Aligned Word Vector Spaces and Document Vectors

13111059

M.Tech. Thesis

Shashwat Chandra

Computer Science and Engineering
Indian Institute of Technology Kanpur

July 2015

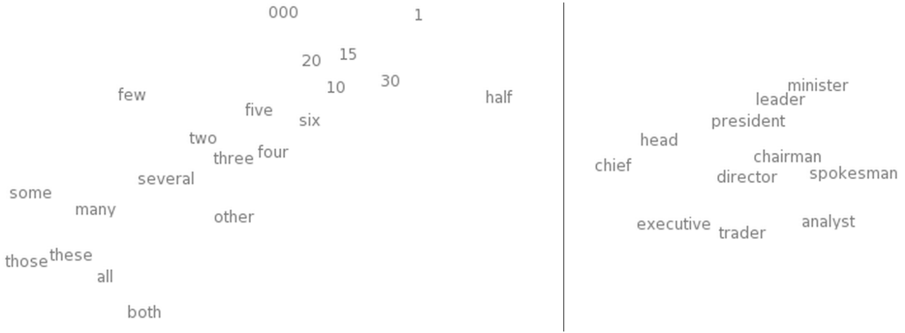
Outline

Outline

Objective

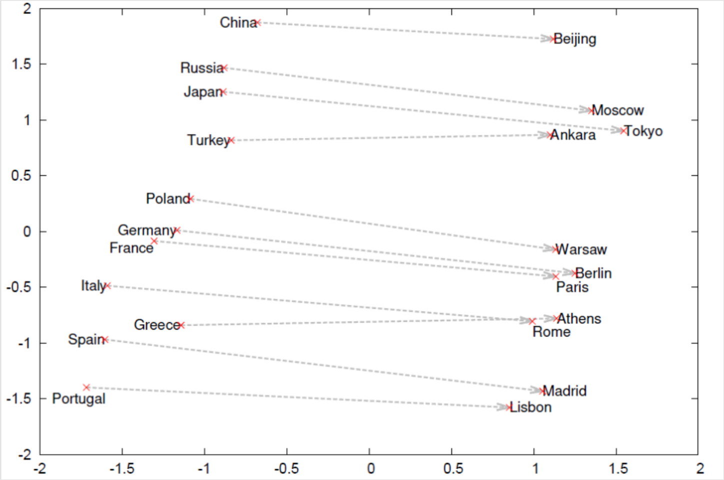
- There have been many approaches that attempt to make sense of the massive amount of unstructured text data that we have.
  - Sentiment Analysis
  - Word Sense Disambituation
  - Word Categorization
  - Discourse Comprehension
- A recent approach to perform tasks similar to these is to generate vector representations of words and documents.

Objective
contd.



t-SNE visualization of example word-vectors. (Image from Turian et. al. 2010)

Objective
contd.



Some vector differences (Image from Mikolov et. al. 2013)

Outline

Previous Work

- Language Dependent
  - Parts of Speech
  - Parse Trees
- Vector Representations of Words
  - Term Frequency
  - Deep Neural Networks (Collobert, Weston 2011)
  - Word2Vec (Mikolov et. al. 2013)
  - GloVe (Pennington et. al. 2014)
- Vector Representations of Phrases/Sentences/Documents
  - Bag of Words
  - TF-IDF
  - LSA (Landauer, Dumais 1997)
  - Paragraph Vectors (Le, Mikolov 2014)

This Work

In this work,

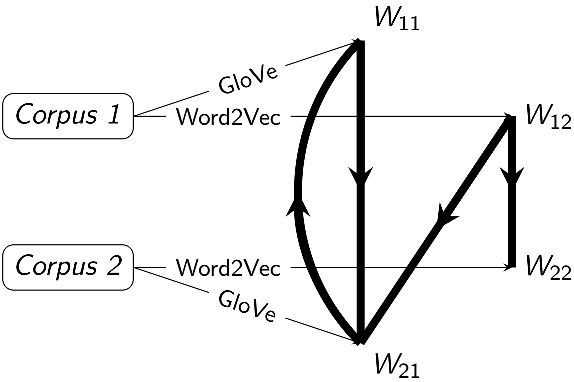
- Attempt a novel approach to merge fixed-length word vectors
  - Smaller corpus sizes/training time
  - Benefits of multiple techniques.
  - Better(?) results through smoothing.
- We modify and extend the existing GloVe algorithm to Document Vectors.
  - Current State-of-the-art technique
  - Highly customizable.

