Analysis for Hierarchical Softmax in Word2Vec

Hello!

Chung-Min Tsai

B04505026

Xi-Zhen Liu

B04505009

You can find me at
b04505026@g.ntu.edu.tw

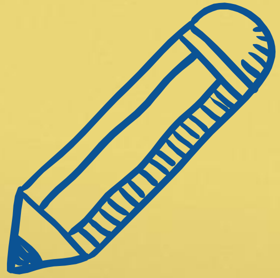
Outline

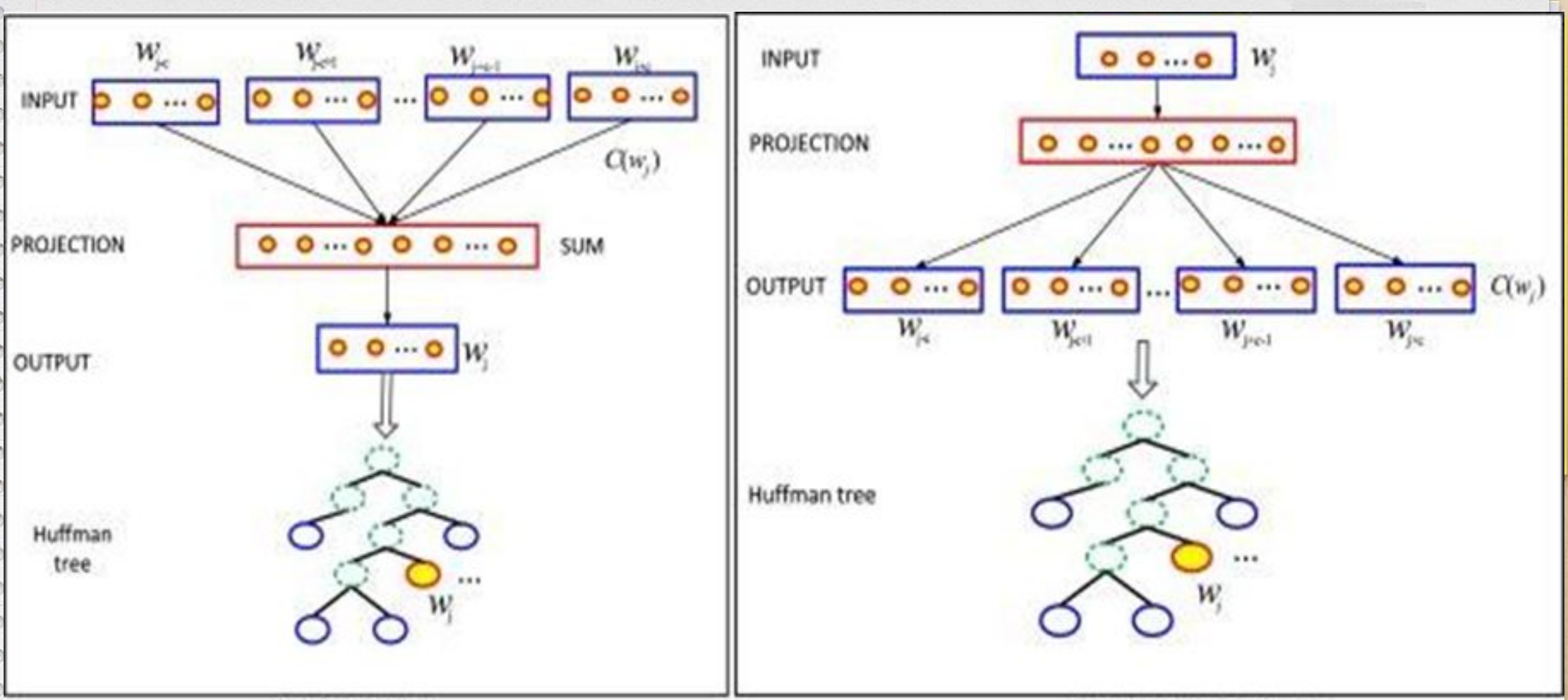
1. Problem Description
2. Experiment
3. Result
4. Conclusion
5. Reference

1.
Problem Description

Let’s start with the first set of slides

"The famous Word2Vec tool **Gensim** we are using now is based on google's algorithm published in 2013[3], and it produces word vector using Hierarchical Softmax and Negative Sampling."





