Batch IS NOT Heavy: Learning Word Representations From All Samples

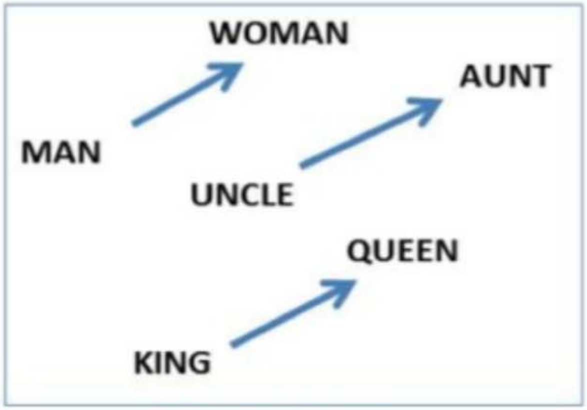
[1]**Xin Xin,** [1]Fajie Yuan,[2] Xiangnan He,[1] Joemon Jose

[1] School of Computing Science, University of Glasgow
[2] School of Computing, National University of Singapore

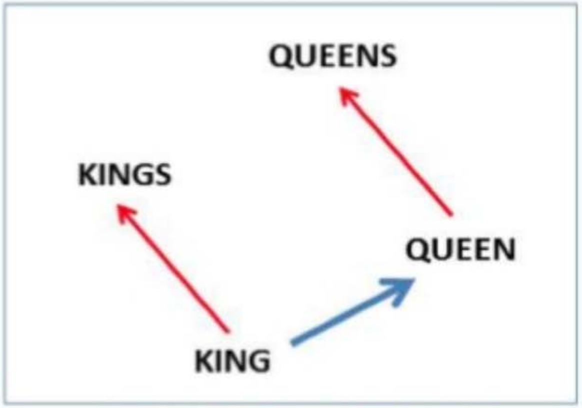
Presented by **Xin Xin**@ACL 2018, July 17, 2018

Word Representations

- Representing words has become a basis for many NLP tasks.
  - One-hot encoding
    - Large dimensionality
    - Sparse representation (most zeros)
  - Dense word embedding
    - 100~400 dims with real-valued vectors
    - Semantic and syntactic meaning in latent space