Outline

Overview

Corpus	Vocabulary Size	Vector Space d
C_1	100,000	200
C_2	100,000	200

Corpora created by randomly sampling 75% of the English Wikipedia dataset (3.2M articles, 1.2B words)



Notation

- Let the two techniques for generating word vectors be s and t.
- s acts on a corpus C_s (with vocabulary V_s). t acts on a corpus C_t (with vocabulary V_t).
- Words are represented by w. Word vectors are represented as $v \in \mathbb{R}^d$
- Word Vectors generated using a technique s are represented by W_s $(|V_s| \times d)$.
- $\hat{W}_s$ represents the matrix of vectors within $\hat{V} = V_s \cap V_t$.
- Objective: Use $\hat{W}_s$ and $\hat{W}_t$ to find mapping $T()$ such that $T(W_s)$ is "near" W_t.

Outline

Vector Alignment

In our attempt to align these vector spaces, there are three differing levels of alignment we can achieve

- Congruent Mapping
- Globally Linear Equivalence
- Locally Linear Equivalence

Congruent Mapping

- This is the strictest equivalence.
- It assumes that we can transform W_s to W_t using only rigid transformations and scaling.
- i.e. All relative distances are preserved:

$$\forall w^i, w^j \in \hat{V}, \quad \frac{\mathrm{dist}(v_s^i, v_s^j)}{\mathrm{dist}(v_t^i, v_t^j)} = \alpha \tag{1}$$

Globally Linear Equivalence

- This relaxes the criterion of rigid transformations.
- It is still a global linear mapping.
- It assumes that we can transform W_s to W_t using only an affine transformation:

$$T(v^i) = M \cdot v^i + b \tag{2}$$

for an affine transformation M, and a translation vector b.

Globally Linear Approach

- We calculate the transformation matrix assuming global equivalence.
- i.e., we need to calculate M and b given $\hat{W}_s$ and $\hat{W}_t$, so as to minimize:

$$||M \cdot \hat{W}_s - \hat{W}_t||_F \tag{3}$$

- To solve this, we augment M with b to get a single matrix multiplication.

$$\left[\begin{array}{c} W_t \\ 1 \dots 1 \end{array} \right] = \left[\begin{array}{c|c} M & b \\ 0 \dots 0 & 1 \end{array} \right] \cdot \left[\begin{array}{c} W_s \\ 1 \dots 1 \end{array} \right] \tag{4}$$

- This can be solved by calculating the matrix pseudo-inverse:

$$\left[\begin{array}{c|c} M & b \\ 0 \dots 0 & 1 \end{array} \right] = \left[\begin{array}{c} W_t \\ 1 \dots 1 \end{array} \right] \cdot \left[\begin{array}{c} W_s \\ 1 \dots 1 \end{array} \right]^+ \tag{5}$$

(where the pseudo-inverse of A is defined as $A^+ = A^T(AA^T)^{-1}$)

Locally Linear Equivalence

- Try to achieve piecewise alignment for local regions.
- We look at the k-nearest neighbours of a word w_s^i within vocabulary $\hat{V}$.
- We can represent v_s^i as:

$$\hat{v}_s^i = \sum_{j=1}^{n} a_{ij} \cdot v_s^j \tag{6}$$

a_{ij} is nonzero if v_i and v_j are neighbours. We can approximate v_t^i using the following approach:

$$\tilde{v}_t^i \approx \sum_{j=1}^{n} a_{ij} \cdot v_t^j \tag{7}$$

Outline

Statistics

Glove

- Median NN Cosine Distance = 0.36
- Average Cosine Distance = 0.98
- Median NN Euclidean Distance = 4.7
- Average Euclidean Distance = 8.0