(a) CBOW (b) SIKP-GRAM

Hierarchical Softmax

- It is for classification task which has giant number of classes (ex. word2vec)
- It is fast
- It is really fast
- Softmax vs Hierarchical Softmax: N vs log(N)
- It is fast, but how about its accuracy?
- From information point of view(- k log P), the word has lower frequency should have more information but it is at the bottom of the tree.

2.
Experiment

We will introduce
4 experiments

Data Set & Setting

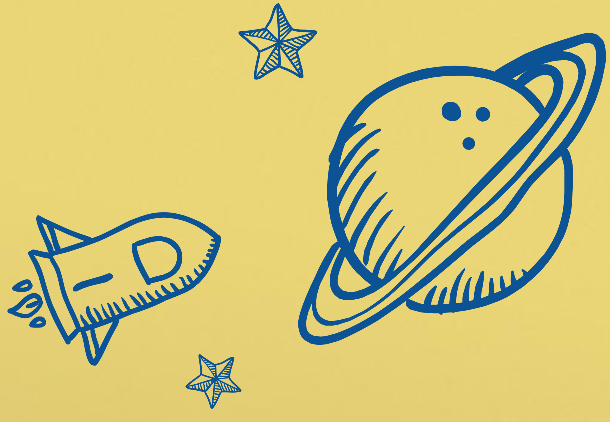
First Billion characters from wikipedia (1GB)

Vocab size: 218317

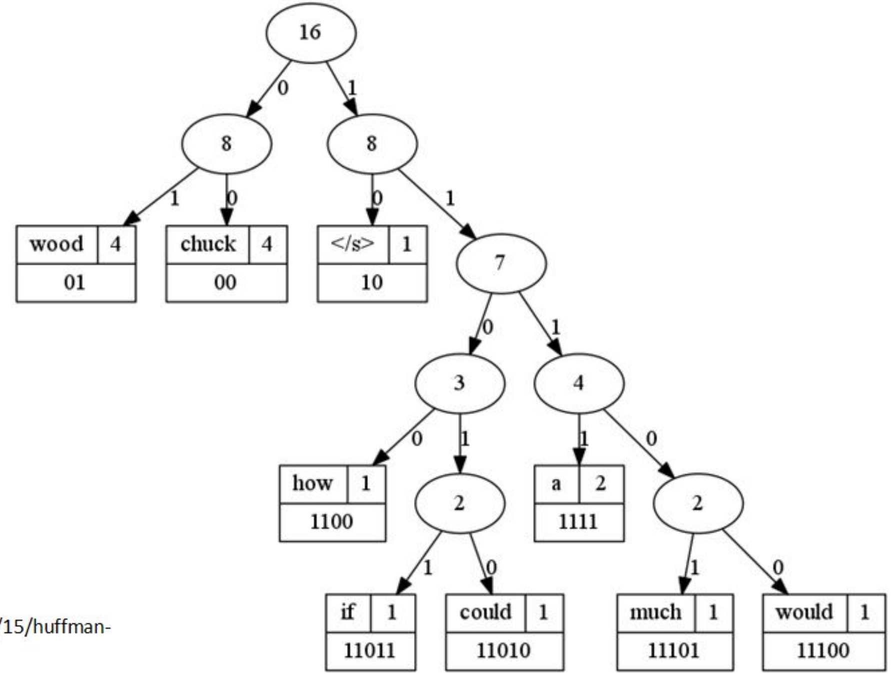
Word in file(after process): 123353508

Dimension: 300

Iteration: 15



Hierarchical Softmax



Source:
http://www.trevorsimonton.com/blog/2016/12/15/huffman-tree-in-word2vec.html

Negative Sampling

- EX.