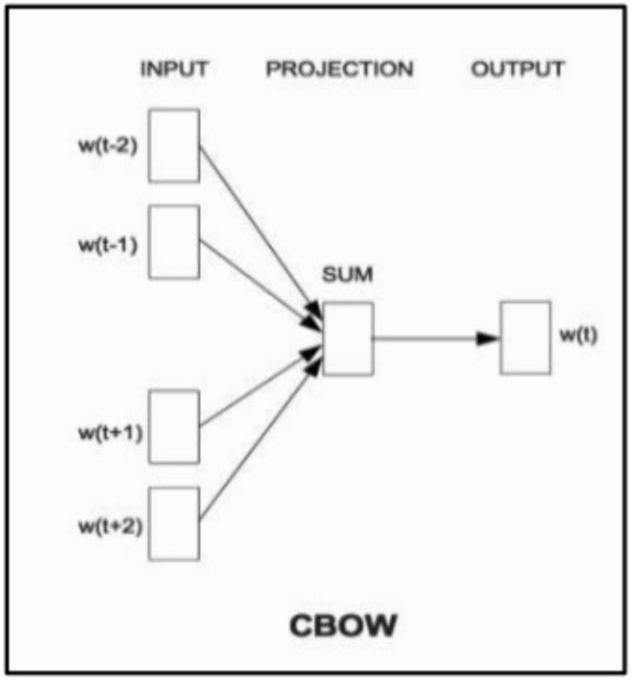
Gender Relation

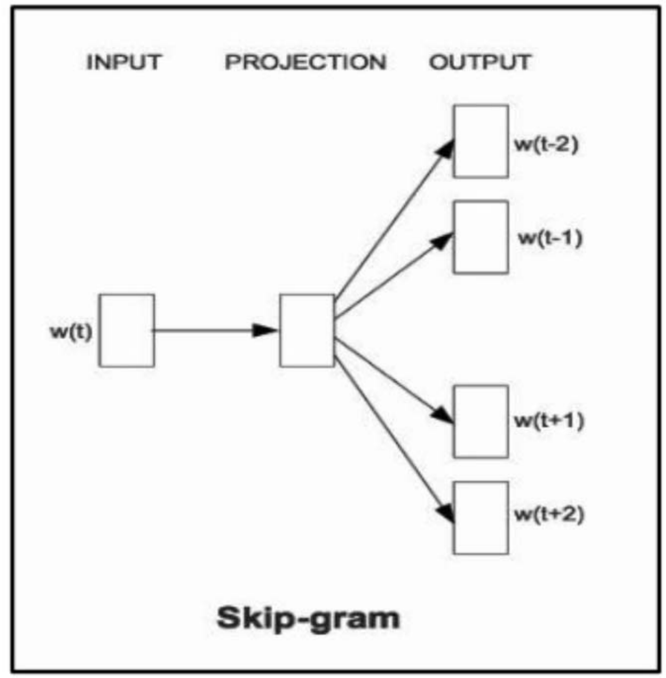


Singular/Plural Relation

Learning word embedding

- Predictive models:
  - Word2vec: CBOW & Skip-gram





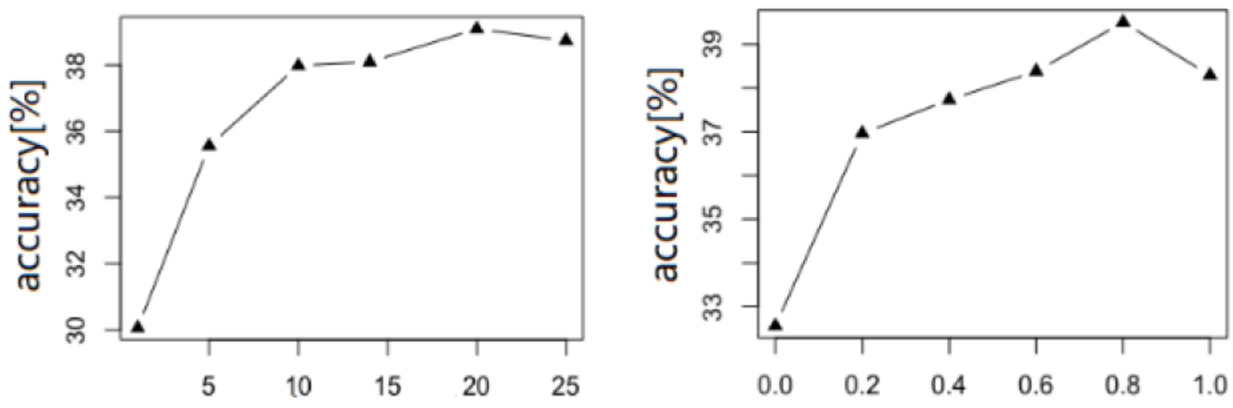
- Count-based models:
  - GloVe: Biased MF on word co-occurrence statistics

Learning word embedding

- Training of Skip-gram
  - Predicting proper context c given target word w
  - Negative sampling to introduce negative c
    - Word frequency-based sampling distribution
  - SGD to perform optimization
- Limitations
  - Sampling is a biased approach
    - Chen et al. (2018) recently found that replacing the original sampler with an adaptive sampler could result in better performance
  - SGD fluctuates heavily

Limitations

- Sampled negative instances have great influence



Sample size Sampling distribution (power)

Word analogy accuracy on text8 corpus

  - Sample size and sampling distribution have great impact
  - Smaller corpora tend to require a large sample size

Motivations

- Can we drop out negative sampling and directly learn from whole data?
- With whole data considered, can we design an efficient learning scheme to perform optimization?