Word2Vec

- Median NN Cosine Distance = 0.29
- Average Cosine Distance = 0.97
- Median NN Euclidean Distance = 17.9
- Average Euclidean Distance = 34.9

Results of Globally Linear Approach

k	Euclidean Distance	Cosine Distance
201	25.39	0.698
401	1.43	0.026
601	1.13	0.017
1001	0.99	0.013
4001	1.19	0.023
10001	1.16	0.022

W_{11} to W_{21} mapping

k	Euclidean Distance	Cosine Distance
201	27.19	0.723
401	1.45	0.026
601	1.14	0.017
1001	0.99	0.013
4001	1.21	0.023
10001	1.17	0.022

W_{21} to W_{11} mapping

Results of Globally Linear Approach

contd.

k	Euclidean Distance	Cosine Distance
201	89.03	0.702
401	10.64	0.090
601	7.95	0.054
1001	6.65	0.038
4001	6.81	0.058
10001	6.41	0.052

W_{12} to W_{22} mapping

k	Euclidean Distance	Cosine Distance
201	58.47	0.86
401	4.53	0.20
601	3.46	0.141
1001	2.95	0.110
4001	3.83	0.24
10001	3.35	0.19

W_{12} to W_{21} mapping

Analysis of Globally Linear Approach

- Neighbourhoods are much tighter than median Neighbourhood.
- Mappings between these spaces are linear, though not in SO(200)
- Inverse mappings very similar, because same $\hat{V}$.
- Beyond $|\hat{V}| = 1$K, results seem to be worsening a little

Results of Local Linear Approach

k	Euclidean Distance	Cosine Distance
3	3.49	0.158
6	3.44	0.145
9	3.48	0.144
12	3.52	0.145
15	3.55	0.146

W_{11} to W_{21} mapping

k	Euclidean Distance	Cosine Distance
3	4.10	0.339
6	3.96	0.302
9	3.95	0.290
12	3.97	0.284
15	4.00	0.285

W_{21} to W_{11} mapping

Results of Local Linear Approach

contd.

k	Euclidean Distance	Cosine Distance
3	10.51	0.226
6	10.19	0.207
9	10.18	0.203
12	10.27	0.206
15	10.34	0.207

W_{12} to W_{22} mapping

k	Euclidean Distance	Cosine Distance
3	4.60	0.338
6	4.42	0.306
9	4.36	0.294
12	4.34	0.289
15	4.35	0.285

W_{12} to W_{21} mapping

Analysis

- Number of neighbours does not seem to matter much.
- Aligning vectors generated using the same technique perform better than aligning vectors generated using different techniques.
- Globally, the approach followed by GloVe and Word2Vec seems to be similar (a global alignment probably exists). Locally, there seem to be discrepancies.
- Smoothing.

Outline

The Word-Analogy Dataset

- This dataset has been used for testing the accuracy of word vectors generated.
- One needs to predict the correct word from the entire vocabulary
- Two subtasks: Semantic and Syntactic
- Examples:
  - Country Capitals: *Baghdad : Iraq* as *Rome : Italy*
  - Family: *boy : girl* as *dad : mom*
  -
  - Opposite: *sure : unsure* as *clear : unclear*
  - Past Tense: *danced : danced* as *fly : flew*
  - Present Participle: *dance : dancing* as *run : running*

Results

W_{11} to W_{21} mapping

Model	Syntactic Subtask	Semantic Subtask
Original s Word Vectors	39.0%	47.3%
Original t Word Vectors	38.6%	47.0%
Local Linear Approach	38.3%	58.8%

W_{21} to W_{11} mapping

Model	Syntactic Subtask	Semantic Subtask
Original s Word Vectors	38.6%	47.0%
Original t Word Vectors	39.0%	47.3%
Local Linear Approach	38.4%	55.6%

Results

contd.