  This is a interesting book

- Randomly pick K negative sample
- Minimize the probability of predicting those K negative sample words.
- Advantages:

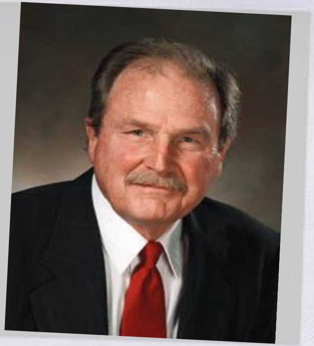
  When randomly picking K negative sample, those higher frequent words might have higher probability being picked!

Hierarchical Softmax

with Learning Rate Tuning

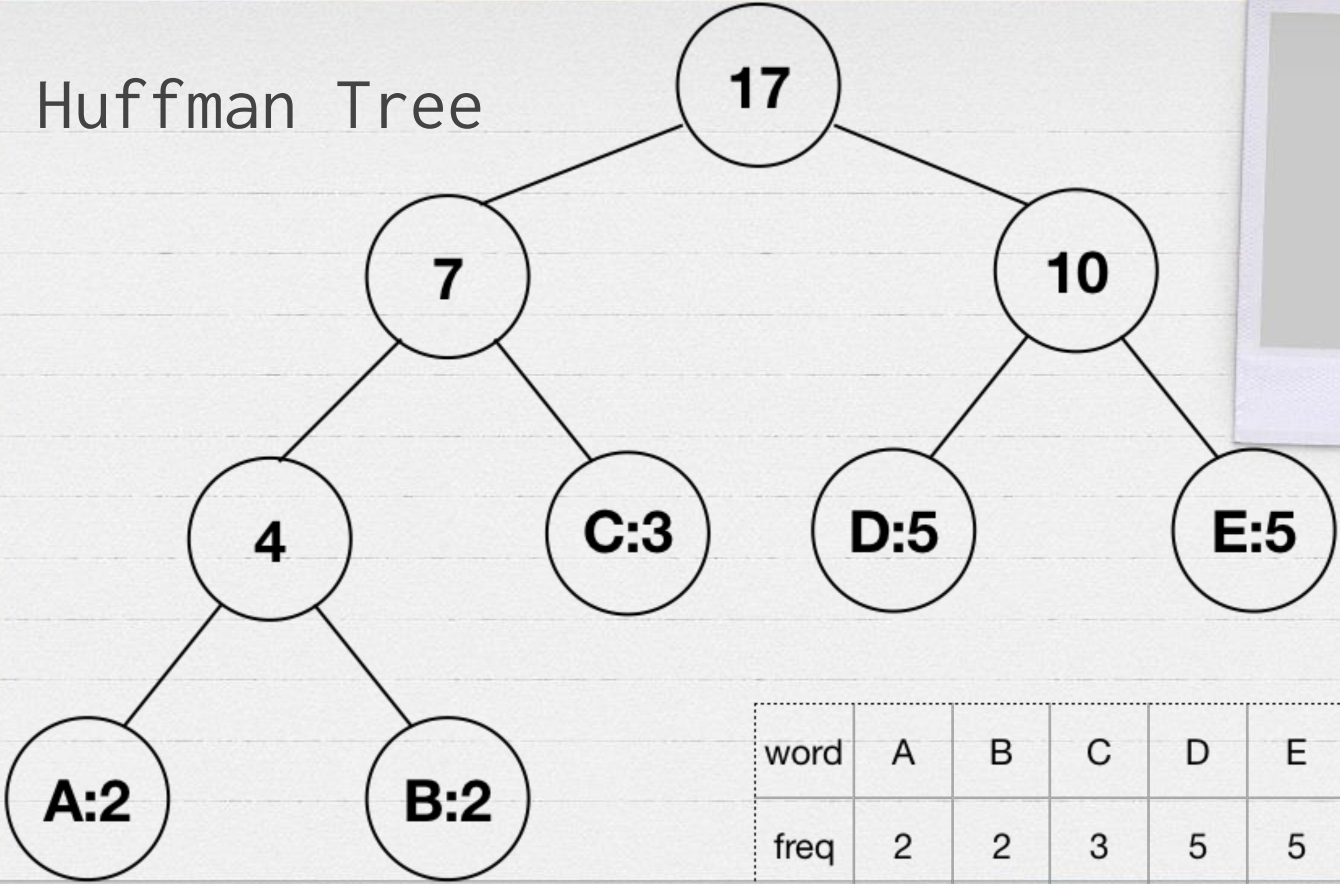
- When facing imbalance multiclass classification problem, we could tune the learning rate depend on the frequency of training sample.
- *Learning rate = Learning rate * (1 + codelen * 0.01)*

