CS546: GloVe

Global Vectors for Word Representation

Aming Ni

Overview

- One Hot Encoding
- Global Matrix Factorization
- Local Context Window
- Short Intro to Skip-Gram
- GloVe
- GloVe V.S. Skip-Gram
- Results on GloVe

One Hot Encoding

dog	5	0 0 0 0 0 1 0 0 … 0 0 0 0 0 ….
UIUC	3000	0 0 0 0 0 0 0 0 … 0 0 0 1 0 ….

- Sparsity: High OOV rate, huge # of parameters.
- Language models such as n-gram?
- We want:
  - Reduce # of parameters
  - Utilize both local and global information
  - Generalization
- Distribution hypothesis: words appear in similar contexts should be similar.

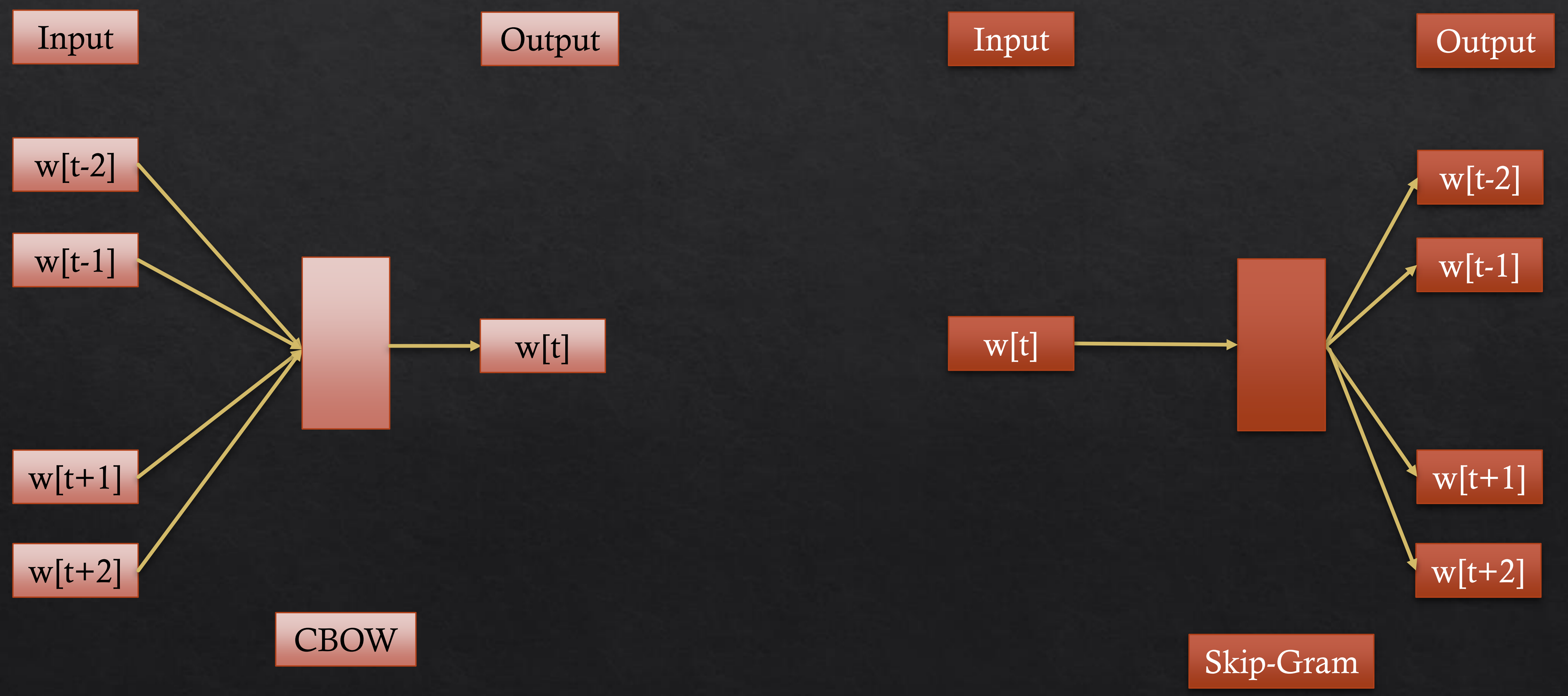
Global Matrix Factorization

- Utilize low-rank approximations to decompose large matrices that capture statistical information about a corpus.
- Latent Semantic Analysis (LSA)
  - The matrices are of “term-document” type
    - The rows correspond to words, and the columns correspond to different documents
  - Use a rank-k SVD to preserve the similarity structure among columns.
- PMI Matrix: perform a rank-k SVD on the matrix.