W_{12} to W_{22} mapping

Model	Syntactic Subtask	Semantic Subtask
Original s Word Vectors	42.0%	51.1%
Original t Word Vectors	41.7%	48.4%
Local Linear Approach	37.0%	57.0%

W_{12} to W_{21} mapping

Model	Syntactic Subtask	Semantic Subtask
Original s Word Vectors	42.0%	51.1%
Original t Word Vectors	38.6%	47.0%
Local Linear Approach	33.0%	56.6%

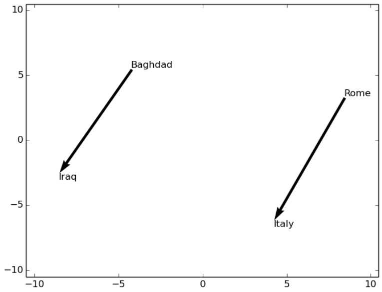
2D Projection

- We took the vectors of the four words (both, original and calculated).
- If they lie in one plane, Rank = 2.

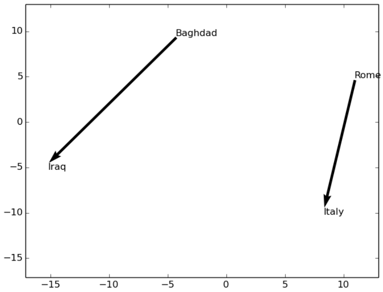
Example	Eigenvalues (Original Vectors)	Eigenvalues (Predicted Vectors)
Bangkok : Thailand Cairo : Egypt	[0.58, 0.35, 0.072]	[0.63, 0.33, 0.044]
Algeria : dinar Mexico : peso	[0.51, 0.41, 0.076]	[0.80, 0.19, 0.008]
father : mother groom : bride	[0.77, 0.14, 0.091]	[0.89, 0.09, 0.019]
known : unknown honest : dishonest	[0.63, 0.29, 0.089]	[0.62, 0.29, 0.088]

Examples
contd.

Predicted Vectors



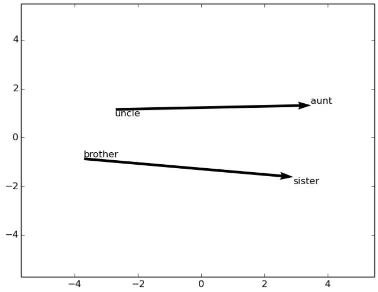
Original Vectors



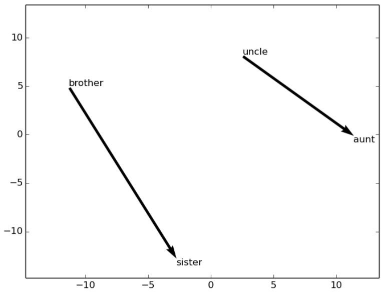
Baghdad : Iraq as *Rome : Italy*

Examples
contd.

Predicted Vectors



Original Vectors



brother : sister as *uncle : aunt*

Outline

Original GloVe Algorithm

- Uses word-word cooccurrence counts.
- The existing GloVe algorithm minimises the following cost function to generate word vectors:

$$J = \sum_{i,j=1}^{V} f\left(\tilde{C}_{ij}\right)\left(v_i^T \tilde{v}_j + b_i + \tilde{b}_j - \log \tilde{C}_{ij}\right) \tag{8}$$

- The algorithm performs least-squares regression to calculate v_i and $\tilde{v}_j$. It then adds v_i and $\tilde{v}_i$ for a word to get the final vector prediction.

Outline

Modifications

$$\begin{bmatrix} \tilde{C}_{ij} \end{bmatrix} \Rightarrow \left[\begin{array}{c|c} \tilde{C}_{ij} & C_{iA} \\ \hline C_{Aj} & C_{AB} \end{array}\right]$$

Figure: Modifying the GloVe co-occurrence matrix

Modifications

- We populate the newly added entries in the following manner:
  - For entries corresponding to a word and a paragraph (C_{iA}), we define a function F_{iA}.
  - For entries corresponding to a paragraph and a paragraph (C_{AB}), we define a function F_{AB}.
- The choice of F_{iA} and F_{AB} depend on us. Any equation that can approximate the co-occurrence counts will be satisfactory. Hence, we have quite a bit of freedom in this choice.