Reverse Hierarchical Softmax



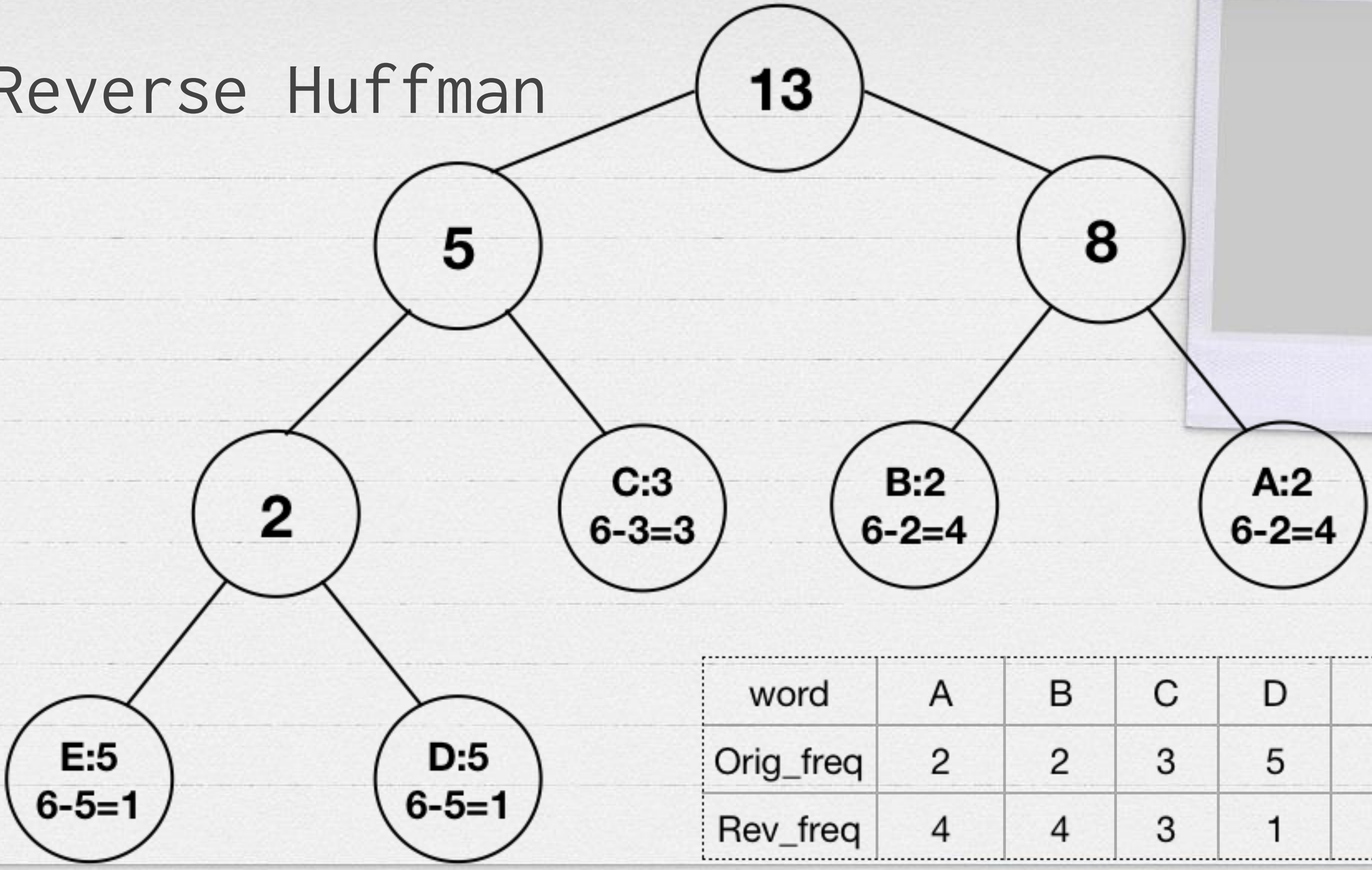
- For every word, minus its frequency by (Maxfreq + 1) to create a new list of Rev-freq.
- *Rev_freq = Max_freq + 1 - Orig_freq*
- Higher the Orig_freq is, lower the Rev_freq is.
- Reverse the order of the original frequency list, then place words to tree node by Huffman's rule.
- The most frequent words will be place at the bottom of the tree, while the less frequent words will be higher.

