

DSME 6635: Artificial Intelligence for Business Research

## Deep-Learning-based NLP: Word2Vec, Evaluations, & Applications

Renyu (Philip) Zhang

1

## Agenda

- Word2Vec: Continuous Bag of Words (CBOW), Skip-Gram, GloVe
- Word2Vec Evaluations
- Applications of Word2Vec in Business/Econ Research

2

2

# Text as Data

Journal of Economic Literature 2019, 57(3), 535–574  
https://doi.org/10.1257/jel.20181020

**Text as Data<sup>†</sup>**

MATTHEW GENTZKOW, BRYAN KELLY, AND MATT TADDY<sup>\*</sup>

An ever-increasing share of human interaction, communication, and culture is recorded as digital text. We provide an introduction to the use of text as an input to economic research. We discuss the features that make text different from other forms of data, offer a practical overview of relevant statistical methods, and survey a variety of applications. (JEL C38, C55, LS2, Z13)

3

**0. Pre-processing:**

1. Represent raw text  $\mathcal{D}$  as a numerical array  $\mathbf{C}$ ;
2. Map  $\mathbf{C}$  to predicted values  $\hat{\mathbf{V}}$  of unknown outcomes  $\mathbf{V}$ ; and
3. Use  $\hat{\mathbf{V}}$  in subsequent descriptive or causal analysis.

3

# NLP Roadmap

**Attention is all you need**  
A Vaswani, N Shazeer, N Parmar, ... - Advances in neural ... , 2017 - proceedings.neurips.cc  
... the number of attention heads and the attention key and value dimensions, keeping the amount of computation constant, as described in Section 3.2.2. While single-head attention is 0.9 ...  
★ Save 99 Cite Cited by 36430 Related articles All 35 versions 80

**Probability & Vector Representations of NLP**  
Before 2010

**Word2vec & Seq2seq**  
2013-2017

**Transformer-based Models**  
2018-2022

**Large Language Models (LLM)**  
2023-now

**Efficient estimation of word representations in vector space**  
T Mikolov, K Chen, G Corrado, J Dean - arXiv preprint arXiv:1301.3781, 2013 - arxiv.org  
... vector  $X = \text{vector}(\text{"biggest"}) - \text{vector}(\text{"big"}) + \text{vector}(\text{"small"})$ . Then, we search in the vector space for the word ... question (we discard the input question words during this search). When the ...  
★ Save 99 Cite Cited by 27227 Related articles All 48 versions 80

**Glove: Global vectors for word representation**  
J Pennington, R Socher, - Proceedings of the 2014 ... , 2014 - aclanthology.org  
... We use our insights to construct a new model for word representation which we call **GloVe**, for **Global Vectors**, because the **global** corpus statistics are captured directly by the model. ...  
★ Save 99 Cite Cited by 26265 Related articles All 34 versions 80

**Bert: Pre-training of deep bidirectional transformers for language understanding**  
J Devlin, MW Chang, K Lee, K Toutanova - arXiv preprint arXiv: ..., 2018 - arxiv.org  
... We introduce **BERT** and its detailed implementation in this ... For finetuning, the **BERT** model is first initialized with the pre-trained ... we present **BERT** fine-tuning results on 11 NLP tasks. ...  
★ Save 99 Cite Cited by 34042 Related articles All 41 versions 80

**Language models are unsupervised multitask learners**  
A Radford, J Wu, R Child, D Luan, D Amodei, - OpenAI blog, 2019 - patagonia.com  
Natural language processing tasks, such as question answering, machine translation, reading comprehension, and language modeling, have been approached with unsupervised learning on language-specific datasets. We demonstrate that language models begin to learn these tasks without any explicit supervision when trained on a new dataset of millions of webpages called WebText. When conditioned on a document plus questions, the answers generated by the language model reach 55 F1 on the CoQA dataset-matching or exceeding ...  
★ Save 99 Cite Cited by 2834 Related articles All 16 versions 80

**Training language models to follow instructions with human feedback**  
LQuyang, J Wu, X Jiang, D Almeida, - Advances in ..., 2022 - proceedings.neurips.cc  
Making language models bigger does not inherently make them better at following a user's intent. For example, large language models can generate outputs that are untruthful, toxic, or simply not helpful to the user. In other words, these models are not aligned with their users. In this paper, we show an avenue for aligning language models with user intent on a wide range of tasks by fine-tuning with human feedback. Starting with a set of labeler-written prompts and prompts submitted through a language model API, we collect a dataset of ...  
★ Save 99 Cite Cited by 4049 Related articles All 10 versions 80

