Controversies between two or more States;---between a State and Citizens of another State;---between Citizens of different States,---between Citizens of the same State claiming Lands under Grants of different States, and between a State, or the Citizens thereof, and foreign States, Citizens or Subjects.

In all Cases affecting Ambassadors, other public Ministers and Consuls, and those in which a State shall be a Party, the Supreme Court shall have original Jurisdiction. In all the other Cases before mentioned, the Supreme Court shall have appellate Jurisdiction, both as to Law and Fact, with such Exceptions, and under such Regulations as the Congress shall make.

The Trial of all Crimes, except in Cases of Impeachment, shall be by Jury; and such Trial shall be held in the State where the said Crimes shall have been committed; but when not committed within any State, the Trial shall be at such Place or Places as the Congress may by Law have directed.

Section 3

Treason against the United States, shall consist only in levying War against them, or in adhering to their Enemies, giving them Aid and Comfort. No Person shall be convicted of Treason unless on the testimony of two Witnesses to the same overt Act, or on Confession in open court.

The Congress shall have Power to declare the Punishment of Treason, but no Attainder of Treason shall work Corruption of Blood, or Forfeiture except during the Life of the Person attainted.

Window-Based Methods

A different approach that focuses on local context windows.

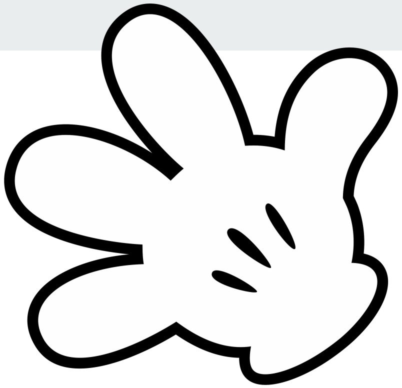
Current approaches either predict a word's context given the word, or predict a word given the context.

- Skip-gram, Continuous bag-of-words (CBOW) (Aka word2vec)
  - Trains a neural network using local windows
  - Uses the probability vector from that neural network
- vLBL, ivLBL (log bilinear models)
  - Predicts next word by linearly combining the representations of previous/context words
- These schemes end up being fairly good at the anologies task

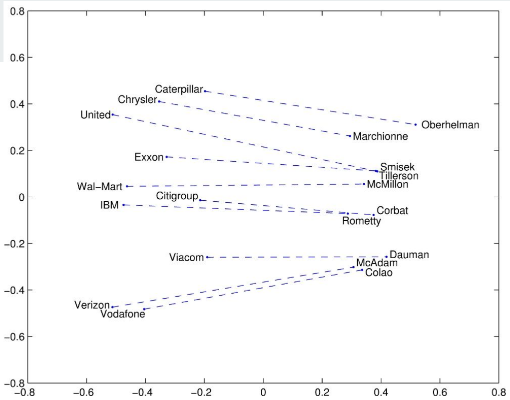
"I [do not like green eggs and ham.] I do not like them, Sam-I-Am."

A window size of 3 centered around 'green'

Introduction to GloVe

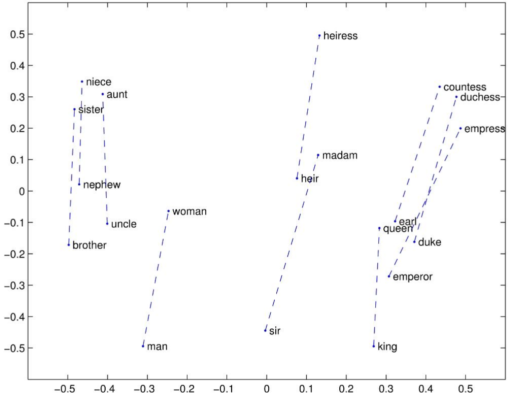


- Authors - Jeffrey Pennington, Richard Socher, Christopher D. Manning. 2014
- Developed by the Computer Science Department, Stanford University
- Problem: current models are great at capturing “fine grained semantic and syntactic regularities” but the causes are “opaque.”