Huffman Tree



word	A	B	C	D	E
freq	2	2	3	5	5

Reverse Huffman



word	A	B	C	D	E
Orig_freq	2	2	3	5	5
Rev_freq	4	4	3	1	1

3.
Result

Which one is the best?

4 Why does this produce good word representations?

Good question. We don't really know.

The distributional hypothesis states that words in similar contexts have similar meanings. The objective above clearly tries to increase the quantity $v_w \cdot v_c$ for good word-context pairs, and decrease it for bad ones. Intuitively, this means that words that share many contexts will be similar to each other (note also that contexts sharing many words will also be similar to each other). This is, however, very hand-wavy.

Can we make this intuition more precise? We'd really like to see something more formal.

How to Evaluate?

1. Accuracy

Put training data into cbow model again and compute the accuracy.

What happen?

It's suck.

Hierarchical Softmax can't predict well.

2. Question Test

Using the test Tomas Mikolov (Google) provide in 2013.

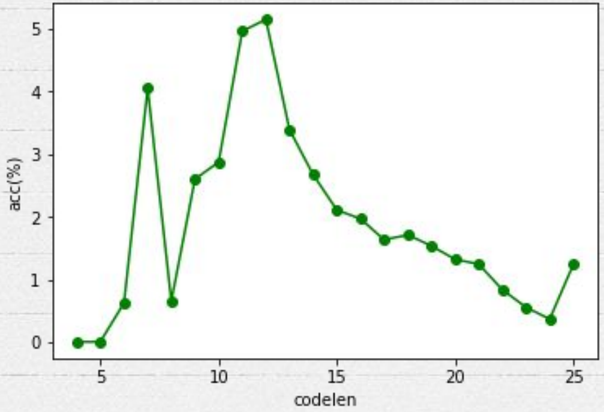
EX.

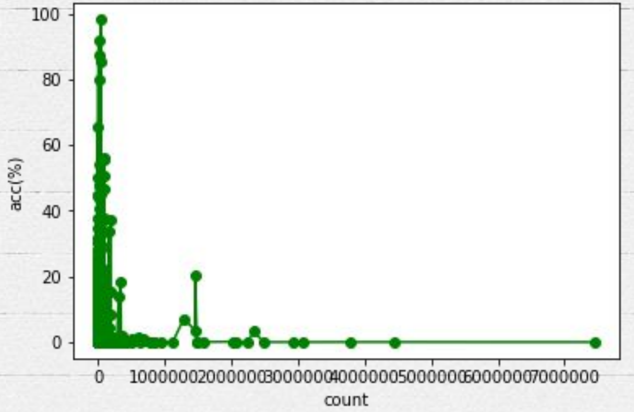
Man:Woman = brother:?

