# Distributed Representation of Documents with Explicit Explanatory Features: Background Research

September 21th, 2015

SNU Data Mining Center

Han Kyul Kim

# Word2Vec Clustering Approach

## From word2vec to doc2vec: an approach driven by Chinese restaurant process

Posted on March 17, 2014 by Yingjie Miao.

Google's word2vec project has created lots of interests in the text mining community. It's a neural network language model that is "both supervised and unsupervised". Unsupervised in the sense that you only have to provide a big corpus, say English wiki. Supervised in the sense that the model cleverly generates supervised learning tasks from the corpus. How? Two approaches, known as Continuous Bag of Words (CBOW) and Skip-Gram (See Figure 1 in this paper). CBOW forces the neural net to predict current word by surrounding words, and Skip-Gram forces the neural net to predict surrounding words of the current w
a few optimization and approximati

Word vectors generated by the neu
"iOS" is close to "Android". Syntactic
checkout more examples here.

Although this provides high quality
high quality document vector. In thi
process called Chinese Restaurant
process and summing word vectors

## From Words to Paragraphs, Attempt 2: Clustering

Word2Vec creates clusters of semantically related words, so another possible approach is to exploit the similarity of words within a cluster. Grouping vectors in this way is known as "vector quantization." To accomplish this, we first need to find the centers of the word clusters, which we can do by using a clustering algorithm such as K-Means.

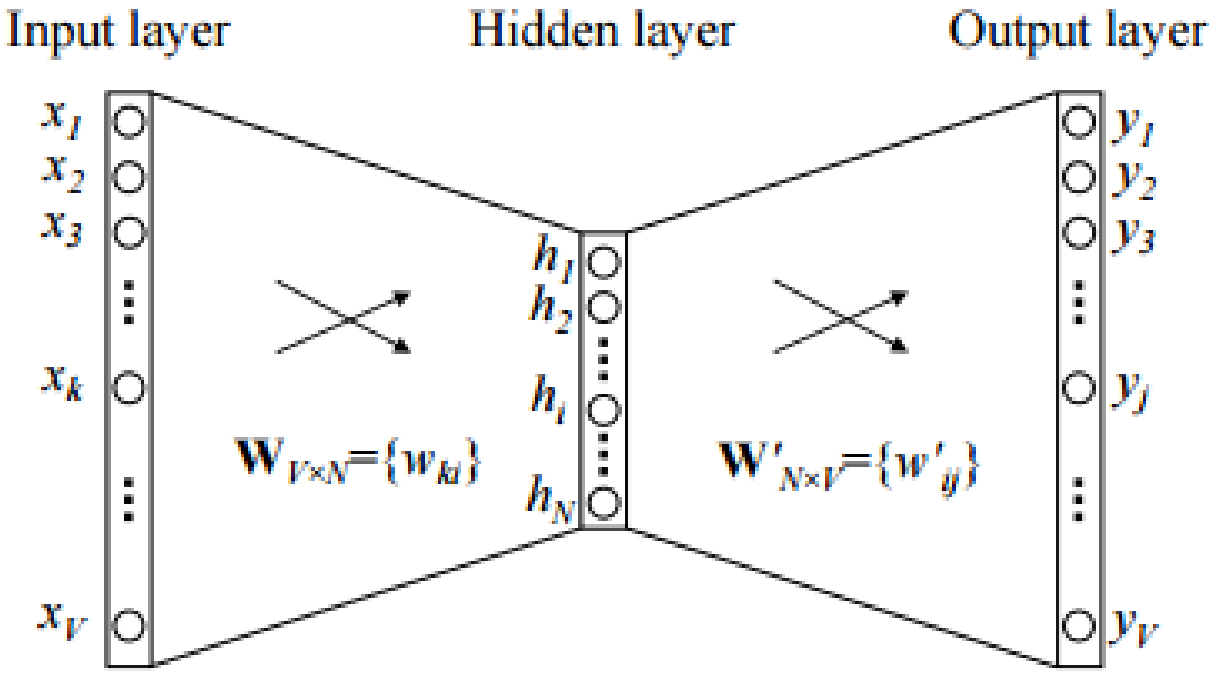
In K-Means, the one parameter we need to set is "K," or the number of clusters. How should we decide how many clusters to create? Trial and error suggested that small clusters, with an average of only 5 words or so per cluster, gave better results than large clusters with many words. Clustering code is given below. We use scikit-learn to perform our K-Means.