GloVe Model Explained

- Unsupervised ML algorithm for obtaining vector representations for words
- Algorithm incorporates contex - other words in close proximity
- Builds a term-term matrix
    - Ex. X(i, j) gets larger and larger the more the word i appears in the context of word j



Correspondence Matrix

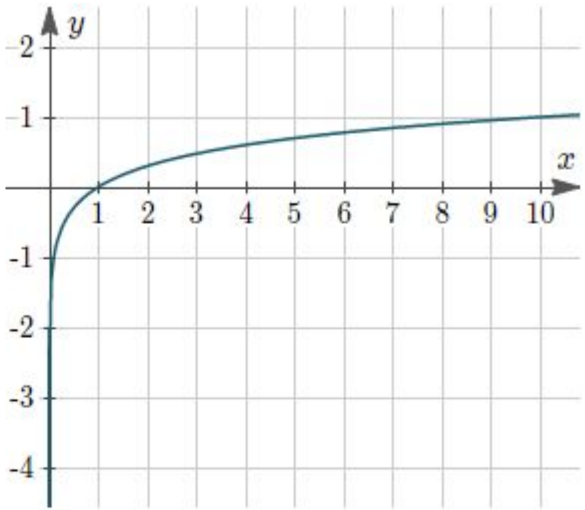
- s = “machine learning is fun”
- X(machine, learning) += 1
- X(machine, is)+= ½
- X(machine, fun) += ⅓
- X(learning, fun) += ½
- X is usually a sparse matrix
- This pattern continues up to a tolerance levels

X(i, j) += 1/(dist(i,j)) where dist(i,j) is the number of positions away j is from i.

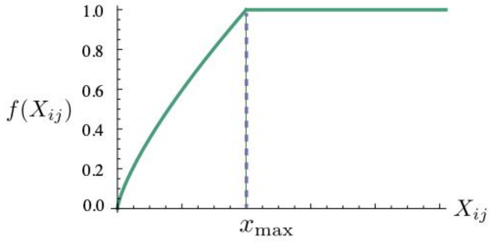
s	s[0]	s[1]	s[2]	s[3]
s[0]	0	1	½	⅓
s[1]	1	0	1	½
s[2]	½	1	0	1
s[3]	⅓	½	1	0

Matrix Scaling

- Values that are non-zero tend to be very large
- To scale down, log(1 + X(i,j)) becomes the target.



GloVe Weighting and Equation



- Assign a weight to all X(i, j) entries

  α = 0.75, Xmax = 100

$$f(x) = \begin{cases} (x/x_{\max})^{\alpha} & \text{if } x < x_{\max} \\ 1 & \text{otherwise} \end{cases}$$

$$w_i^T \tilde{w}_k + b_i + \tilde{b}_k = \log(X_{ik})$$

- Final loss function:

$$\hat{J} = \sum_{i,j} f(X_{ij})(w_i^T \tilde{w}_j - \log X_{ij})^2$$

Model Complexity

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2 \qquad (8)$$

- Complexity of evaluating the cost function (8) scales with the number of non-zero entries in X denoted |X|.

- Worst Case: $|X| = |V|^2$
- To achieve better bounds we need to make some assumptions!

Assumptions

Assume that the co-occurrence of word i with word j can be modeled using a power-law function of the frequency rank of that word pair, i.e.

$$X_{ij} = \frac{k}{(r_{ij})^{\alpha}} \qquad (17)$$

Where r_{ij} is the frequency ranking of the word pair i and j.

The most frequently occurring pair of words has r = 1 , the second most frequent has r = 2. etc.

Implications

Under the assumption on the previous slide we have two cases for |X| summarized below.

$$|X| = \begin{cases} \mathcal{O}(|C|) & \text{if } \alpha < 1, \\ \mathcal{O}(|C|^{1/\alpha}) & \text{if } \alpha > 1. \end{cases} \qquad (22)$$

The authors say that their corpora are well modeled by $\alpha = 1.25$ in which case we have

$$|X| = \mathcal{O}(|C|^{0.8}).$$

Experiments - What Can We Test

- Analogies are a common measure of word embeddings, written as

a:b::c:d

or, equivalently, "a is to be as c is to d." Goal is to predict d given a,b, and c.