Hierarchical Softmax

Total Accuracy = 0.0211

Less than 30K Accuracy = 0.0201

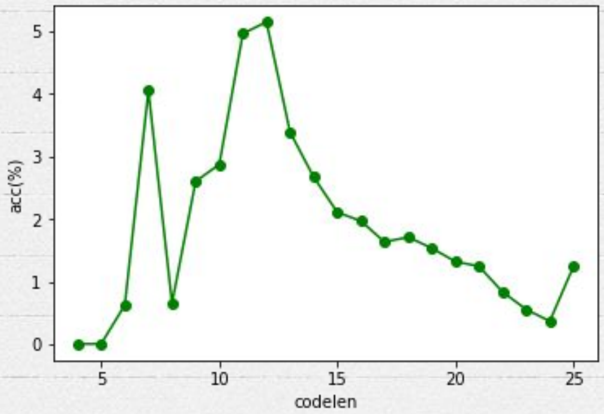


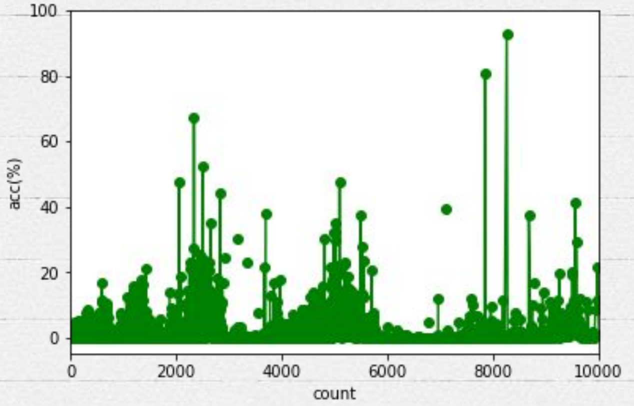


Hierarchical Softmax

Total Accuracy = 0.0211

Less than 30K Accuracy = 0.0201



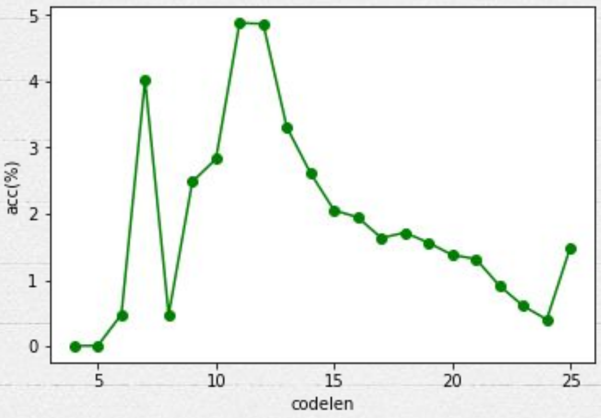


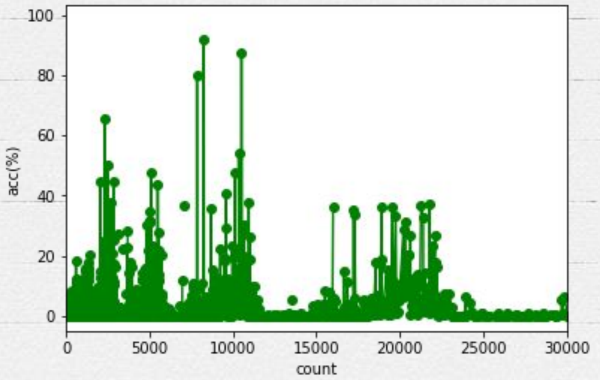
Hierarchical Softmax

with learning rate tuning

Total Accuracy = 0.0207

Less than 30K Accuracy = 0.0203

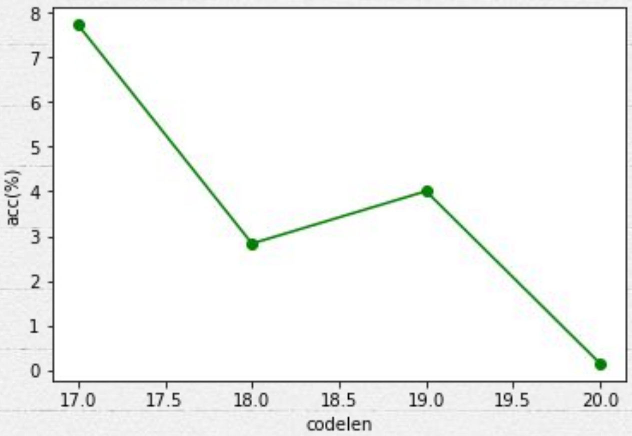


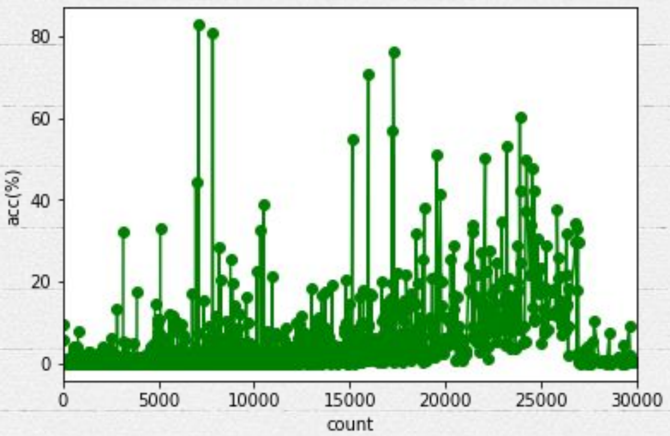


Reversed Hierarchical Softmax

Total Accuracy = 0.0262

Less than 30K Accuracy = 0.0241





Results

	Time	Semantic Test	Syntactic Test	Acc
hs	132(min)	62.94%	53.05%	2.11%
hs lr	137(min)	63.27%	53.48%	2.07%
Reverse hs	158(min)	54.47%	45.55%	2.62%
neg15	187(min)	76.06%	67.06%	-

hs: hierarchical softmax
hs lr:
hierarchical softmax with learning rate tuning
reversed hs:
reversed hierarchical softmax
neg15:
negative sampling with rate15

Dataset	HS	Rev-HS	HS-LR	Neg	Best
EN-WS-353-ALL	72.800929	73.400269	72.231402	68.528415	Rev-HS
EN-WS-353-REL	65.460239	67.571580	65.530073	60.809344	Rev-HS
