Provable word embedding for the skip-gram word2vec model

Anonymous

Abstract

To be written

(In magenta color is our comments to drive the writing; in plain black color will be our text)

Introduction

What is the problem we focus on? Word-embedding, why it is important, where it is used, some references from public science & media showing its significance.

What is the current state of the art & what are the shortcomings? Here, we should describe how people solve this problem, what are the tools they use (SGD, Riemannian, etc.) and what are the problems with these methods. We shouldn't spend that much space on related work as there will be a Related Work section next.

What is our perspective? Computational. We need to stress out that we do not focus on finding a better linguistic metric, but given word2vec model interpretation as matrix factorization, we identify the problems of using classical methods that involve huge matrix manipulations, and we propose alternatives. At the same time, we are interested in proposing theory that justifies partially what we observe in practice. E.g., here we should say that current approaches are expensive computationally, as well as non-convex with no theory.

Word embedding represents one of the most successful applications of unsupervised learning. It has shown geralization power in varities of NLP tasks, part-of-speech tagging (Abka 2016), document level metric learning(Kusner et al. 2015), machine translation (Mikolov et al. 2013)(Zou et al. 2013) etc.

In this paper, we train Skip-Gram model with negative sampling (SGND) under the framework of Bi-factorized gradient descent (BFGD). We show that with BFGD, which does not require singular value decomposition at each iteration, we can obtain similar liguistic performance compared to Splitting Projection Algorithm. And BFGD is able to train the model efficiently as the size of a corpus goes up, since it does not require singular value decomposition at each iteration. As we will see, with update rule in word2vec a special version of stochastic gradient descent under BFGD framework, this discussion opens a critical topic on how to train SGNS both efficiently and effectively — how to design a good stochastic/batched version of BFGD and train Skip-Gram model without loss of performance. What are our contributions? We need to write them down in bullets; this will be written after we have all the rest.

Background & Related Work

Set up the problem: notation + mathematical description What are the works before us: a more detailed description. What did they contribute, how did they evolve the field?

What questions are still open?

The Skip-Gram model is introduced in (Mikolov et al. 2013), which assumes a corpus of words w_1, w_2 ... w_n and correponding contexts. For every individual word w_i, a context c is defined as a word within a $L-$sided window surrounding it, i.e. $w_{i-L}, \cdots, w_{i-1}, w_{i+1}, \cdots, w_L$. Following notations in (Levy and Goldberg 2014), we denote $\#(w,c)$ as number of word-context pairs and $\#(w)$ are abbreviation for

$$\#(w) = \sum_{c'} \#(w,c'), \quad \#(c) = \sum_{w'} \#(w',c) \tag{1}$$

It is also convenient to define the set of observed word-context pairs as D, and $|D| = \sum_{w,c} \#(w,c)$

The negative sampling takes a word-context pair and samples k negative pairs (w, c_N) aligning with it, where k is a hyperparameter and equals 5 in experiments. In every negative pairs, the context is generated from the context distribution from the corpus, which is:

$$c_N \sim P_D(c) = \frac{\#(c)}{|D|} \tag{2}$$[1]

Therefore for every word-context pair in vocabulary, the SGNS objective is:

$$l_{SGNS}(w,c) = \#(w,c)\log[\sigma(w^\top c)] + k\cdot\mathbb{E}_{c_N\sim P_D}\log[\sigma(-w^\top c_N)] \tag{3}$$

[1] In word2vec implementation, $c_N \sim P_D(c) = \frac{\#(c)^{3/4}}{|D|}$, but in mathematics view SGNS is still doing factorization

When the online learning goes through all word-context pairs in corpus, SGNS model learns distributed word representation by maximizing the following objective function:

$$\max_{w,c}\sum_{w,c}\#(w,c)\log\sigma(w^\top c)+k\cdot\mathbb{E}_{c_N\sim P_D}[\log\sigma(-w^\top c_N)] \tag{4}$$

Matrix Factorization

As what is commonly accepted in literature, some of the ”simplest” word embedding models can be viewed as matrix factorizations (Li et al. 2015)(Mikolov et al. 2013): SGNS is factorizing *shifted* Pointwise Mutual Information matrix, Noise-Contrastive Estimation (Gutmann and Hyvärinen 2010) is factorizing *shifted* log-conditional-probability matrix for instance. Despite different views on the contrary(Arora et al. 2015), it has been of parculiar interests to view word embedding views as matrix factorization and studying the landscape of those objectives (Li et al. 2015)(Mimno and Thompson 2017).