Contributions

- We directly learn word embeddings from whole data without any sampling
  - All observed (positive) and unobserved (negative) (w, c) pairs are considered
  - Fine-grained weights for negative pairs
- We propose an efficient training algorithm to tackle the huge whole data
  - Keeps the same complexity with sampling based methods
  - More stable convergence

Loss function for all data

- Count-based loss function
- Account for all examples without any sampling

$$L = \underbrace{\sum_{\boxed{(w,c)\in S}} \boxed{\alpha_{wc}^{+}}(\boxed{r_{wc}^{+}} - \boxed{U_w \tilde{U}_c^T})^2}_{L_P} + \underbrace{\sum_{(w,c)\in(V\times V)\setminus S} \boxed{\alpha_{wc}^{-}}(\boxed{r^{-}} - \boxed{U_w \tilde{U}_c^T})^2}_{L_N}$$

  - S: set of positive (w, c) co-occurrence pairs
  - V: vocabulary
  - $U_w(\tilde{U}_c)$: embedding vectors for word (context)
  - $\alpha_{wc}^{+}(\alpha_{wc}^{-})$: weights for positive(negative) (w, c) pairs
  - $r_{wc}^{+}(r^{-})$: target values for positive(negative) (w, c) pairs

Difficulties to Optimize

- Time complexity

$$L = \underbrace{\sum_{(w,c)\in S} \alpha_{wc}^{+}(r_{wc}^{+} - U_w \tilde{U}_c^T)^2}_{L_P} + \underbrace{\sum_{(w,c)\in (V\times V)\setminus S} \alpha_{wc}^{-}(r^{-} - U_w \tilde{U}_c^T)^2}_{L_N}$$

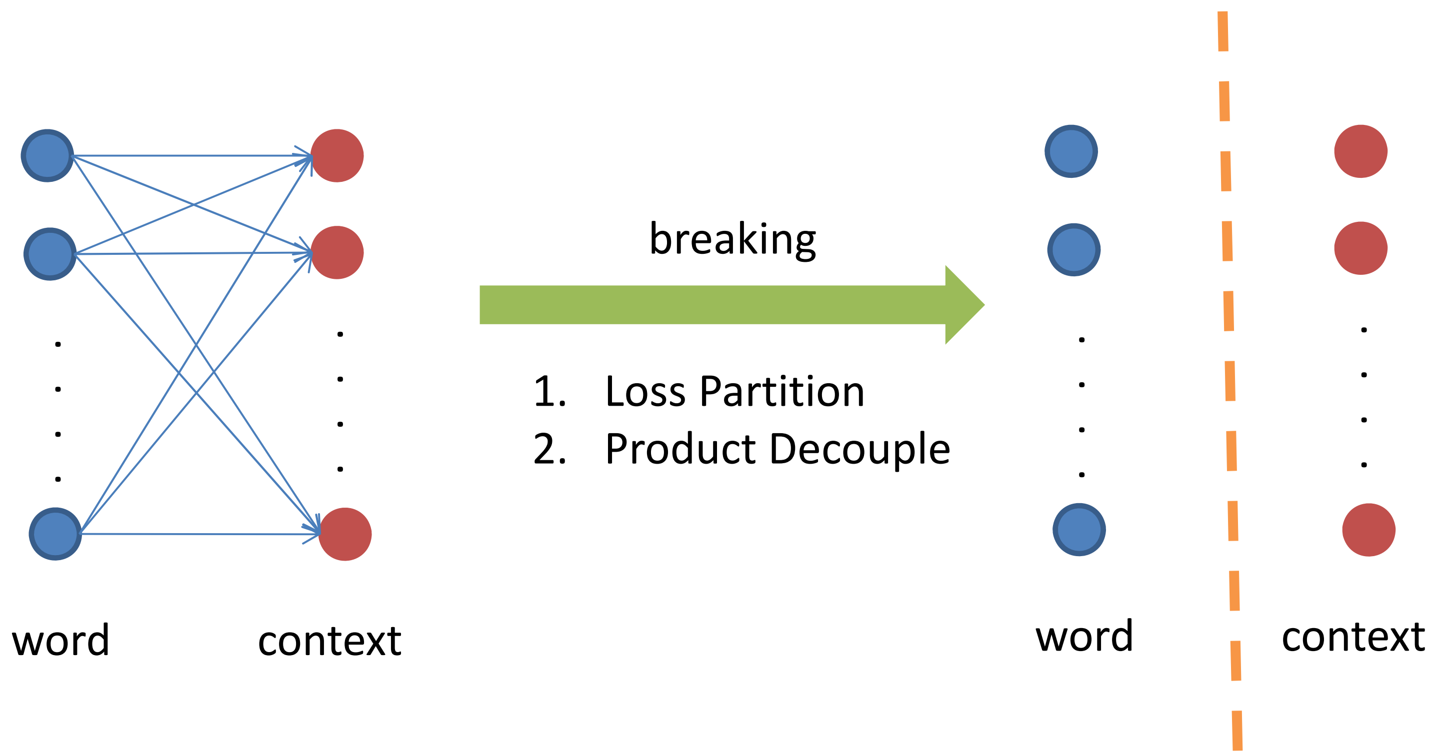
  - $O(|V|^2 k)$: easily reach tens of billions (e.g., with a 100K vocabulary, $|V|^2$ reaches 10 billion, k: embedding size)