Local Context Window

- Learn the word representations in full context, rather than just the preceding context as is the case with language models.
- Continuous Bag of Words(CBOW)
  - Objective is to predict a word given its context
- Word2vec/Skip-Gram
  - Objective is to predict a context given a word.

CBOW and Skip-Gram Models



Short Intro Skip-Gram

- Maximize the average log probability: $\frac{1}{N}\sum_{t=1}^{N}\sum_{-c\leq j\leq c, j\neq 0} log p(w_{t+j}|w_t)$
- One possible model softmax: $p(w_j|w_i) = \frac{\exp(w_i^T w_j)}{\sum_{k=1}^{V}\exp(w_i^T w_k)}$
- One problem: vocabulary size can be huge, each w_j in $p(w_j|w_i)$ takes O(|V|) to compute
  - Solution: Hierarchical Softmax, Negative Sampling.

Why GloVe?

- Try to use both global statistics and local window context.
- Train only on the ***nonzero elements*** in a word-word co-occurrence matrix
- Propose a specific ***weighted least squares model*** that trains on global word-word co-occurrence counts

Some notations

- X: the matrix of word-word co-occurrence
- X_{ij}: number of times word j occurs in the context of word i
- $X_i = \sum_k X_{ik}$: the number of times any word appears in the context of the word i
- $P_{ij} = \mathrm{P}(j|i) = X_{ij} / X_i$: the probability that word j appears in the context of word i
- w_i: the representation of word i (if in vector form, $w_i \in R^d$)

Simple Example for Co-occurrence Probabilities

- Co-occurrence probabilities for target words ***ice*** and ***stream*** with selected context words
- Noise words like ***water*** and ***fashion*** cancel out (close to zero)
- Intuitively, the score for solid/gas given context ice/stream should be high.
- This suggests that we should look at the **ratios(relatively normalized)** of co-occurrence probabilities rather than the pure co-occurrence probabilities: $F(w_i, w_j, w_k) = \frac{P_{ik}}{P_{jk}}$

keypoint

Probability and Ratio	k = solid	k = gas	k = water	k = fashion
P(k|ice)	1.9 x 10^{-4}	6.6 x 10^{-5}	3.0 x 10^{-3}	1.7 x 10^{-5}
P(k|stream)	2.2 x 10^{-5}	7.8 x10^{-4}	2.2 x 10^{-3}	1.8 x 10^{-5}
P(k|ice) / P(k|stream)	**8.9**	**8.5 x 10^{-2}**	1.36	0.96

Adding Assumptions && Derivations

- We want the ***ratio***, now start with this equation: $F(w_i, w_j, w_k) = \frac{P_{ik}}{P_{jk}}$
- Let's enforce linear structures in vector space, we could use the difference. This assumption restricts us to only those functions of: $F(w_i - w_j, w_k) = \frac{P_{ik}}{P_{jk}}$
- Since $\frac{P_{ik}}{P_{jk}}$ is a scalar, we could further restrict F to be: $F((w_i - w_j)^T w_k) = \frac{P_{ik}}{P_{jk}}$
- We can restrict ***F*** to be a homomorphism function ***exp***(structure preserving mapping)

Adding Assumptions && Derivations Cont…

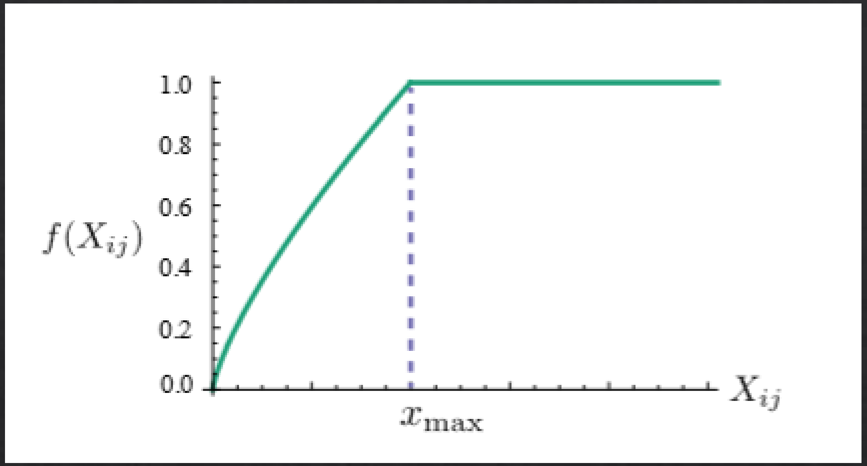
- F is an **exponent**, then: $F\left((w_i - w_j)^T w_k\right) = \frac{P_{ik}}{P_{jk}} = \frac{F(w_i^T w_k)}{F(w_j^T w_k)}$
- This means: $F(w_i^T w_k) = \exp(w_i^T w_k) = P_{ik} = \frac{X_{ik}}{X_i}$. If we solve for $w_i^T w_k = \log(P_{ik}) = \log(X_{ik}) - \log(X_i)$
- (1) Since $\log(X_i)$ is independent of k, we can set it as a bias term b_i. (2) We can also add another bias term b_k for w_k: $w_i^T w_k + b_i + b_k = \log(X_{ik})$
- Log is ill defind when $X_{ik} = 0$ (a simple fix is to change $\log(X_{ik})$ to $\log(X_{ik} + 1)$)
- One problem with the above objective function: it weights co-occurrence equally.