- Model prediction is given by

$$d = \underset{d \in V}{\operatorname{argmax}} \operatorname{sim}(d, b - a + c)$$

Where sim(x,y) is a function which measures similarity of vectors, e.g. cosine similarity

Experiments - Original Paper

- Reported over 70% accuracy on an analogy test set published by Mikolov et al. (2013a)
  - This dataset is heavily biased towards analogies of the form "capital is to country as capital is to country"(1) e.g. Paris is to France as Rome is to Italy"

- Our experiments are performed on the "Bigger Analogy Test Set (BATS)" published by Gladkova et al. (2016) using the pretrained GloVe vectors available at (GloVe: Global Vectors for Word Representation)

1. Gladkova et al. (2016)

Experimental Difficulties

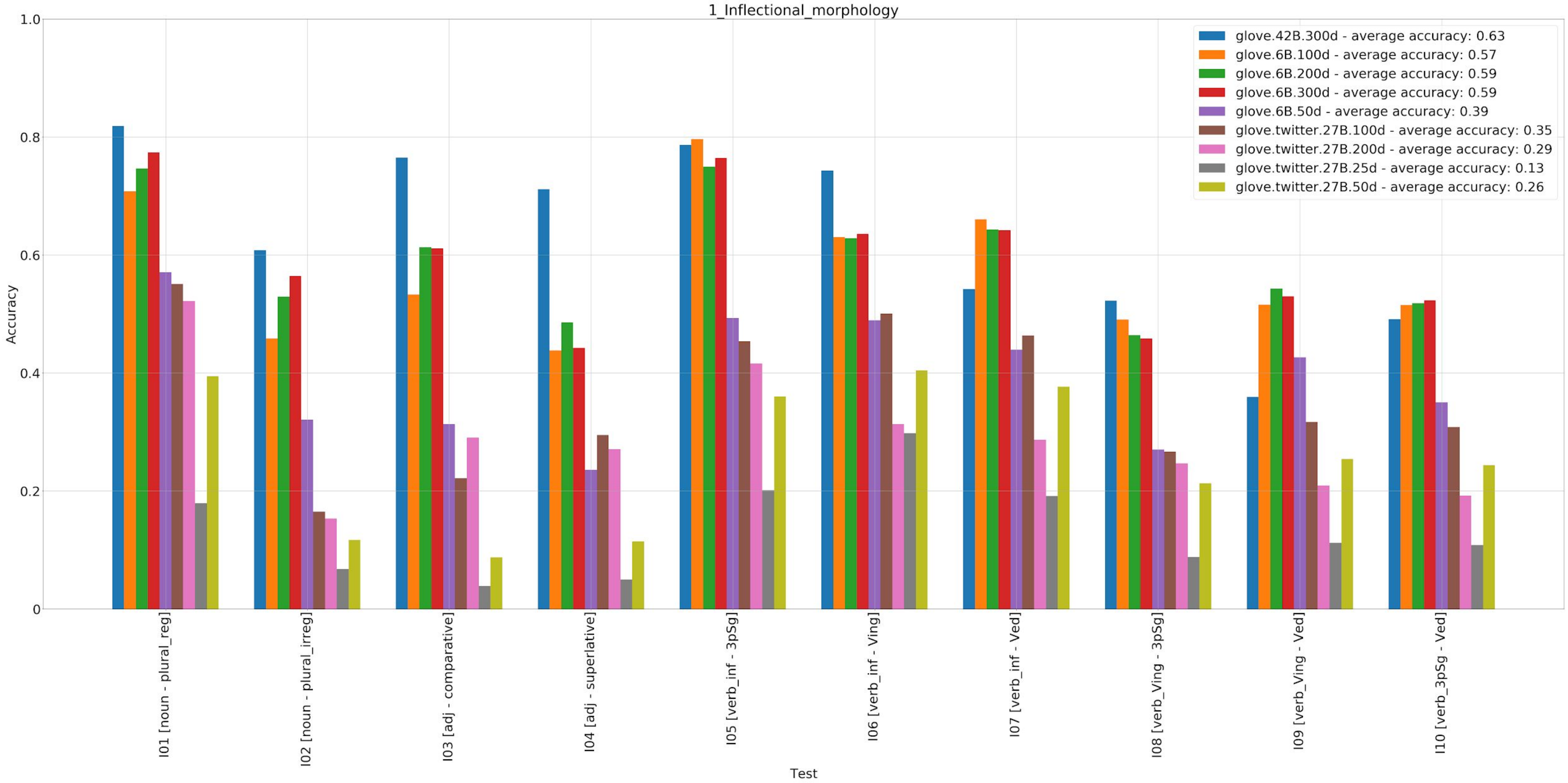
- Tried (and failed) to use open source library “vecto”
- Some of the pretrained vocabularies are LARGE i.e. 2.2. million words each stored as a 300 dimensional vector.
- Analogies can have multiple correct answers which increases difficulty of parsing input files and makes it difficult to use a fixed batch size for computation.