Project-Splitting algorithm on SGNS

On (Fonarev et al. 2017) illustrates a general two-step scheme for training SGNS word embedding model and suggested a search of a solution in the low-rank form via Riemannian optimization framework.

Linguistic scores

Our approach

Description of the algorithm, details and discussion on initialization + step size, maybe already here have some plots to show how these behave (without giving away comparison results, just showing what is their trend in

In this paper, we follow literature dicussions characterizing SGNS as a matrix factorization problem (Levy and Goldberg 2014)(Levy, Goldberg, and Dagan 2015). The expectation term $\mathbb{E}_{c_N\sim P_D}[\log\sigma(-w^\top c_N)]$ can be explicitly expressed as:

$$\mathbb{E}_{c_N\sim P_D}[\log\sigma(-w^\top c_N)]=\sum_{c_N}\frac{\#(c_N)}{|D|}\log\sigma(-w^\top c_N) \tag{5}$$

And one can show that SGNS objective 4 is factorizing the following matrix (Levy and Goldberg 2014):

$$X=W^\top C=PMI(w_i,c_j)-\log k \tag{6}$$

where columns in W and C are w_i and c_i respectively. And in our setting, the vocabulary size is V , and the hidden layer size is d, so both W and C are $d\times V$.

Different from projector-splitting scheme(Fonarev et al. 2017) which shows advantages in optimizing a low-rank X on SGNS model, we observed that the matrix form SGNS objective is both smooth and convex in X. And with the explicit expression of SGNS objective $L_{SGNS}(X)$:

$$L_{SGNS}(X)=\sum_{w,c}\{\#(w,c)\log\sigma(w^\top c)+k\cdot\sum_{c_N}\frac{\#(c_N)}{|D|}[\log\sigma(-w^\top c_N)]\} \tag{7}$$

$L_{SGNS}(X)$ is $L-$smooth with lipschitz constant

$$L=\frac{1}{4}\|\{\#(w,c)+k\frac{\#(w)\#(c)}{|D|}\}_{w,c}\|_F$$

Then we can borrow ideas from Bi-factorized gradient descent (Park et al. 2016)

Algorithm 1 BFGD on Skip-Gram Model with Negative Sampling

```
procedure BFGD($W_0, C_0, \eta, K$)    ▷ $W_0, C_0$ are initial encoding and decoding matrices, $\eta$ is the step size and $K$ is the total number of iterations
    $W \leftarrow W_0$
    $C \leftarrow C_0$
    for $i \leftarrow 1, \cdots, K$ do
        Calculate gradient of loss $\nabla L(W^\top C)$
        $W \leftarrow W + \eta \cdot C\nabla L(W^\top C)$
        $C \leftarrow C + \eta \cdot W\nabla L(W^\top C)^\top$
    end for
    return W, C
end procedure
```

In the BFGD agorlithm 1, the complexity for each iterations is $O(V^2d)$: with hidden dimension fixed, the running time scales quadratically with the size of the vocabulary. As a comparision, Project-Splitting(Fonarev et al. 2017) operates QR factorization in updating parameters and requires $O(V^3)$ unit operations at each iteration. And in experiments, our see our approach is at least two times faster than the Project-Splitting scheme.

Here, we should have a figure with the algorithm's steps etc.