EN-WS-353-SIM	76.47234	75.47357	75.13711	74.40610	HS
EN-RG-65	76.132924	71.698485	78.311897	75.623696	HS-LR
EN-MTurk-771	65.782173	65.108425	65.656249	65.517286	HS
EN-MC-30	82.955053	87.027149	80.596353	75.033380	Rev-HS
EN-YP-130	29.810025	29.552059	27.277414	39.399098	Neg
EN-MTurk-287	57.211054	57.303214	57.351206	67.066612	Neg
EN-RW-STANFORD	33.803207	34.013382	33.977009	38.629509	Neg
EN-MEN-TR-3k	71.191519	72.149495	71.523698	72.639810	Neg
Average	63.161946	63.329762	62.759241	63.765325	Neg

Red > Orange > Green > Blue

4.

Conclusion

So what have we got?

Conclusion

- There is no common criterion to evaluate word vector. It highly depends on the task.
- The advantages of Hierarchical Softmax is speed, but it couldn't improve accuracy.
- Negative sampling is currently the best.

5.
Reference

We learn a lot from...

1. Goldberg, Y. and Levy, O. (2014). word2vec explained: deriving mikolov et al.'s negative sampling word-embedding method.

2. Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space.

3. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In Advances in Neural Information Processing Systems, pages 3111–3119.