(General) Word Embeddings Explained or Why Word Embeddings Allow to Make Coffee

Rémi Gilleron

Inria Lille - Nord Europe & LIFL & Univ Lille

Nov. 2016

Base Material

Today Presentation

Among a bunch of papers defining methods or trying to compare approaches or trying to explain why word embeddings work, we choose to present papers by Arora et al:

- RAND-WALK: a latent variable model approach to word embeddings, S. Arora, Y. Li, Y. Liang, T. Ma, A. Risteski, TACL, 2016
- Linear Algebraic Structure of Word Senses, with Applications to Polysemy, S. Arora, Y. Li, Y. Liang, T. Ma, A. Risteski, still unpublished.

Plan

What do I mean by (general) word embeddings

Word Embeddings

- A word embedding is a mapping from a vocabulary (a set of words) into the set $\mathbb{R}^d$ of tuples of real numbers (vectors)
- considered as representations of words for solving NLP tasks
- which is along the "Learning Representations" route of research

Why "general" ?

- Because we focus on task-independent learning methods,
- i.e. we consider unsupervised methods

The problem

Origin

- Find the meaning of words in Linguistics and Psychology
- Find semantic similarity in Information Retrieval

Definition

- Given a corpus of texts $\mathcal{C}$,
- define a vocabulary V_w of words and a vocabulary V_c of contexts,
- and produce a matrix W in $\mathbb{R}^{V_w \times d}$,
- where each row v_w is a representation of a word w, and,
- optionally, a column is a representation of something,
- such that it works well (makes coffee).

The problem in Magnet

Finding Word Representations

- for multi-lingual parsing (Mathieu),
- for coreference resolution (Thibault, Orange),
- for NLP problems and their applications

The Magnet Approach

- using linguistic contexts
- adapting general embeddings
- for solving NLP tasks for both research and collaborative projects
- Magneto: complete software for constructing embeddings: beyond english, beyond words, beyond windows, evaluation framework, ...
  Main contributors: Nathalie and François

Methods for Computing Word Embeddings

Count-based

- $W(w, c)$ is the number of occurrences of w in c in $\mathcal{C}$,
- $PMI(w, c) = \log(\frac{P(c|w)}{P(c)})$ and its variants *PPMI*, shifted *PPMI*, ...

Dimensionality Reduction

applied to the count-based matrix : *PCA*, *SVD*, *NMF*, ...

Learning

- Word2vec (shallow NN): c is a symmetric window, either predict the central word (*CBOW*) or predict a word in the window (*SkipGram*)
- Glove: online learning for an optimization criterion derived from the factorization of the shifted *PMI* matrix

Comparison

Evaluation Methods

The methods are unsupervised. They are compared on

- Synonym Detection, Similarity Judgments, Semantic and Syntactic Categorization, ...
- Analogy (Mikolov13): find b^* in $a : a^*, b : b^*$ as man:woman, king:? for syntactic or semantic analogies.

Comparison

- No clear conclusion because of: unsupervised methods, non task specific, choice of $\mathcal{C}$, of V_w, of V_c, of d, of the method, . . . , but
- word2vec works well and allows to consider very large sets $\mathcal{C}$

Among many works, we choose to consider the work by Arora et al trying to compare methods and to explain why analogies can be solved.

Plan

Objective of the paper

Propose a probabilistic model of text generation

in order to

- explain non linear models for word embeddings such as *PMI* and its variants, *word2vec* and *Glove*,
- give hints on the choice of hyper parameters,
- experimentally show that latent word vectors are fairly uniformly dispersed,
- explain why embeddings contain linear structure allowing to solve word analogies.

raising the question: Which comes first ? The coffee bean or the coffee plant

The Model

A probabilistic dynamic process for text generation

- at each time step t, we suppose a discourse vector $c_t \in \mathbb{R}^d$,
- each word is supposed to have a time-invariant latent representation $v_w \in \mathbb{R}^d$ which captures its correlation with c_t,
- the loglinear word production model is defined by

$$Pr[\text{w emitted at time t} \mid c_t] \propto exp(< c_t, v_w >)$$

- the vector c_t does a slow random walk meaning that c_{t+1} is obtained by adding a small displacement vector to c_t

So far left unprecise

- From where the v_w come from ?
- What about the slow random walk ?

Choosing the v_w and the random walk

Spherical Gaussian Distribution

- every component of $v_w \in \mathbb{R}^d$ is drawn according to $N(0, 1)$
- the normalized random vector is uniformly distributed on the sphere

Prior for the v_w

The v_w are obtained from i.i.d draws generated by $v = s\tilde{v}$ where

- $\tilde{v}$ is drawn from the spherical Gaussian distribution
- s is a scalar random variable with expectation $\tau = \Theta(1)$ and upper bounded by a constant κ

Conditions on the random walk

- The stationary distribution of the random walk is the uniform distribution over the unit sphere
- Each movement of the discourse vector is at most $\epsilon/\sqrt{d}$ in l_2-norm