Experimental results

We will move a bit unconventionally and show first some experimental results: this is what we are currently working on

Experimental Settings In experiments, we trained skip-gram negative sampling with Bi-Factorized gradient descent on two corpora: ”enwik9” corpus (Mahoney 2011) and New York Times corpus (NYT) (Sandhaus 2008). The ”enwik9” contains the first billions bytes of the Wikipedia dump on Mar. 3, 2006. We ignore words that appear less than 100 times in this dump and train a model with vocabulary size 37,360. The New York Times Annotated Corpus contains over a million articles tagged with metadata. These articles are published between 1987 and 2007. We pick articles from 2000 to 2005 for training. We preprocessed the data with Stanford CoreNLP tookenizer (Manning et al. 2014). It has 13,567,603 sentences and we use a dictionary of the 40,000 most frequent words from this subcorpus.

In order to reduce training noise from frequent words: we do subsampling and ignore a word w in a sentence with a probability

$$P(f(w))=1-(\sqrt{\frac{f(w)}{t}}+1)\cdot\frac{t}{f(w)} \tag{8}$$

dim		sem	syn	wordsim	men	simlex	murk
100	SGD						
	PS						
	BFGD						
300	SGD						
	PS						
	BFGD						
300	SGD						
	PS						
	BFGD						

Table 1: Comparison of different methods on liguistic scores, on different dimensions

dataset		sem	syn	wordsim	men	simlex	murk
NYT	SGD						
	PS						
	BFGD						
enwik	SGD						
	PS						
	BFGD						

Table 2: Comparison of different methods on liguistic scores, on different dataset, we kept dimension $d = 300$

where where t, $f(w)$ are subsampling threshold and the frequency of the word respectively. We have to point out equation 8 is used in word2vec[2] and is an adapted version of subsampling in (Mikolov et al. 2013).

Evaluation

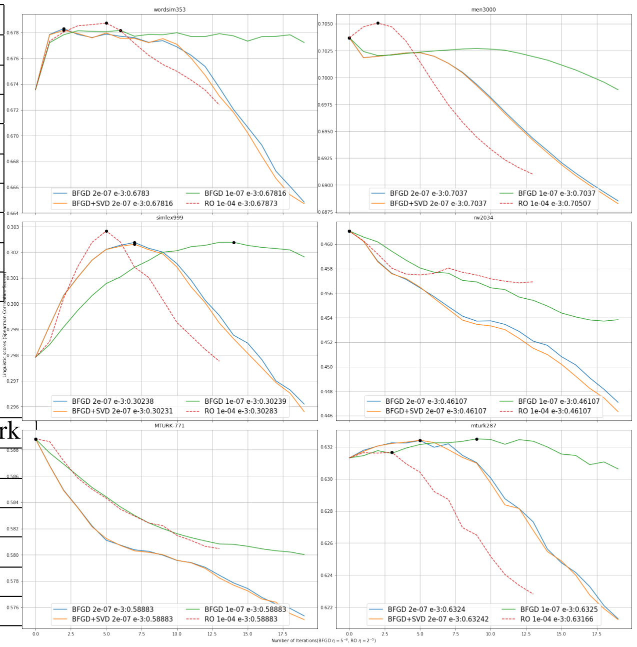
We illustrate equivalence of BFGD and PS analogy tasks and similarity tasks.

Analogy The analogy test datasets come from Google analogy (Mikolov et al. 2013) and MSR's analogy dataset (Mikolov, Yih, and Zweig 2013). Google's analogy dataset contains 19,544 questions of two types — semantic and syntatic analogies. Semantic questions are analogies in semantic sense, as "Greece is to Athens as Iraq is to Baghdad", whereas syntactic questions are related to tokens and their syntactic variants, as "amazingly is to amazing as apparently is to apparent". MSR's analogy test set has 8,000 morpho-syntactic analogy questions. it is composed of the syntactic kind of tasks.

Theoretical guarantees

This is where we will try to focus on after we have fixed the experiments.