weighted least squares

- From previous, we have $w_i^T w_k + b_i + b_k = \log(X_{ik})$
- The author proposes the following objective function:

keypoint

  - $J = \sum_{i,j=1}^{V} f(X_{ij}) \left(w_i^T w_k + b_i + b_k - \log(X_{ik}) \right)^2$
  - We want f(0) = 0: 0 weight for zero elements in the matrix.
  - f(x) to be non-decreasing: more weight for high co-occurrence
  - f(x) to be relatively small for large values of x: frequent co-occurrence are not over-weighted.



- A lot functions can satisfy above properties for f(x), in the paper they used:
- f(x) = $\begin{cases} (x/x_{max})^{\alpha} \; if \; x < x_{max} \\ 1 \; otherwise \end{cases}$

keypoint

- For their experiments, they use $x_{max} = 100$, and $\alpha = 3/4$

Differs in weighting function and loss function

Relationship to Skip-Gram

GloVe Objective: J $=\sum_{i,j=1}^{V} f(X_{ij})\ (\ w_i^T w_k + b_i + b_k - \log(X_{ik})\ )^2$

- Let Q_{ij} in Skip-Gram be a softmax function: $Q_{ij} = \frac{\exp(w_i^T w_j)}{\sum_{k=1}^{V} \exp(w_i^T w_k)}$
- The objective function is to maximize the log probability: J $= -\sum_{\substack{i \in corpus \\ j \in context(i)}} logQ_{ij}$
- We can group terms that have the same values: : J $= -\sum_{i=1}^{V}\sum_{j=1}^{V} X_{ij} logQ_{ij}$
  - Again: X_{ij} is an element in the co-occurrence matrix X, $X_i = \sum_k X_{ik}$, P_{ij} = P(j|i) = X_{ij} / X_i
- Rewrite as: J $= -\sum_{i=1}^{V} X_i \sum_{j=1}^{V} P_{ij} logQ_{ij} = \sum_{i=1}^{V} X_i H(P_i, Q_i)$, where $H(P_i, Q_i)$ is the ***cross entropy*** of the distributions P_i *and* Q_i
- Rewrite again with ***least square*** measure: J $= \sum_{i=1}^{V} X_i H(P_i, Q_i) \approx \sum_{i,j} X_i (P_{ij} - Q_{ij})^2 \approx \sum_{i,j} X_i (logP_{ij} - logQ_{ij})^2 = \sum_{i,j} X_{ij} (w_i^T w_j - logX_{ij})^2$
- Replace X_i with a ***weight function*** $f(X_{ij})$: J $= \sum_{i,j} f(X_{ij})(w_i^T w_j - logX_{ij})^2$

Complexity of the Model

GloVe Objective: J $=\sum_{i,j=1}^{V} f(X_{ij})\ (w_i^T w_k + b_i + b_k - \log(X_{ik}))^2$

- GloVe Computational Complexity: nnz(X), or No worse than O($|V|^2$).
  - V could be huge!
- Assume the number of co-occurrence of X_{ij} can be modeled as a power-law function of the frequency work pair rank r_{ij}: $X_{ij} = \frac{k}{(r_{ij})^{\alpha}}$
- For the corpora they used in the paper, the frequencies can be modeled with $\alpha = 1.25$. This is roughly O($|C|^{0.8}$).
  - Window Based model: scales with the corpus size O(|C|).