First results

For word probabilities – Self-normalization in loglinear models

Recall that $Pr[w_t|\ c_t] \propto exp(< c_t, v_w >)$, it can be proved that

- word probabilities satisfy a power law
- $Z_c = \sum_w exp(< c, v_w >)$ is close from some constant Z

For co-occurrence probabilities

Theorem For window size q, let $p_q(w, w')$ be the co-occurrence probability in windows of size q, then

$$log(p_q(w, w')) = \frac{\|v_w + v_{w'}\|_2^2}{2d} + log(\frac{q(q-1)}{2}) \pm \epsilon$$

$$PMI(w, w') = \log(\frac{p_q(w, w')}{p_q(w)p_q(w')}) = \frac{< v_w, v_{w'} >}{d} + log(\frac{q(q-1)}{2}) \pm O(\epsilon)$$

Plan

Training Objective and Their Algorithm

Assuming the generative model, the previous formulas for $log(p_q(w, w'))$ and $PMI(w, w')$, and more hypothesis on the random walk

Training Objective with co-occurrence probabilities

the maximum likelihood values for the word vectors in $\mathbb{R}^d$ correspond to

$$\min_{\{v_w\},B} \sum_{w,w'} X_{w,w'}(log(X_{w,w'}) - \|v_w + v_{w'}\|_2^2 - B)^2$$

Training Objective with PMI

$$\min_{\{v_w\},C} \sum_{w,w'} X_{w,w'}(PMI(w, w') - < v_w, v_{w'} >)^2$$

Algorithm

- $X_{w,w'}$ is chosen to be $min\{X_{w,w'}, X_{max}\}$ with $X_{max} = 100$
- solved by adaptive gradient descent (Adagrad)

The Glove Model – Pennington, Socher, Manning

Empirical derivation of the model

The authors introduce the importance of the ratios $\frac{P(c|w)}{P(c|w')}$ as $\frac{P(c|king)}{P(c|queen)}$, use scalar product to transform vectors into scalars, and the exponentiation to transform sums into products. They obtain $\forall w \in V_W, \forall c \in V_C, < v_w, v_c > + b_w + b_c = \log(X_{w,c})$

Glove algorithm

$\min_{\{v_w, v_c, b_w, b_c\}} \sum_{w,c} f(X_{w,c})(log(X_{w,c}) - < v_w, v_c > - b_w - b_c)^2$
where $f(X_{w,c}) = min\{(\frac{X_{w,c}}{X_{max}})^{\frac{3}{4}}, 1\}$ and solved by Adagrad

Glove (simplified) as a factorization problem

If we fix $b_w = log(X_w)$ and $b_c = log(X_c)$, the problem is equivalent to factorizing the log-count matrix shifted by $\log(D)$

$$M^{\log(X_{w,c})} \approx WC^t + log(D)$$

The Rand-walk model and the Glove model

Reminder

- $\min_{\{v_w\},B} \sum_{w,w'} X_{w,w'}(log(X_{w,w'}) - \|v_w + v_{w'}\|_2^2 - B)^2$
- $\min_{\{v_w,v_c,b_w,b_c\}} \sum_{w,c} f(X_{w,c})(log(X_{w,c}) - < v_w, v_c > -b_w - b_c)^2$

Claims

- Contexts are windows, i.e. we consider co-occurrences of words w and w' in a window of size q,
- They consider $f(X_{w,w'}) = min\{X_{w,w'}, X_{max}\}$
- They argue a formal derivation of the Glove criterion by choosing $b_w = \|v_w\|_2^2$.

Rand-Walk and Mikolov's methods

Rand-Walk and CBOW

- CBOW is a neural network method for creating word embeddings. The learning criterion is to predict the central word of a window.
- They show that CBOW can be formulated as a MAP problem in their model

Skip-Gram with negative sampling (Levy and Goldberg)

- SGNS is a neural network method for creating word embeddings. The learning criterion is to predict a word in the window containing the current word.
- SGNS construct matrices W and C maximizing $< v_w, v_c >$ for pairs (w, c) in D and minimizing it for stochastically corrupted pairs.
- Levy and Goldberg show that SGNS construct matrices W and C such that $WC^t = M^{PMI} - log(k)$. Note that the factorization's loss is not l_2.

Plan

Solving Analogies with Word Embeddings

The Analogy Problem

Given words a, a^* and b, find b^* in $a : a^*, b : b^*$,

i.e. a is to a^* as b is to b^*
as in the famous *man* is to *woman* as *king* is to ??