- What are the properties of the objective?
- What is known out there w.r.t. theory? What can we reuse for our algorithm?
- Initialization: what can we say about it? Any theory?



- Are there local minima for the non-convex problem we have?
- what about the stochastic version? SVRG version? Are there prior results on this?
- Can we use momentum/acceleration? Can we gain theoretically?

In this section, we provide theoretical guarantees for the convergence of BFGD algorithm. As shown in analysis above, SGNS objective L_{SGNS} is smooth as well as strictly convex, we focus our discussions based on this setting.

Since BFGD gives the best results starting from SPPMI matrix, we provide a convergence guarantee for SPPMI initialization.

From matrix factorization on SGNS objective (Levy and Goldberg 2014) sparse SPPMI matrix and dense PPMI matrix. PPMI is X^*, the optimal solution for SGNS. SPPMI is the X^0, with its rank$-r$ approximation $X_r^0 = W_0^\top C_0$ the starting point.

From defintion of SPPMI and PPMI matrix we have Assume for any number of rows in X, $L - smoothness$ holds.

FGD

On one iteration X is $V \times V$:

$$U^+ = U - \eta(\nabla f(X)U)^i = U - \eta(\nabla f)^i U = U - \eta \begin{bmatrix} 0 \\ 0 \\ (\nabla f)_i U \\ 0 \end{bmatrix}$$

Here ∇f is a shorthand for $\nabla f(X)$ and η is given by

$$\eta = \frac{1}{16(L\|X^0\|_2 + \|\nabla f(X^0)\|_2)}$$

We are using $(\nabla f)^i$ for the full $V \times V$ matrix and $(\nabla f)_i$ for a random component/row:

[2] https://code.google.com/archive/p/word2vec

$$(\nabla f)^i = \begin{bmatrix} 0 \\ 0 \\ (\nabla f)_i \\ 0 \end{bmatrix}$$

For simplification, we define $\Lambda = I - \frac{\eta}{2} Q_U Q_U^\top (\nabla f(X))^i$, which corresponds to the stochastic gradient $(f(X))^i$. It is easy to check that

$$X^+ = X - \eta (\nabla f(X))^i X \Lambda - \eta \Lambda^\top X (\nabla f(X))^i \tag{9}$$

For the randomly picked i we have:

$$\begin{aligned} &\mathrm{Tr}(\nabla^\top f (\nabla f)^i X \Lambda) \\ &= \mathrm{Tr}(\begin{bmatrix} \cdots (\nabla f)_i^\top \cdots \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ (\nabla f)_i \\ 0 \end{bmatrix} X \Lambda) \end{aligned} \tag{10}$$

$$\begin{aligned} &\mathrm{Tr}(\nabla^\top f \Lambda^\top X (\nabla f)^i) \\ &= \mathrm{Tr}((\nabla f)^i \nabla^\top f \Lambda^\top X) \\ &= \mathrm{Tr}(\begin{bmatrix} 0 \\ 0 \\ (\nabla f)_i \\ 0 \end{bmatrix} \begin{bmatrix} \cdots (\nabla f)_i^\top \cdots \end{bmatrix} \Lambda^\top X) \end{aligned} \tag{11}$$

Thus

$$\begin{aligned} &\mathbb{E}[\mathrm{Tr}(\nabla^\top f \Lambda^\top X (\nabla f)^i)] = \mathbb{E}[\mathrm{Tr}(\nabla^\top f \Lambda^\top X (\nabla f)^i)] \\ &= \frac{1}{V} \mathrm{Tr}(\nabla f(X) \nabla f(X) X \Lambda) \end{aligned} \tag{12}$$