What We Were Able to Do

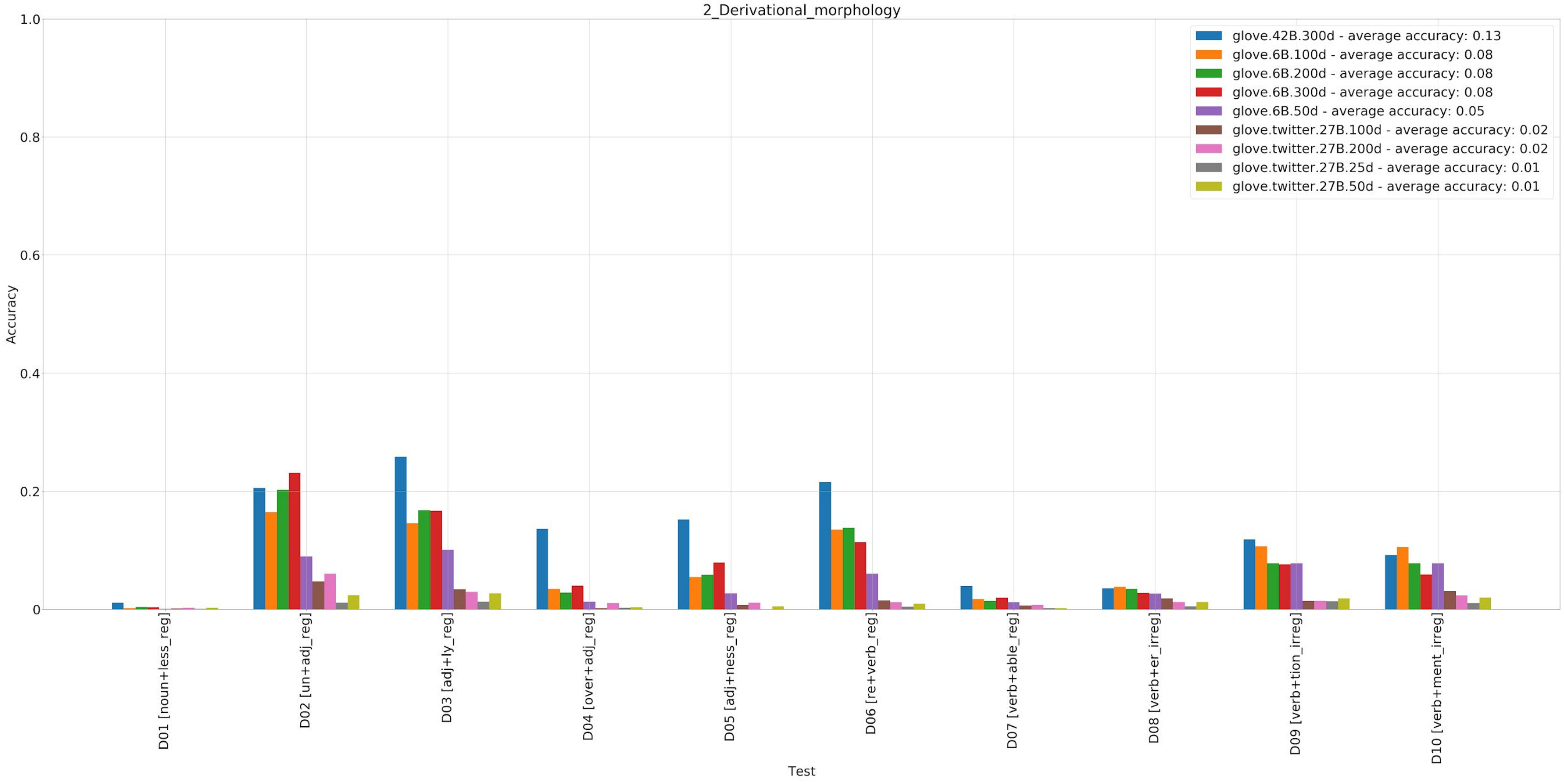
- Tested 9 out of 10 of the pretrained GloVe models from (https://nlp.stanford.edu/projects/glove/) on most of the analogy test sets in BATS.