The Analogy Problem and Word Embeddings

- introduced by Mikolov
- showing that word embeddings allow to solve analogies
- without knowledge of the type of analogy (some cheating anyway)

Reformulating Analogies using Word Embeddings

Given words a, a^* and b and their embeddings, find b^* in $a : a^*, b : b^*$

3COSADD formulation

- reformulated as $argmax_{b^* \in V_W} sim(b^*, b - a + a^*)$
- using Cosine similarity as $argmax_{b^* \in V_W} \cos(b^*, b - a + a^*)$
- using basic algebra and normalized vectors as

$$argmax_{b^* \in V_W}(\cos(b^*, b) - \cos(b^*, a) + \cos(b^*, a^*))(3COSADD)$$

i.e. b^* is similar to b and to a^*, but dissimilar from a as in queen is similar to king and to woman and dissimilar from man

PAIRDIRECTION formulation

- states that the direction of the transformation is conserved leading to
- $argmax_{b^* \in V_W}(\cos(b^* - b, a^* - a))(PAIRDIRECTION)$
- omitting the similarity between b^* and b

Why embeddings allow to solve analogies by Arora at al

Facts

- the difference between the best and the 2nd best solution of 3COSADD is small
- low dimensional embeddings work better than count-based embeddings

Their arguments

- Relations $\equiv$ Lines: for a relation R (gender, royalty, ...), every pair a, a^* satisfying R, there is a direction vector μ_R such that $v_{a^*} - v_a = \mu_R + nv$, where nv is a noise vector
- Due to the isotropy of word vectors and due to the low dimensionality, the noise is small
- allowing to use the best solution of 3COSADD

Why embeddings allow to solve analogies by Levy and Goldberg

They experimentally study regularities in embeddings

- comparing the 3COSADD and the PAIRDIRECTION criteria, and
- introducing a new criterion

$$argmax_{b^* \in V_W} \frac{\cos(b^*, b)\cos(b^*, a^*)}{\cos(b^*, a) + \epsilon} (3COSMUL)$$

instead of $argmax_{b^* \in V_W}(\cos(b^*, b) - \cos(b^*, a) + \cos(b^*, a^*))$

Their experimental results

- 3COSMUL is always better than 3COSADD,
- Analogy solving is not limited to neural word embeddings or to low dimensional embeddings

Plan

Model Verification

Partition function

Recall that $Pr[w_t|\ c_t] \propto exp(< c_t, v_w >)$, the normalization factor is $Z_c = \sum_w exp(< c, v_w >)$. they verify that for the learned v_w

- Over randomly chosen c's, Z_c-values are concentrated
- which is also true for Glove and CBOW embeddings

Isotropy w.r.t. singular values

The matrix of learned word vectors behaves like a random matrix, i.e. the ratio between quadratic mean of singular values and the minimum non zero singular value is a small constant (also for Glove and CBOW)

Squared norms versus word frequencies

Recall that $log(p(w)) = \frac{\|v_w\|_2^2}{2d} - \log(Z) \pm \epsilon$. They verify that, for the learned v_w, the linear relationship between $\|v_w\|_2^2$ and $log(p(w))$ is observed

Performance on analogy tasks

Comparison with other methods

Results are close to state of the art given by Glove, CBOW and skipgram

Verifying Relations $\equiv$ Lines

Based on $v_{a^*} - v_a = \mu_R$, they design a cheating solver as follows

- With some examples of pairs (a, a^*) for a relation R, learn the direction by rank 1 SVD
- find b^* s.t. $b^* - b$ closest to the learned direction

They get up to 10% improvement

They give another cheater solver based on clustering of the $v_{a^*} - v_a$ vectors

Plan

Word Embeddings and Polysemous Words

Base Idea

Word embeddings contain word senses. Multiple word senses reside within a word embedding in linear superposition

An insightful experiment

Consider a word embedding over V_W constructed from a corpus $\mathcal{C}$

- Choose randomly two words w_1 and w_2,
- in $\mathcal{C}$, replace every occurrence of w_1 and w_2 by w,
- construct a new word embedding
- check whether v_w is in the subspace spanned by v_{w_1} and v_{w_2}

The cosine of the angle was found to be 0.97 with sd 0.02.
Therefore it can be asserted that $v_w = \alpha v_{w_1} + \beta v_{w_2}$
For frequency ratio r, $\alpha \approx 1$ and $\beta \approx 1 - c \log(r)$

Extracting Word Senses

The previous experiment suggests that for a polysemous word such as spring

$$v_{spring} = \alpha_1 v_{spring1} + \alpha_2 v_{spring2} + \dots$$

Model and algorithm

- From the randwalk paper, they assume a direction for each discourse
- Given word vectors v_w, a sparsity parameter k, and an upper bound m, find a set of unit vectors $A_1, \dots, A_m$ such that

$$v_w = \sum_{j=1}^{m} \alpha_{w,j} A_j + \eta_w$$

  where at most k of the $\alpha_{w,j}$ are nonzero and η_w is a noise vector
- sparse coding using k-SVD algorithm

Experiments

The size of V_W was set to 60,000, d was set to 300, $\mathcal{C}$ is English Wikipedia dump, they use their algorithm.

Results

- Best value of basis size was found to be $m = 2000$
- Best value of the sparsity parameter was found to be $k = 5$
- For the word spring, closest words to the word senses are

$spring_1$	beginning	until	months	earlier	year	last	
$spring_2$	dampers	brakes	suspension	absorbers	wheels	damper	
$spring_3$	flower	flowers	flowering	flagrant	lilies	flowered	
$spring_4$	creek	brook	river	fork	piney	elk	
$spring_5$	humid	winters	summers	ppen	warm	temperatures	

- hierarchy of senses setting $m = 200$ and $k = 2$

Conclusion