K-Means clustering with large K can be very slow; the following code took more than 40 minutes on my computer. Below, we set a timer around the K-Means function to see how long it takes.

# Review of Word2Vec

Simple Word2Vec Architecture

- Amongst vocabulary of size V, let's say we want to predict one target word(output) when we are given one context word (input) → bigram structure
- Input vector is one-hot encoded vector (only one node with for designated context word will be 1)
- $W_{VXN}$ and $W'_{NXV}$ are different matrix
- Output transformed through soft-max



# Review of Word2Vec

Simple Word2Vec Architecture

- Output transformed through soft-max
- $v_{w_I}$: vector representation of the input context word $w_I$ ($x^T W$, k-th row of W)
- $v'_{w'_j}$: vector representation of the output word (j-th column of W')
- Training objective is to maximize this probability, conditional probability of observing the actual output word $w_O$ given the input context word $w_I$
- E = loss function used for finding the gradient to propagate the error to the weight matrix

$$p(w_j|w_I) = \frac{\exp\left(\mathbf{v}'_{w_O}{}^T \mathbf{v}_{w_I}\right)}{\sum_{j'=1}^{V} \exp\left(\mathbf{v}'_{w'_j}{}^T \mathbf{v}_{w_I}\right)}$$

$$\max \log p(w_O|\, w_I) =$$

$$v'_{w_O}{}^T v_{w_I} - \log \sum_{j'=1}^{V} \exp\left(v'_{w'_j}{}^T v_{w_I}\right) := -E$$

# Approach 1: Average Pooling Approach

1. **Xing, Chao, et al. "Document classification with distributions of word vectors." Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA). IEEE, 2014.**

- **Approach 1:** Simple average pooling approach

$$v_i = \frac{1}{J_i} \sum_{j=1}^{J_i} c_{i,j}$$

  - Derives a document vector as the centroid of word vectors within the document
  - But words from different classes of documents will have different distributions
  - Bias towards words without significant contribution to representing the semantics of the documents
  - Word order neglected

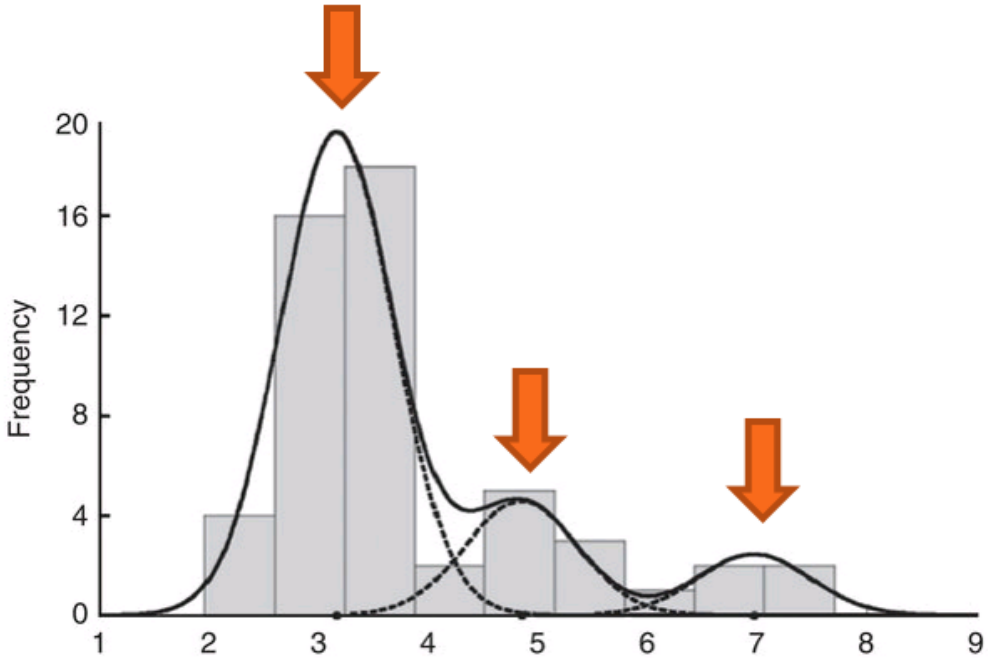
[Doc 1] = "I am Batman"