Conditioned on X, we have

$$\begin{aligned} &\mathbb{E}[f(X^+)] - f(X) \\ &\le \mathbb{E}[\langle \nabla_i f, X^+ - X \rangle] + \frac{L}{2} \mathbb{E}[\|X^+ - X\|_F^2] \\ &= -\frac{2\eta}{V} \mathrm{Tr}(\nabla f(X) \nabla f(X) X \Lambda) + \frac{L\eta^2}{2V} \|\nabla f(X) X \Lambda + \nabla \Lambda X f(X)\|_F^2 \\ &\overset{(i)}{\le} -\frac{2\eta}{V} \mathrm{Tr}(\nabla f(X) \nabla f(X) X \Lambda) + \frac{2L\eta^2}{V} \|\nabla f(X) X \Lambda\|_F^2 \\ f(X) - f(X_r^*) &\le -\frac{2\eta}{V} \mathrm{Tr}(\nabla f(X) \nabla f(X) X \Lambda) + \frac{2L\eta^2}{V} \|\nabla f(X) U\|_F^2 \|X\|_2^2 \|\Lambda\|_2^2 \\ &\overset{(ii)}{\le} -\frac{2\eta}{V} \mathrm{Tr}(\nabla f(X) \nabla f(X) X \Lambda) + \frac{\eta}{7V} (\frac{33}{32})^2 \|\nabla f(X) U\|_F^2 \\ &\overset{(iii)}{\le} -\frac{62\eta}{32V} \|\nabla f(X) U\|_F^2 + \frac{\eta}{7V} (\frac{33}{32})^2 \|\nabla f(X) U\|_F^2 \\ &\le -\frac{17\eta}{10V} \|\nabla f(X) U\|_F^2 \end{aligned} \tag{13}$$

(i) is from triangle inequality, (ii) is from equation (18) in (Park et al. 2016) and (iii) is from Lemma $A.5$ in (Park et al. 2016) which implies $\|\Lambda\|_2 \le \frac{33}{32}$ and $\sigma_V(\Lambda) \ge \frac{31}{32}$.

Which is to say for the optimal X_r^*:

$$\mathbb{E}[f(X^+)] - f(X_r^*) \le f(X) - f(X_r^*) - \frac{17\eta}{10V} \|\nabla f(X) U\|_F^2 \tag{14}$$

By equation(18) in (Tu et al. 2015) we have

$$f(X) - f(X_r^*) \le \frac{5}{2} \|\nabla f(X)\|_F \mathrm{DIST}(U, U_r^*) \tag{15}$$

Thus with

$$\delta^+ = \mathbb{E}[f(X^+)] - f(X_r^*) \quad \delta = \mathbb{E}[f(X)] - f(X_r^*)$$

We can easily see

$$\begin{aligned} \delta^+ &\le \delta - \frac{17\eta}{10V} \|\nabla f(X) U\|_F^2 \\ &\le \delta - \frac{17\eta}{10V} \times (\frac{2}{5})^2 \cdot \frac{\delta^2}{\mathrm{DIST}^2(U, U_r^*)} \end{aligned} \tag{16}$$

$$\frac{1}{\delta^+} \le \frac{1}{\delta} + \frac{\eta}{5V \cdot \mathrm{DIST}(U, U_r^*)} \tag{17}$$

Word Embedding

Define $U = \begin{bmatrix} W \\ C \end{bmatrix}$, $X = WC^T$

$$\nabla_U f = \begin{bmatrix} \nabla_X f C \\ \nabla_X^T f W \end{bmatrix} = \begin{bmatrix} 0 & \nabla_X f \\ \nabla_X^T f & 0 \end{bmatrix} \begin{bmatrix} W \\ C \end{bmatrix} \tag{18}$$

Going back to the skip gram iterations, which can be seen as updating the matrix U.

Assume from U to U^+, we observed a word-context pair (w_i, c_j) with k negative samples oriented to the oriented word w_i, namely $(w_i, c_{n1}), \cdots, (w_i, c_{nk})$, Define matrix gradient matrix G as follows:

$$G = \begin{bmatrix} 0 & G' \\ G'^T & 0 \end{bmatrix} \tag{19}$$

where

$$\begin{aligned} \{G'\}_{i,j} &= -\sigma(-c_j w_i^T) \\ \{G'\}_{i,n1} &= \sigma(c_{n1} w_i^T) \\ &\cdots \\ \{G'\}_{i,nk} &= \sigma(c_{nk} w_i^T) \end{aligned}$$

with other elements in G' zeros;

Therefore in matrix form, U is updating to U^+ with gradient matrix G:

$$U^+ = U + \eta G U \tag{20}$$

Intuitively, G' is updating $\nabla_X f$ row by row.