A more efficient training algorithm needs to be developed

Difficulties to Optimize

$$L = \underbrace{\sum_{(w,c)\in S} \alpha^{+}_{wc}(r^{+}_{wc} - U_w \tilde{U}_c^T)^2}_{L_P} + \underbrace{\sum_{(w,c)\in (V\times V)\setminus S} \alpha^{-}_{wc}(r^{-} - U_w \tilde{U}_c^T)^2}_{L_N}$$

$|V| \times |V|$ interactions



Loss Partition

$$L = \underbrace{\sum_{(w,c)\in S} \alpha^{+}_{wc}(r^{+}_{wc} - U_w \tilde{U}_c^T)^2}_{L_P} + \underbrace{\sum_{(w,c)\in (V\times V)\setminus S} \alpha^{-}_{wc}(r^{-} - U_w \tilde{U}_c^T)^2}_{L_N}$$

- The major computation lies in L_N
  - Transfer L_N

$$L_N = \underbrace{\boxed{\sum_{w\in V}\sum_{c\in V} \alpha^{-}_{c}(r^{-} - U_w \tilde{U}_c^T)^2}}_{L_A} - \underbrace{\boxed{\sum_{(w,c)\in S} \alpha^{-}_{c}(r^{-} - U_w \tilde{U}_c^T)^2}}_{\text{Merge with } L_P}$$

  - Now, the major part falls in L_A

Product Decouple

- Inner product Decouple
  - Rewrite L_A into $\tilde{L}_A$ with the constant part $\alpha_c^-(r^-)^2$ omitted

$$\widetilde{L}_A = \sum_{\boxed{w \in V \; c \in V}} \alpha_c^- \sum_{d=0}^{k} \boxed{u_{wd}\tilde{u}_{cd}} \sum_{d'=0}^{k} \boxed{u_{wd'}\tilde{u}_{cd'}}$$

$$- 2r^- \sum_{\boxed{w \in V \; c \in V}} \alpha_c^- \sum_{d=0}^{k} \boxed{u_{wd}\tilde{u}_{cd}}$$

$|V| \times |V|$ interactions between u_w and $\tilde{u}_c$

Product Decouple

- Inner product Decouple
  - Rewrite L_A into $\widetilde{L}_A$ with the constant part $\alpha_c^-(r^-)^2$ omitted

$$\widetilde{L}_A = \sum_{w \in V} \sum_{c \in V} \alpha_c^- \sum_{d=0}^{k} u_{wd} \tilde{u}_{cd} \sum_{d'=0}^{k} u_{wd'} \tilde{u}_{cd'}$$

$$- 2r^- \sum_{w \in V} \sum_{c \in V} \alpha_c^- \sum_{d=0}^{k} u_{wd} \tilde{u}_{cd}$$