<u>Word2Vec:</u>
I = [0.05, 0.55, 0.4]
am = [0.35, 0.4, 0.25]
Batman = [0.07, 0.03, 0.9]

Doc 1 Representation (average pooling) = [0.1567, 0.3267, 0.5167]

# Approach 2: Class-Specific Gaussian Mixture Distribution

- **Approach 2:** Class-Specific Gaussian Mixture Distribution (CSGMM)
  - Word vectors within a class of documents assumed to follow Gaussian mixture distributions
  - Gaussian Mixture Model = data points are formed from aggregation of multiple normal distributions
  - In a sense, each distribution from the mixture model can be regarded as one of topics with the document



# Approach 2: Class-Specific Gaussian Mixture Distribution

- **Approach 2:** Class-Specific Gaussian Mixture Distribution (CSGMM)

  - Number of Document Classes = K, Number of Gaussian Components = M
  - Probability of a word vector $C_{i,j}$ in one of the classes in K:

$$p_k(c_{i,j}) = \sum_m \pi_{k,m} N(c_{i,j}; \theta_{k,m})$$

  - $\pi_{k,m} =$ mixture weight (prior probability of component m)
  - $\theta_{k,m}$ = Gaussian parameter (covariance with in each component)
  - Parameters $\pi_{k,m}$ and $\theta_{k,m}$ estimated by maximizing following likelihood function:

$$L(\{\theta_m\}, \{\pi_m\}) = \prod_k \prod_{i \in \Delta_k} \prod_j \sum_m \pi_m N(c_{i,j}; \theta_m)$$

  - $\Delta_k$ = documents within training document of class k
  - Due to two parameters affecting the resulting value of likelihood function, Expectation Maximization (EM) algorithm used

# Approach 2: Class-Specific Gaussian Mixture Distribution

- **Approach 2:** Class-Specific Gaussian Mixture Distribution (CSGMM)
  - Once CSGMM are trained, the class of a test document d can be determined as follow:

$$l(d) = \arg\max_{k} P(k|d)$$

  - $P(k|d)$ = Probability that test document d belongs to document class k

$$P(k|d) = \frac{p(d|k)}{\sum_r p(d|r)} = \frac{\prod_{c_j \in d} p_k(c_j)}{\sum_r \prod_{c_j \in d} p_r(c_j)}$$

  - Since classification is directly made in part from GMM, actual no document vectors are derived

# Approach 3: Semantic Space Allocation

- **Approach 3:** Semantic Space Allocation (SSA)
  - Instead of each document class having separate Gaussian components, use global GMM component
  - GMM on entire word vector space (for training set) to generate distributions from samples of word vectors
  - $P(m|d) =$ Probability that test document d belongs to component m

$$P(m|d) = \frac{p(d|m)}{\sum_r p(d|r)} = \frac{\prod_{c_j \epsilon\, d} p_m(c_j)}{\sum_r \prod_{c_j \epsilon d} p_r(c_j)}$$

  - Basically, multiplying each word's probability of belonging to each component
  - Use this posterior probability of each component for representing documents

$$v = [P(1|d), P(2|d), ..., P(M|d)]^T$$

# Data

- **Data: <Chinese articles published from Sohu research center>**

  - 9 different document classes: Automobile, IT, finance, health, sports, tour, education, recruitment, culture and military
  - Total number of documents: 16,110
    - Training: 14,301 (approximately, 1,589 per class)
    - Testing: 1809
  - Applied SCWS word segmentation tool to pre-process the Chinese documents
  - Total number of words: 150,000

# Document classification with distributions of word vectors

- **Result: <1. Average Pooling vs. LDA>**
  - Compared average pooling method with LDA
    - LDA: one of the most effective “pre-word2vec” method for representing documents
    - SVD on word co-occurrence matrix
    - Each component of SVD represents a specific topic of a document

TABLE I
CLASSIFICATION PRECESSION WITH AVERAGE POOLING

| Classifier | W2V | LDA |
|---|---|---|
| NB | 72% | 63% |
| k-NN | 83.91% | 81.21% |
| SVM | 83.91% | 70.04% |