Outline

Baseline Comparison Algorithms

We tested the following baseline algorithms along with our approach:

- Bag of Words
- Paragraph Vectors - Distributed Memory Model
- Word Vector Averaging
- Word Vector Weighted Averaging
- Clustering based on Chinese Restaurant Process
  - Two variants
  - Two word-vector selection functions for each variant.

Outline

Testing on Wikipedia Corpora

contd.

Dataset	Number of Documents	Average Document Length	Average Number of Words per Document
English Wikipedia	106,497 (39693 articles)	119.75 characters	22.28
Hindi Wikipedia	119,079 (35499 articles)	100.45 characters	20.97

Our Results - Hindi Wikipedia Dataset

- Very little change with varying F_{iA}, however, modifying F_{AB} changes the accuracy of our results by a large percentage.

Accuracy	$F_{AB} = (15)$	$F_{AB} = (16)$	$F_{AB} = (17)$
$F_{iA} = (9)$	33%	44%	49%
$F_{iA} = (10)$	32%	44%	49%
$F_{iA} = (11)$	33%	44%	50%
$F_{iA} = (12)$	33%	43%	45%
$F_{iA} = (13)$	36%	45%	53%
$F_{iA} = (14)$	35%	43%	52%

Baseline Results - Hindi Wikipedia Dataset

Approach	Accuracy
Baseline	33%
BoW	60%
PV-DM	61%
Averaging	55%
Weighted Averaging	**64%**
CRP - Variant 1	51%
CRP - Variant 2	50%
CRP - Variant 1 (IDF Selection)	46%
CRP - Variant 2 (IDF Selection)	45%
GloVe Paragraph	53%

Baseline Results - English Wikipedia Dataset

Approach	Accuracy
Baseline	33%
BoW	49%
PV-DM	57%
Averaging	65%
Weighted Averaging	**68%**
CRP - Variant 1	53%
CRP - Variant 2	51%
CRP - Variant 1 (IDF Selection)	50%
CRP - Variant 2 (IDF Selection)	51%
GloVe Paragraph	47%

Outline

Testing on SemEval Corpora

- SemEval 2014 Task 3 dataset, – cross-level semantic similarity task (specifically, phrase to word similarity).
- 1000 training pairs, 1000 test pairs. Gold Standard:
    - 4, Very Similar
    - 3, Somewhat Similar
    - 2, Somewhat Related but not Similar
    - 1, Slightly Related
    - 0, Unrelated
- Evaluation is done by calculating the Pearson correlation.
- The current state-of-the-art for this subtask is a Pearson correlation of 0.457. The approach used language-specific resources such as POS tags, WordNet, and Lemmatization.

SemEval Results

- In this case, as expected, modifying the F_{AB} functions seemed to have little effect, while the F_{iA} functions changed the results considerably.

Approach	Pearson Correlation
Baseline	0.000
Meerkat (State-of-the-art)	0.457
PV-DM	**0.103**
Averaging	0.053
Weighted Averaging	0.063
GloVe Paragraph	0.075

- Our approach outperforms the naive averaging and weighted averaging approaches
- Its performance is close to the PV-DM approach, which is the current unsupervised state-of-the-art.

Summary

- Surprisingly good Global Alignment
- Word Analogy Task - Results improve in aligned spaces. Possibly due to smoothing.
- Generalizable extension for GloVe to paragraph vectors.
- Future Work
  - Alignment applications in Parallel corpora
  - Further analysis of smoothing.
  - Finalize on F_{iA} and F_{AB} for Paragraph Vector approach.

For Further Reading I

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning.
Glove: Global vectors for word representation..
Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014) 12 (2014): 1532-1543.

Mikolov, T., Chen, K., Corrado, G., and Dean, J.
Efficient estimation of word representations in vector space.
arXiv preprint arXiv:1301.3781 (2013).