Results on Word Analogy

- Word analogy task: ***a*** is to ***b*** as ***c*** is to ?   ->> ***man*** is to ***king*** as ***woman*** is to ?
- Model the problem as: which word d w_d is closest to $w_b - w_a + w_c$ by similarity metric(cosine)
- Underlined scores are best within groups of similarly-sized models.
- Bold scores are best overall.
- Size Scalability: can be trained on 42 billion token corpus.
- Performance Scalability: increasing corpus size improves GloVe
  - Not necessary true for other corpus. Example: SVD-L decreases.

Model	Dim.	Size	Sem.	Syn.	Tot.
ivLBL	100	1.5B	55.9	50.1	53.2
HPCA	100	1.6B	4.2	16.4	10.8
GloVe	100	1.6B	67.5	54.3	60.3
SG	300	1B	61	61	61
CBOW	300	1.6B	16.1	52.6	36.1
vLBL	300	1.5B	54.2	64.8	60.0
ivLBL	300	1.5B	65.2	63.0	64.0
GloVe	300	1.6B	80.8	61.5	70.3
SVD	300	6B	6.3	8.1	7.3
SVD-S	300	6B	36.7	46.6	42.1
SVD-L	300	6B	56.6	63.0	60.1
CBOW†	300	6B	63.6	67.4	65.7
SG†	300	6B	73.0	66.0	69.1
GloVe	300	6B	77.4	67.0	71.7
CBOW	1000	6B	57.3	68.9	63.7
SG	1000	6B	66.1	65.1	65.6
SVD-L	300	42B	38.4	58.2	49.2
GloVe	300	42B	**81.9**	**69.3**	**75.0**

Results on Word Similarity

- All vectors are 300 dimension
- Compute Cosine Similarity, and Use Spearman's rank correlation coefficient between this score and human judgment.
- GloVe outperforms CBOW* while using 42B tokens.

Model	Size	WS353	MC	RG	SCWS	RW
SVD	6B	35.3	35.1	42.5	38.3	25.6
SVD-S	6B	56.5	71.5	71.0	53.6	34.7
SVD-L	6B	65.7	<u>72.7</u>	75.1	56.5	37.0
CBOW†	6B	57.2	65.6	68.2	57.0	32.5
SG†	6B	62.8	65.2	69.7	<u>58.1</u>	37.2
GloVe	6B	<u>65.8</u>	<u>72.7</u>	<u>77.8</u>	53.9	<u>38.1</u>
SVD-L	42B	74.0	76.4	74.1	58.3	39.9
GloVe	42B	<u>**75.9**</u>	<u>**83.6**</u>	<u>**82.9**</u>	<u>**59.6**</u>	<u>**47.8**</u>
CBOW*	100B	68.4	79.6	75.4	59.4	45.5

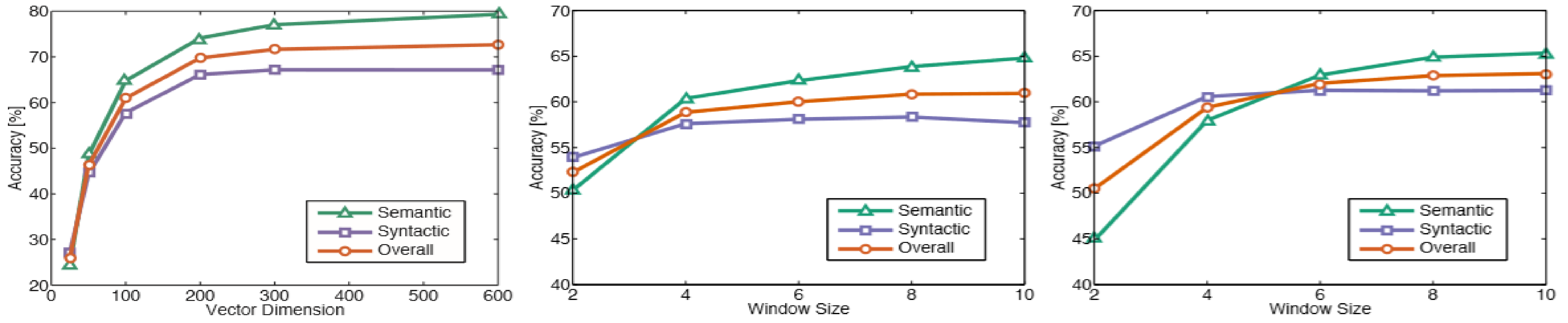
Results on NER Task

- Used as features to CRF-based model.
- GloVe model outperforms all other methods except for the CoNLL test set.