We have to show G is doing coordinate ascend:

Lemma: Intuitively, G' is updating $\nabla_X f$ row by row, specifically,

$$\mathbb{E}(G') = \nabla_X f \tag{21}$$

Proof: In one epoch, as the observed pairs go through all corpus, (w_i, c_j) appears with distribution:

$$\{G'\}_{i,j} \sim \frac{\#(w_i, c_j)}{\sum_{i'} \sum_{j'} \#(w_{i'}, c_{j'})}$$

with negative samples from the distribution:

$$\{G'\}_{i,ns} \sim \frac{\sum_i \#(w_i, c_{ns})}{\sum_{i'} \sum_{j'} \#(w_{i'}, c_{j'})}$$

Thus, for an individual element $\{G'\}_{k,l}$ in G', we have

$$\mathbb{E}(\{G'\}_{k,l}) = \frac{\#(w_k, c_l)}{\sum_{i'} \sum_{j'} \#(w_{i'}, c_{j'})}$$

Conclusions

In discussion, we should claim that our purpose is to design a distributed version of the non-convex algorithm that can scale up and out.

References

Abka, A. F. 2016. Evaluating the use of word embeddings for part-of-speech tagging in bahasa indonesia. In *Computer, Control, Informatics and its Applications (IC3INA), 2016 International Conference on*, 209–214. IEEE.

Arora, S.; Li, Y.; Liang, Y.; Ma, T.; and Risteski, A. 2015. Rand-walk: A latent variable model approach to word embeddings. *arXiv preprint arXiv:1502.03520*.

Fonarev, A.; Grinchuk, O.; Gusev, G.; Serdyukov, P.; and Oseledets, I. 2017. Riemannian optimization for skip-gram negative sampling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 2028–2036.

Gutmann, M., and Hyvärinen, A. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 297–304.

Kusner, M.; Sun, Y.; Kolkin, N.; and Weinberger, K. 2015. From word embeddings to document distances. In *International Conference on Machine Learning*, 957–966.

Levy, O., and Goldberg, Y. 2014. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, 2177–2185.

Levy, O.; Goldberg, Y.; and Dagan, I. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics* 3:211–225.

Li, Y.; Xu, L.; Tian, F.; Jiang, L.; Zhong, X.; and Chen, E. 2015. Word embedding revisited: A new representation learning and explicit matrix factorization perspective. In *IJCAI*, 3650–3656.

Mahoney, M. 2011. Large text compression benchmark. *URL: http://www. mattmahoney. net/text/text. html*.

Manning, C.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S.; and McClosky, D. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 55–60.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In Burges, C. J. C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc. 3111–3119.

Mikolov, T.; Yih, W.-t.; and Zweig, G. 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*, 746–751.

Mimno, D., and Thompson, L. 2017. The strange geometry of skip-gram with negative sampling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2873–2878.

Park, D.; Kyrillidis, A.; Caramanis, C.; and Sanghavi, S. 2016. Finding low-rank solutions to matrix problems, efficiently and provably. *arXiv preprint arXiv:1606.03168*.

Sandhaus, E. 2008. The new york times annotated corpus, linguistic data consortium. *Philadelphia* 6(12):e26–752.

Tu, S.; Boczar, R.; Simchowitz, M.; Soltanolkotabi, M.; and Recht, B. 2015. Low-rank solutions of linear matrix equations via procrustes flow. *arXiv preprint arXiv:1507.03566*.

Zou, W. Y.; Socher, R.; Cer, D.; and Manning, C. D. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1393–1398.