Results

contd.

- Note that the results for the original vectors are worse than reported in the literature. This is due to at least two reasons:
  - The corpora these vectors are trained are smaller than the ones used by the papers
  - We used the default values of the training parameters (window, epochs, etc.). Tweaking them should give better results.
- The results of the Local Approach on the semantic subtask seem to be considerably better than the original vectors.
- We interpret this improvement as a result of smoothing on the local subspace around the target word vector. This is similar to what the GloVe approach does by adding the context word vectors to the word vectors, which gives a "small boost in performance, with the biggest increase in the semantic analogy task."

Modifications

To learn document vectors in the same subspaces as the word vectors, we propose the following modifications to the cooccurrence matrix:

- Append $|S|$ Rows and $|S|$ Columns to the matrix (where $|S|$ is the number of paragraphs for which we are calculating vectors).
- Thus, C is a $(|V| + |S| \times |V| + |S|)$-size square matrix.
- We populate the newly added entries in the following manner:
    - For entries corresponding to a word and a paragraph (C_{iA}), we define a function F_{iA}.
    - For entries corresponding to a paragraph and a paragraph (C_{AB}), we define a function F_{AB}.
- Once we have written these functions F_{iA} and F_{AB} in terms of C_{ij} (word-word cooccurrence counts), we will be able to populate the entire matrix.
- The choice of F_{iA} and F_{AB} depend on us. Any equation that can approximate the co-occurrence counts will be satisfactory. Hence, we have quite a bit of freedom in this choice.

Clustering based on Chinese Restaurant Process (CRP)

- This is a modified form of the Word Vector Averaging approach. It seeks to select only relevant words and takes the weighted average of those words, rather than all words in the article.
- The variants modify the CRP clustering algorithm to be more (or less) selective in creating new clusters.
- Each variant originally returns a list of vectors. Our selection functions select one of these vectors.
  - Standard Selection: For two lists of vectors (while comparing two documents), take every combination of vectors, and return the minimum distance.
  - IDF Selection: For every vector in the list of vectors, take the average frequency within the entire corpus of the words averaged in that vector. Select the vector with the lowest average frequency, and return it.

Equations for F_{iA}

$$C_{iA} = 0 \tag{9}$$

$$C_{iA} = C_i \times \prod_{j=1}^{n} C_{a_j} \tag{10}$$

$$C_{iA} = C_i \times \prod_{j=1}^{n} C_{ia_j} \tag{11}$$

$$C_{iA} = C_i \times C_{ia_1} \times \prod_{j=1}^{n-1} \frac{C_{a_j a_{j+1}}}{C_{a_j}} \tag{12}$$

$$C_{iA} = C_i \times \sum_{j=1}^{n} \left(C_{ia_j} \times \left(1 - \frac{C_{a_j}}{\sum_k^V C_k} \right) \right) \times \frac{1}{n} \tag{13}$$

$$C_{iA} = C_i \times \frac{\sum_{j=1}^{n} C_{ia_j}}{n} \tag{14}$$

Equations for F_{AB}

$$C_{AB} = 0 \tag{15}$$

$$C_{AB} = \sum_{w}^{A \cap B} C_w \tag{16}$$

$$C_{AB} = |\{a_1, a_2, \ldots, a_n\} \cap \{b_1, b_2, \ldots, b_m\}| \times \frac{\sum_k^V C_k}{|V|} \tag{17}$$

Testing on Wikipedia Corpora

- To test these approaches, we created our own datasets using the English and Hindi Wikipedia corpora.
- We selected 35,000 - 40,000 random Wikipedia articles satisfying the following conditions:
  - The article has more than 3 paragraphs
  - Each paragraph has a length larger than 5 words
- For each query in our test, we create a triplet of paragraphs: two paragraphs are selected from the same Wikipedia article, whereas the third paragraph is randomly selected from the rest of the collection. Our goal is to identify the paragraph not belonging to the Wikipedia article.
- A baseline approach should give an accuracy of 33%.