Model	Dev	Test	ACE	MUC7
Discrete	91.0	85.4	77.4	73.4
SVD	90.8	85.7	77.3	73.7
SVD-S	91.0	85.5	77.6	74.3
SVD-L	90.5	84.8	73.6	71.5
HPCA	92.6	**88.7**	81.7	80.7
HSMN	90.5	85.7	78.7	74.7
CW	92.2	87.4	81.7	80.2
CBOW	93.1	88.2	82.2	81.1
GloVe	**93.2**	88.3	**82.9**	**82.2**

Results on Vector Dim and Context Size

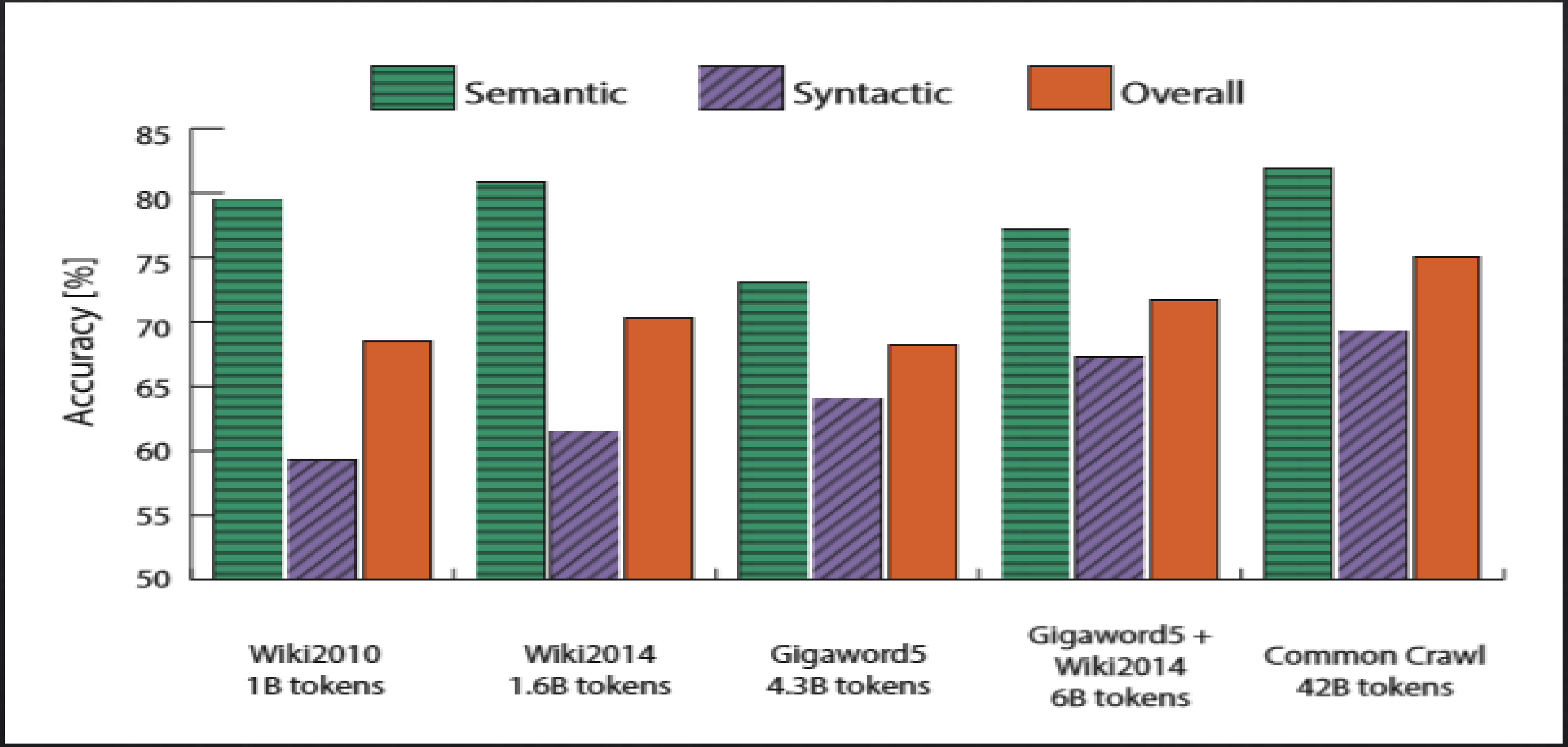
- Trained on 6 billion token corpus.
- (a) the window size is 10. (b) and (c) the vector size is 100
- Symmetric: context window left + right. Asymmetric: only left.
- Small window size: syntactic is better. Long window size: semantic is better.