Commutative property

Product Decouple

- Inner product Decouple
  - Rewrite L_A into $\tilde{L}_A$ with the constant part $\alpha_c^-(r^-)^2$ omitted

$$\widetilde{L}_A = \sum_{w \in V} \sum_{c \in V} \alpha_c^- \sum_{d=0}^{k} u_{wd} \tilde{u}_{cd} \sum_{d'=0}^{k} u_{wd'} \tilde{u}_{cd'}$$

$$- 2r^- \sum_{w \in V} \sum_{c \in V} \alpha_c^- \sum_{d=0}^{k} u_{wd} \tilde{u}_{cd}$$

Commutative property

$$\widetilde{L}_A = \sum_{d=0}^{k} \sum_{d'=0}^{k} \sum_{w \in V} u_{wd} u_{wd'} \sum_{c \in V} \alpha_c^- \tilde{u}_{cd} \tilde{u}_{cd'}$$

$$- 2r^- \sum_{d=0}^{k} \sum_{w \in V} u_{wd} \sum_{c \in V} \alpha_c^- \tilde{u}_{cd}$$

Product Decouple

- Inner product Decouple
  - Rewrite L_A into $\widetilde{L}_A$ with the constant part $\alpha_c^-(r^-)^2$ omitted

$$\widetilde{L}_A = \sum_{w\in V}\sum_{c\in V}\alpha_c^-\sum_{d=0}^{k} u_{wd}\tilde{u}_{cd}\sum_{d'=0}^{k} u_{wd'}\tilde{u}_{cd'}$$

$$-2r^-\sum_{w\in V}\sum_{c\in V}\alpha_c^-\sum_{d=0}^{k} u_{wd}\tilde{u}_{cd}$$

Commutative property

$$\widetilde{L}_A = \sum_{d=0}^{k}\sum_{d'=0}^{k}\sum_{w\in V}\boxed{u_{wd}u_{wd'}}\sum_{c\in V}\boxed{\alpha_c^-\tilde{u}_{cd}\tilde{u}_{cd'}}$$

$$-2r^-\sum_{d=0}^{k}\sum_{w\in V}\boxed{u_{wd}}\sum_{c\in V}\boxed{\alpha_c^-\tilde{u}_{cd}}$$

u_w and $\tilde{u}_c$ are now independent

Product Decouple

- Fix one term and update the other

$$\widetilde{L}_A = \sum_{d=0}^{k} \sum_{d'=0}^{k} \sum_{w \in V} u_{wd} u_{wd'} \boxed{\sum_{c \in V} \alpha_c^- \tilde{u}_{cd} \tilde{u}_{cd'}}$$
$$- 2r^- \sum_{d=0}^{k} \sum_{w \in V} u_{wd} \boxed{\sum_{c \in V} \alpha_c^- \tilde{u}_{cd}}$$

- We can achieve a $|V| \times |V|$ to $|V| + |V|$ acceleration
  - Time complexity of L_A reduces from $O(|V|^2 k)$ to $O(k^2 |V|)$
  - Embedding size k is much smaller than vocabulary size $|V|$

Efficient training

- Total time complexity
  - The total time complexity is $O(|S|k + |V|k^2)$
  - $\frac{|S|k}{|V|k^2} = \frac{\bar{c}}{k} \gg 1$ ($\bar{c}$: the average number of positive contexts for a word)
  - The complexity is determined by the number of positive samples

We can train on whole data without any sampling but the time complexity is only determined by the positive part.

Experiments

- Evaluation tasks
  - Word analogy (semantic&syntactic)
    - King is to man as queen is to ?
  - Word Similarity
    - MEN,MC,RW,RG,WSim,WRel
  - QVEC (Tsvetkov et al., 2015)
    - Intrinsic evaluation based on feature alignment
- Training Corpora: Text8, NewsIR, Wiki
- Baseline: Skip-gram, Skip-gram with adaptive sampler, GloVe, LexVec (Salle et al., 2016).