4

## Word2Vec: Continuous Bag of Words (CBOW)

- References:
  - <https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture01-wordvecs1-public.pdf>
  - [https://web.stanford.edu/class/cs224n/readings/cs224n\\_winter2023\\_lecture1\\_notes\\_draft.pdf](https://web.stanford.edu/class/cs224n/readings/cs224n_winter2023_lecture1_notes_draft.pdf)
- So far, our word representations are based on **word-document co-occurrence matrices**.
- Word embedding is a way to obtain word representations via prediction, called **self-supervised learning**.
- Distributional semantics: A word's meaning is provided by the words that **frequently appear close-by**.
- Context of a word w: The set of words that appear **nearby with a fixed window size**.

...government debt problems turning into **banking** crises as happened in 2009...

...saying that Europe needs unified **banking** regulation to replace the hodgepodge...

...India has just given its **banking** system a shot in the arm...

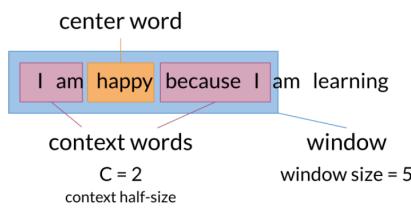
These **context words** will represent **banking**

5

5

## Word2Vec: Continuous Bag of Words (CBOW)

- References:
  - <https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture01-wordvecs1-public.pdf>
  - [https://web.stanford.edu/class/cs224n/readings/cs224n\\_winter2023\\_lecture1\\_notes\\_draft.pdf](https://web.stanford.edu/class/cs224n/readings/cs224n_winter2023_lecture1_notes_draft.pdf)
- Continuous Bag of Words (CBOW): Use the **outside words (o)** to predict the **center word (c)**.
- Skip-gram: Use the **center word (c)** to predict the distribution of **outside words (o)**.



$$\begin{aligned}
 \text{minimize } J &= -\log P(w_c | w_{c-m}, \dots, w_{c-1}, w_{c+1}, \dots, w_{c+m}) \\
 &= -\log P(u_c | \hat{v}) \\
 &= -\log \frac{\exp(u_c^T \hat{v})}{\sum_{j=1}^{|V|} \exp(u_j^T \hat{v})} \\
 &= -u_c^T \hat{v} + \log \sum_{j=1}^{|V|} \exp(u_j^T \hat{v})
 \end{aligned}$$

6

6

## CBOW: Deep Learning Architecture

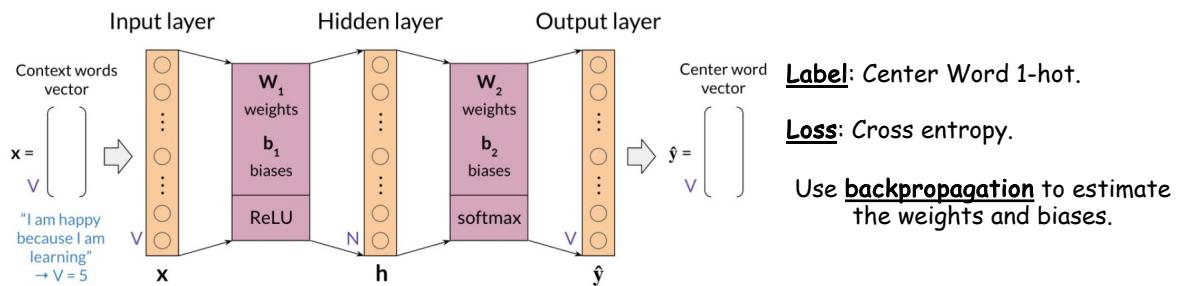
- References:

- <https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture01-wordvecs1-public.pdf>
- [https://web.stanford.edu/class/cs224n/readings/cs224n\\_winter2023\\_lecture1\\_notes\\_draft.pdf](https://web.stanford.edu/class/cs224n/readings/cs224n_winter2023_lecture1_notes_draft.pdf)

$$\left[ \begin{array}{c} \text{I} \\ \text{am} \\ 0 \\ 0 \\ \text{because} \\ 0 \\ 0 \\ \text{happy} \\ 0 \\ 1 \\ \text{I} \\ 1 \\ \text{learning} \\ 0 \end{array} \right] + \left[ \begin{array}{c} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{array} \right] + \left[ \begin{array}{c} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{array} \right] + \left[ \begin{array}{c} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{array} \right] \right] / 4 = \left[ \begin{array}{c} 0.25 \\ 0.25 \\ 0 \\ 0.5 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{array} \right]$$

I am because I am happy because I am learning.