(a) Symmetric context (b) Symmetric context (c) Asymmetric context

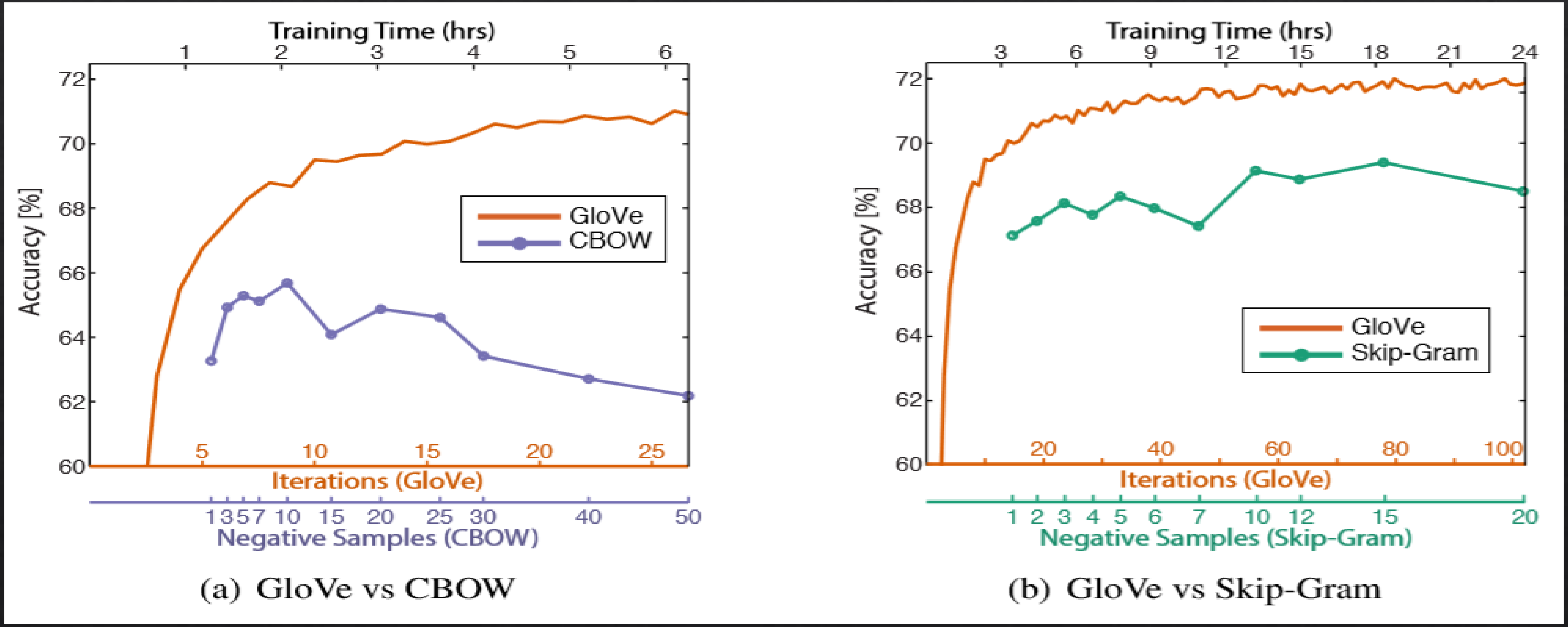
Results on Corpus Size

- Vector dimension 300.
- Syntactic subtask: monotonic increases in performance as the corpus size increases
  - Large corpus produces better statistics.



Results on Runtime(Iterations)

- Vector dimension is 300, 6B token corpus, vocabulary size 400,000, and window size 10
- Learning cuves:



(a) GloVe vs CBOW (b) GloVe vs Skip-Gram

Conclusion

- The paper shows that GloVe outperforms other methods on different experiments.
- However, as the math shown previously: all these models share some commonalities and only differ in weight functions, loss functions, and training time.
- There is many parameters that can have an impact on word2vec.
  - As the author points out that it’s possible that parameters in word2vec is not tuned to be optimal since there is so many parameters while GloVe is almost optimal in parameters.
  - For example, word2vec code they used is only designed for a single epoch for its study while GloVe is trained over many iterations for the LS problem.

Reference

- Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In NIPS, pages 3111–3119.