- We were unable to test every model against every analogy test set due to limited compute power.
  - For the smaller models we were able to test against >90% of the total analogies
  - For the larger models only tested against ~50% of the total analogies.

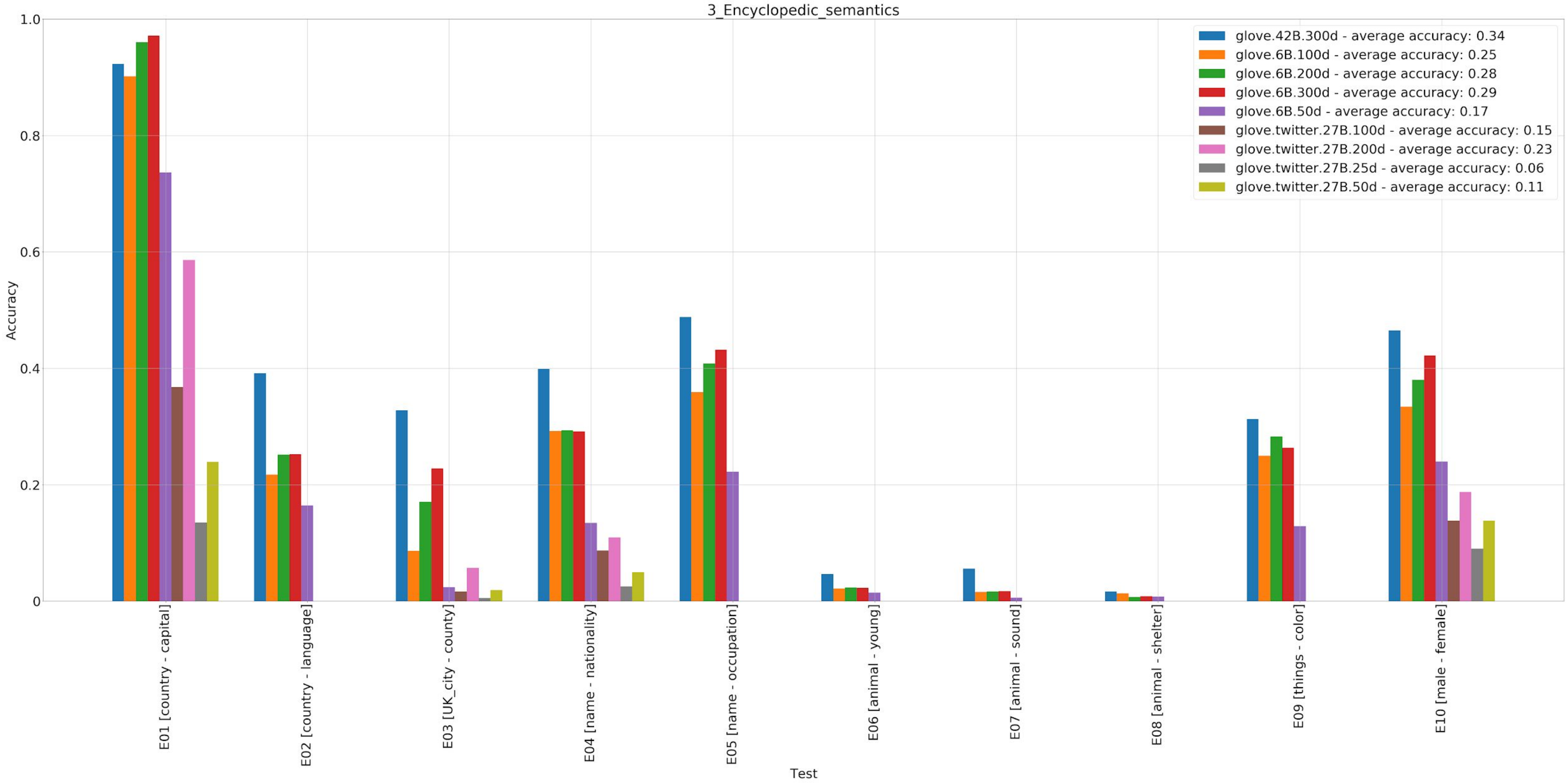
Experimental Results



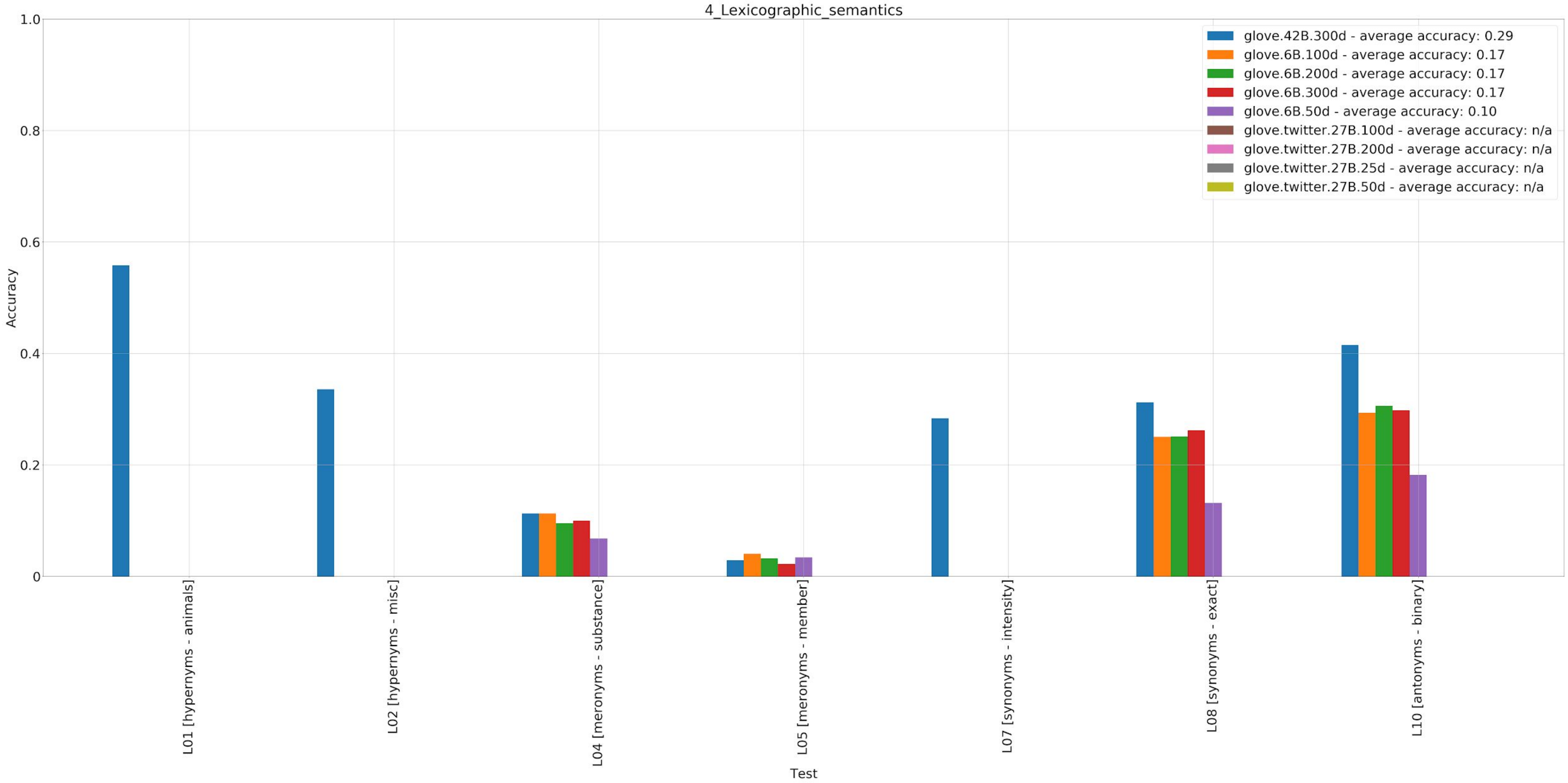
Experimental Results



Experimental Results



Experimental Results



Experiment - Conclusion

Glove works very well on:

- Modifications that don't change the meaning by a lot (e.g. book to books, angry to angrier)
- Countries to capitals

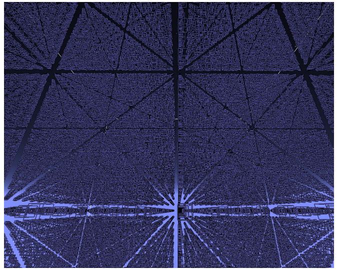