Center Word (c)

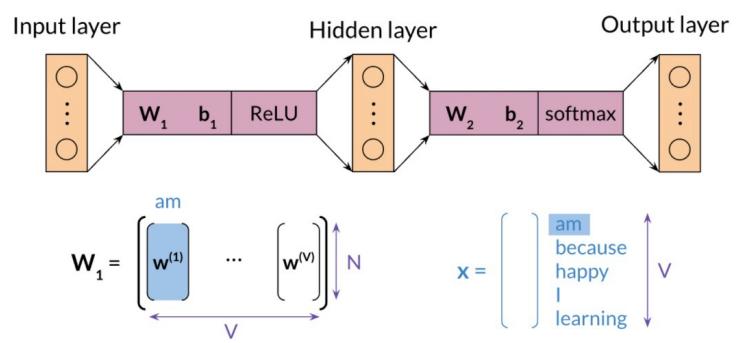


7

7

## CBOW: Everything Together

- Convert the text into training data (center word, outside words).
- Create a 3-layer NN (Dense + ReLU + Softmax)
- Use back propagation to train the model.
- Extract the word embeddings with the model.
- Test the performance of the word embeddings.



If you were to use  $w_1$ , each column will correspond to the embeddings of a specific word. You can also use  $w_2$  as follows:

$$W_2 = \begin{bmatrix} w^{(1)} \\ \dots \\ w^{(V)} \end{bmatrix}$$

$x = \begin{bmatrix} \vdots \\ V \end{bmatrix}$

8

8

## More on CBOW

- Original code: <https://github.com/tmikolov/word2vec>
- Trained on Google News Corpus with 6B Tokens (~10Bb)
  - The widely used public data set wikiText 103 has 100M tokens.
  - <https://pytorch.org/text/stable/datasets.html#id16>
- Trained vectors can be found here:
  - <https://github.com/mmmihaltz/word2vec-GoogleNews-vectors>
- Required computational power:

Table 6: Comparison of models trained using the DistBelief distributed framework. Note that training of NNLM with 1000-dimensional vectors would take too long to complete.

Model	Vector Dimensionality	Training words	Accuracy [%]			Training time [days x CPU cores]
			Semantic	Syntactic	Total	
NNLM	100	6B	34.2	64.5	50.8	14 x 180
CBOW	1000	6B	57.3	68.9	63.7	2 x 140
Skip-gram	1000	6B	66.1	65.1	65.6	2.5 x 125

Efficient estimation of word representations in vector space  
T.Mikolov, X.Chen, G.Cortada, J.Dean - arXiv preprint arXiv:1301.3781, 2013 - arxiv.org  
vector  $X = \text{vector}(\text{"biggest"}) + \text{vector}(\text{"big"}) + \text{vector}(\text{"small"})$ . Then, we search in the vector space for the word  $\ldots$  question (we discard the input question words during this search). When the ...  
☆ Save 00 Cite Cited by 28076 Related articles All 48 versions 00

Distributed representations of words and phrases and their compositionality  
T.Mikolov, I.Sutskever, K.Chen - Advances in neural information processing systems, 2013 - neurips.cc  
The recurrent skip-gram model uses a single shared vector for each word in a vocabulary to learn high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships. In this paper we present several improvements that make the Skip-gram model more expressive and enable it to learn higher quality vectors more rapidly. We show that by subsampling frequent words we obtain significant speedup, and also learn higher quality representations as measured by our tasks. We also introduce ...  
☆ Save 00 Cite Cited by 33549 Related articles All 56 versions 00

NeurIPS 2023 Test-of-Time Award

Table 2: Accuracy on subset of the Semantic-Syntactic Word Relationship test set, using word vectors from the CBOW architecture with limited vocabulary. Only questions containing words from the most frequent 30k words are used.