# Experiment Result

- **<1. Average Pooling vs. LDA>**

  - Compared average pooling method with LDA for document classification
    - LDA: one of the most effective “pre-word2vec” method for representing documents
    - SVD on word co-occurrence matrix
    - Each component of SVD represents a specific topic of a document

TABLE I
CLASSIFICATION PRECESSION WITH AVERAGE POOLING

| Classifier | W2V | LDA |
|---|---|---|
| NB | 72% | 63% |
| k-NN | 83.91% | 81.21% |
| SVM | 83.91% | 70.04% |

# Experiment Result

- **<2. Average Pooling vs. CSGMM vs. SSA>**

  - Compared three suggested methods
    - X axis: represents varying number of Gaussian mixture components
  - Surprisingly, both CSGMM and SSA perform worse than average pooling
    - CSGMM can even perform worse than traditional LDA
  - Hybrid approach of SSA and average pooling does not improve the accuracy

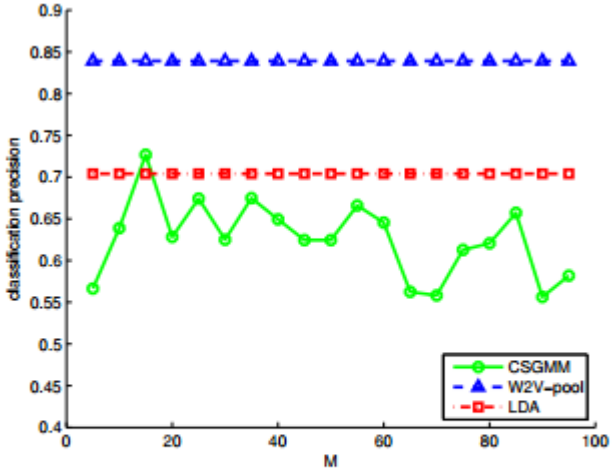



Fig. 2. Performance of the CSGMM model.

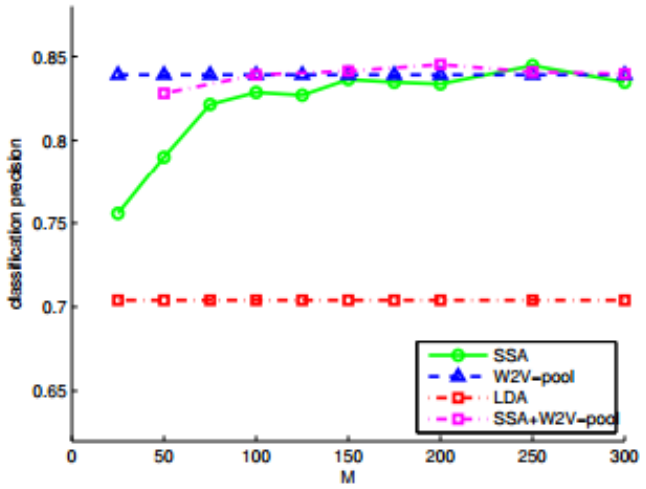



Fig. 3. Performance of the SSA model.

# Conclusion

- **Conclusion**
  - For word2vec based document representation, average pooling method outperforms LDA and GMM
    - Simple word2vec average pooling is still powerful!
  - Research attempts on utilizing word2vec for representing documents are almost non-existent
  - Even those few papers (including this paper) require training set for classifying or clustering the documents
    - Defeats the whole purpose of word2vec and doc2vec! (unsupervised)

# Course of Action

- **Course of Action**

1. Sample(proof of concept) Experiment Result Documentation
2. Final Experiment
3. Background Research on word2vec based document representation
4. Background Research on **constructing ontology for labeling the clusters**

# Reference

Le, Quoc V., and Tomas Mikolov. "Distributed representations of sentences and documents." arXiv preprint arXiv:1405.4053 (2014).

Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in Neural Information Processing Systems. 2013.

R. Liu, D. Wang, and C. Xing, “Document classification based on word vectors.” ISCSLP, 2014

Rong, Xin. "word2vec Parameter Learning Explained." arXiv preprint arXiv:1411.2738 (2014).

Xing, Chao, et al. "Document classification with distributions of word vectors." Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA). IEEE, 2014.