Glove performs poorly on:

- Modifications that change the meaning (e.g. bone to boneless, authorized to unauthorized)
- Words that are related but don't mean the same thing / don't have the same form (e.g. cat to kitten, flag to fabric)

Citations

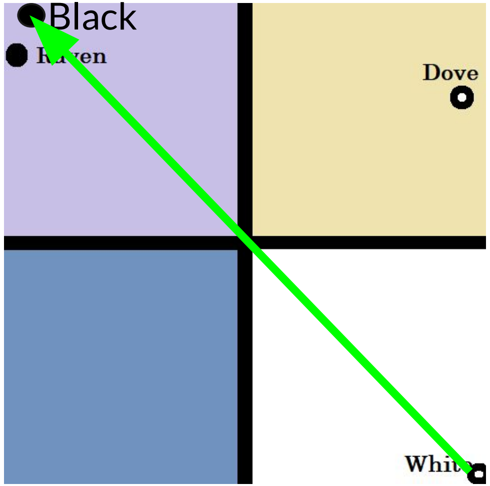
1. Pennington, Jeffrey, et al. “GloVe: Global Vectors for Word Representation.” *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543, www.aclweb.org/anthology/D14-1162.
2. Gladkova, Anna, et al. “Analogy-Based Detection of Morphological and Semantic Relations with Word Embeddings: What Works and What Doesn't.” *Proceedings of the NAACL-HLT SRW*, 2016, pp. 8–15., doi:10.18653/v1/N16-2002.