Dimensionality / Training words	24M	49M	98M	196M	391M	783M
50	13.4	15.7	18.6	19.1	22.5	23.2
100	19.4	23.1	27.8	28.7	33.4	32.2
300	23.2	29.2	35.3	38.6	43.7	45.9
600	24.0	30.1	36.5	40.8	46.6	50.4



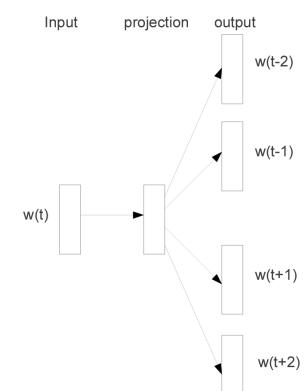
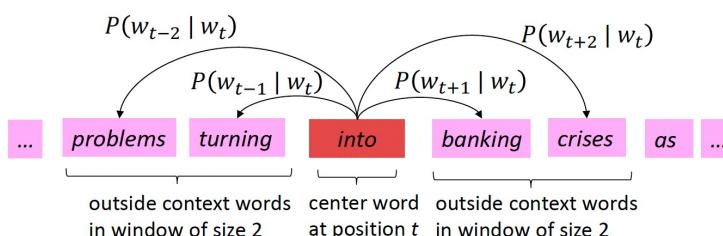
9

9

## Word2Vec: Skip-Gram

- References:
  - <https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture01-wordvecs1-public.pdf>
  - [https://web.stanford.edu/class/cs224n/readings/cs224n\\_winter2023\\_lecture1\\_notes\\_draft.pdf](https://web.stanford.edu/class/cs224n/readings/cs224n_winter2023_lecture1_notes_draft.pdf)
- Skip-gram: Like CBOW but the inverse, using the center word to predict the outside words.

Example windows and process for computing  $P(w_{t+1} | w_t)$



10

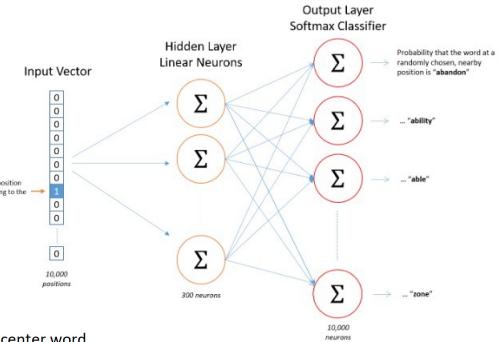
10

## Word2Vec: Skip-Gram

- References:

- <https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture01-wordvecs1-public.pdf>
- [https://web.stanford.edu/class/cs224n/readings/cs224n\\_winter2023\\_lecture1\\_notes\\_draft.pdf](https://web.stanford.edu/class/cs224n/readings/cs224n_winter2023_lecture1_notes_draft.pdf)

Source Text	Training Samples
The quick brown fox jumps over the lazy dog.	(the, quick) (the, brown)
The quick brown fox jumps over the lazy dog.	(quick, the) (quick, brown) (quick, fox)
The quick brown fox jumps over the lazy dog.	(brown, the) (brown, quick) (brown, fox) (brown, jumps)
The quick brown fox jumps over the lazy dog.	(fox, quick) (fox, brown) (fox, jumps) (fox, over)



We want to minimize the objective function:

$$J(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_{t+j} | w_t; \theta)$$

Then for a center word  $c$  and a context word  $o$ :

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

11

11

## Approximation Training

- References:

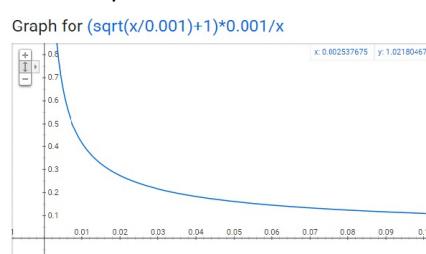
- <https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture02-wordvecs2.pdf>
- [https://web.stanford.edu/class/cs224n/readings/cs224n\\_winter2023\\_lecture1\\_notes\\_draft.pdf](https://web.stanford.edu/class/cs224n/readings/cs224n_winter2023_lecture1_notes_draft.pdf)

An issue with skip-gram is that computing the probability over **all possible words in the dictionaries** is very **computationally costly** when the dictionary size is large.