Experiments

- Word analogy accuracy (%) on Text8

	Semantic	Syntactic	Total
Skip-gram	47.51	32.26	38.60
Skip-gram-a	48.10	33.78	39.74
GloVe	45.11	26.89	34.47
LexVec	51.87	31.78	40.14
Our model	56.66	32.42	**42.50**

- Our model performs especially good
- GloVe performs poorly (lack of negative information)
- Syntactic performance is not so good as semantic performance

Experiments

- Word similarity & QVEC tasks on Text8

	MEN	MC	RW	RG	WSim	WRel	QVEC
Skip-gram	0.6868	0.6776	0.3336	0.6904	0.7082	0.6539	0.3999
Skip-gram-a	0.6885	0.6667	0.3399	**0.7035**	0.7291	0.6708	0.4062
GloVe	0.4999	0.3349	0.2614	0.3367	0.5168	0.5115	0.3662
LexVec	0.6660	0.6267	0.2935	0.6076	0.7005	0.6862	**0.4211**
Our model	**0.6966**	**0.6975**	**0.3424**	0.6588	**0.7484**	**0.7002**	**0.4211**

  - Similar results with word analogy tasks
  - GloVe performs poorly on these two tasks

Experiments

- Word analogy accuracy (%) on NewsIR

	Semantic	Syntactic	Total
Skip-gram	70.81	47.48	58.10
Skip-gram-a	71.74	48.71	59.20
GloVe	78.79	41.58	58.52
LexVec	76.11	39.09	55.95
Our model	78.47	48.33	**61.57**

- GloVe's performance is improved
- The proposed model still over-performs GloVe
  - The importance of negative examples

Experiments

- Word similarity & QVEC tasks on NewsIR

	MEN	MC	RW	RG	WSim	WRel	QVEC
Skip-gram	0.7293	0.7328	0.3705	0.7184	0.7176	0.6147	0.4182
Skip-gram-a	**0.7409**	0.7513	0.3797	0.7508	0.7442	0.6398	0.4159
GloVe	0.5839	0.5637	0.2487	0.6284	0.6029	0.5329	0.3948
LexVec	0.7301	**0.8403**	0.3614	**0.8341**	0.7404	**0.6545**	0.4172
Our model	0.7407	0.7642	**0.4610**	0.7753	**0.7453**	0.6322	**0.4319**

  – GloVe still performs poorly on these two tasks

Experiments

- Word analogy accuracy (%) on Wiki

	Semantic	Syntactic	Total
Skip-gram	73.91	61.91	67.37
Skip-gram-a	75.11	61.94	67.92
GloVe	77.38	58.94	67.33
LexVec	76.31	56.83	65.48
Our model	77.64	60.96	**68.52**