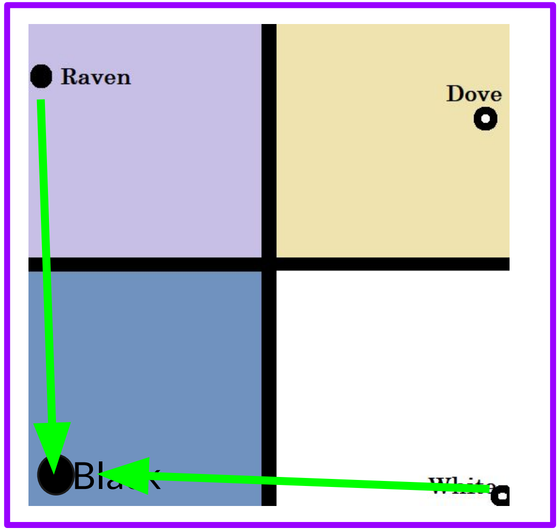
Thank You



???

How should our semantic vector space be made so it has a meaningful structure?





Workaround

The main solution is to transform the matrix.

- COALS (Correlated Occurrence Analogue to Lexical Semantics) method:
  - Basically removes all but 14,000 most frequent columns, negative values
  - Also square roots values (makes extremely small values (1e-5 to 1e-3) much closer to other values)
- PPMI: Positive Pointwise Mutual Information
- HPCA: a square root type transformation in the form of Hellinger PCA