**Subsampling of words:** We do not want to sample the too frequent words (appearing in the neighbor of every word).

- $P(w_i)$  = the probability word  $w_i$  is being kept.
- $z(w_i)$  = the fraction of appearances of word  $w_i$  in the corpus.

$$P(w_i) = \left( \sqrt{\frac{z(w_i)}{0.001}} + 1 \right) \cdot \frac{0.001}{z(w_i)}$$



12

12

# Approximation Training

- References:
  - <https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture02-wordvecs2.pdf>
  - [https://web.stanford.edu/class/cs224n/readings/cs224n\\_winter2023\\_lecture1\\_notes\\_draft.pdf](https://web.stanford.edu/class/cs224n/readings/cs224n_winter2023_lecture1_notes_draft.pdf)
- **Negative Sampling:** At each observation, we observe a positive case (numerator), and we only use a small set of negative cases (denominator) to update the weights, saving a lot of computations.
  - When training the network on the word pair ("fox", "quick"), recall that the "label" or "correct output" of the network is a one-hot vector. That is, for the output neuron corresponding to "quick" to output a 1, and for all the other thousands of output neurons to output a 0.
  - With negative sampling, we are instead going to randomly select just a small number of "negative" words (let's say 5) to update the weights for. (In this context, a "negative" word is one for which we want the network to output a 0 for). We will also still update the weights for our "positive" word (which is the word "quick" in our current example).
- **Question:** If your embedding size is 300 and you have 10,000 unique words, when you train on (quick, x), if you don't use negative sampling, how many weights are you updating? If you use 5-word negative sampling, how many weights are you updating?

13

13

# Word2Vec: GloVe

(PDF) GloVe: Global vectors for word representation  
 J.Pennington, R.Socher - Proceedings of the 2014 ... 2014 · aclanthology.org  
 We use our insights to construct a new model for word representation which we call GloVe.  
 for Global Vectors, because the global corpus statistics are captured directly by the model...  
 ☆ Save 99 Cite Cited by 26265 Related articles All 34 versions 88

- References:
  - <https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture02-wordvecs2.pdf>
  - <https://web.stanford.edu/class/cs224n/readings/cs224n-2019-notes02-wordvecs2.pdf>
- Global Vectors for Word Representation (GloVe): Word2Vec based on word co-occurrence matrices.
- Window-based co-occurrence matrix:

Window length 1 (more common: 5–10)

Symmetric (irrelevant whether left or right context)

Example corpus:

- I like deep learning
- I like NLP
- I enjoy flying

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

## Co-occurrence Matrix:

- X: word-word co-occurrence matrix
- $X_{ij}$ : number of times word  $j$  occur in the context of word  $i$
- $X_i = \sum_k X_{ik}$ : the number of times any word  $k$  appears in the context of word  $i$
- $P_{ij} = P(w_j|w_i) = \frac{X_{ij}}{X_i}$ : the probability of  $j$  appearing in the context of word  $i$

14

14

## Word2Vec: GloVe

- Model:

$$Q_{ij} = \frac{\exp(\vec{u}_j^T \vec{v}_i)}{\sum_{w=1}^W \exp(\vec{u}_w^T \vec{v}_i)}$$

- Adjusted Square loss:  $\hat{J} = \sum_{i=1}^W \sum_{j=1}^W X_i (\hat{P}_{ij} - \hat{Q}_{ij})^2$

where  $\hat{P}_{ij} = X_{ij}$  and  $\hat{Q}_{ij} = \exp(\vec{u}_j^T \vec{v}_i)$

- Some approximations:

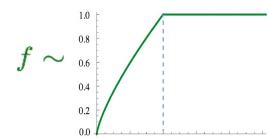
$$\hat{J} = \sum_{i=1}^W \sum_{j=1}^W X_i (\log(\hat{P})_{ij} - \log(\hat{Q}_{ij}))^2$$

$$= \sum_{i=1}^W \sum_{j=1}^W X_i (\vec{u}_j^T \vec{v}_i - \log X_{ij})^2$$

Final Loss Function to Minimize:

Loss:  $J = \sum_{i,j=1}^V f(X_{ij}) (w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2$

- Fast training
- Scalable to huge corpora



15

15

## Agenda

- Word2Vec: Continuous Bag of Words (CBOW), Skip-Gram, GloVe
- Word2Vec Evaluations
- Applications of Word2Vec in Business/Econ Research