- Models tend to have similar performance in large datasets

Experiments

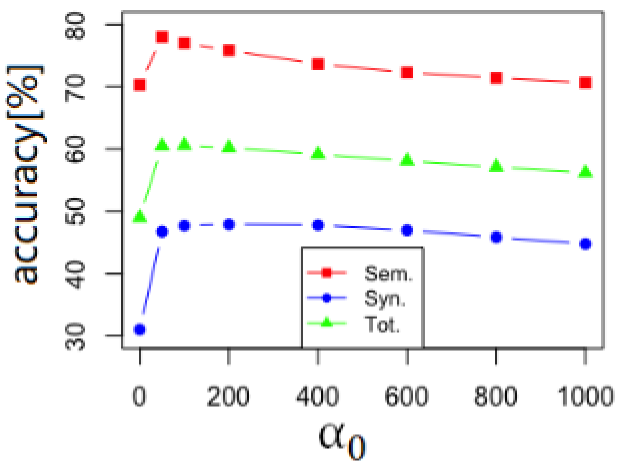
- Word similarity & QVEC tasks on Wiki

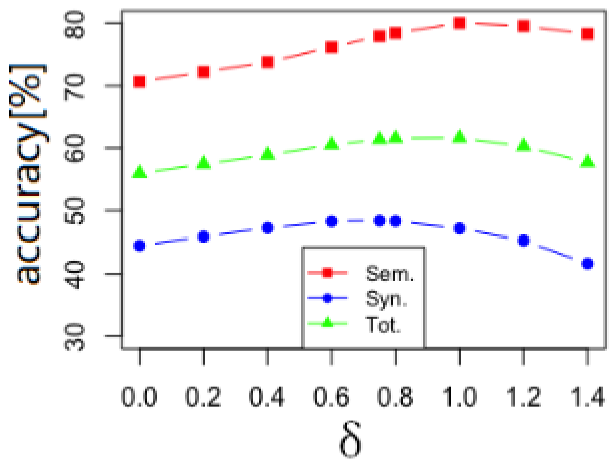
	MEN	MC	RW	RG	WSim	WRel	QVEC
Skip-gram	0.7564	0.8083	0.4311	0.7678	**0.7662**	0.6485	0.4306
Skip-gram-a	**0.7577**	0.7940	0.4379	0.7683	0.7110	0.6488	0.4464
GloVe	0.7370	0.7767	0.3197	0.7499	0.7359	0.6336	0.4206
LexVec	0.7256	**0.8219**	0.4383	0.7797	0.7548	0.6091	0.4396
Our model	0.7396	0.7840	**0.4966**	**0.7800**	0.7492	**0.6518**	**0.4489**

- To conclude
  - Our model performs especially good in smaller datasets
  - GloVe performs poorly on word similarity and QVEC tasks
  - The difference between models tends to become smaller in large datasets

Experiments

- Effect of weight parameters $\alpha_{wc}^{-} = \alpha_{c}^{-} = \alpha_0 \frac{M_{*c}^{\delta}}{\sum_{c \in V} M_{*c}^{\delta}}$



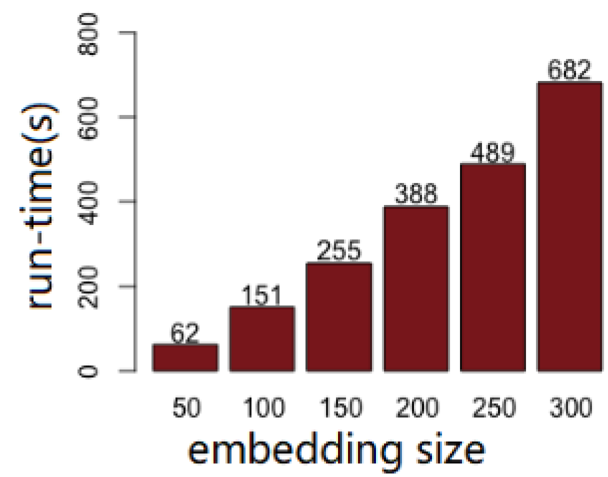


  - Performance boosts when α_0 becomes non-zero
    - Negative information is of vital importance
  - Best performance achieves when δ is around 0.75
    - Same with the power used in negative sampling

Experiments

- Running time on NewsIR corpus

	Single iter	Iteration	Total
SG-3	259s	15	65m
SG-7	521s	15	131m
SG-10	715s	15	179m
Ours	388s	50	322m



- In a single iteration, our model has the same level of running time with skip-gram
- The proposed model need to run more iteration, resulting in a little longer total time
- Running time has almost a linear relationship with embedding size
  - $O(|S|k)$ (positive pairs) accounts for the main part in total $O(|S|k + |V|k^2)$ complexity

Conclusion&Future works

- Conclusion:
  - We proposed a new embedding method which can directly learn from whole data without any sampling
  - We developed a new learning scheme to perform efficient optimization
    - Complexity of learning from whole data is only determined by the positive part.
- Future works:
  - Generalize the proposed learning scheme to other loss functions
  - Full example learning for deep models

Thank you