16

16

## How to Evaluate Language Models?

- Reference: <https://web.stanford.edu/class/cs224n/slides/cs224n-2024-lecture02-wordvecs2.pdf>
- We know how to evaluate downstream supervised learning tasks with a defined loss function.
- However, what if you are given word vectors or n-gram model, how do you evaluate their performance?
- Intrinsic Evaluation:**
  - Measure the quality of a model on a **specific/intermediate task independent of any application**.
  - For example, an intrinsic evaluation is analog test. It allows you to capture semantic **analogies** as, "France" is to "Paris" as "Italy" is to **<?>** and syntactic analogies as "seen" is to "saw" as "been" is to **<?>**.
  - Fast to compute, helpful to understand the system, but unclear how it correlates with real tasks.
  - Analogies, similarities, etc.
- Extrinsic Evaluation:**
  - The best way to evaluate the performance of a language model is to embed it in an application and **measure how much the application improves**.
  - Computationally costly**.
  - Unclear which part of the system (the subsystem or its interactions to other subsystems) should the performance changes be attributed to.
  - Name Entity Recognition (NER)**, etc.

17

17

## Intrinsic Evaluation: Word Analogy

- Word analogy task:**

When given a pair of words  $a$  and  $a^*$  and a third word  $b$ , the analogy relationship between  $a$  and  $a^*$  can be used to find the corresponding word  $b^*$  to  $b$ . Mathematically, it is expressed as

$$a : a^* :: b : \_, \quad (7)$$

where the blank is  $b^*$ . One example could be

$$\text{write} : \text{writing} :: \text{read} : \text{reading}. \quad (8)$$

The 3CosAdd method [33] solves for  $b^*$  using the following equation:

$$b^* = \underset{b'}{\operatorname{argmax}}(\cos(b', a^* - a + b)), \quad (9)$$

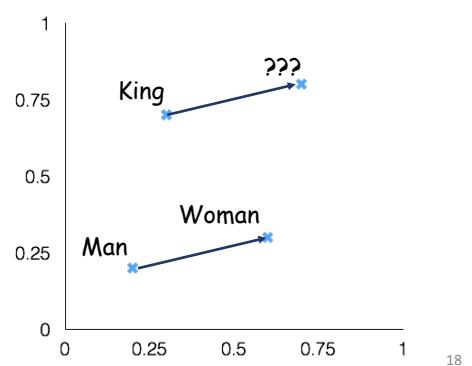
- You should remove the input vectors from the search of  $b'$ .

- Issue:** What if the information is there but nonlinear?

- Datasets:

[https://aclweb.org/aclwiki/Google\\_analogy\\_test\\_set\\_\(State\\_of\\_the\\_art\)](https://aclweb.org/aclwiki/Google_analogy_test_set_(State_of_the_art))

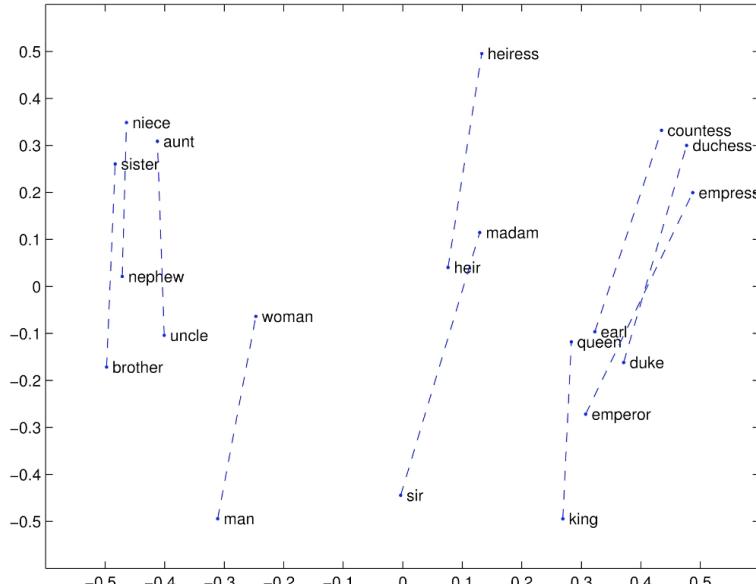
	Word Analogy Datasets							
	Google		Semantic		Syntactic		MSR	
	Add	Mul	Add	Mul	Add	Mul	Add	Mul
SGNS	<b>71.8</b>	<b>73.4</b>	<b>77.6</b>	<b>78.1</b>	67.1	<b>69.5</b>	<b>56.7</b>	<b>59.7</b>
CBOW	70.7	70.8	74.4	74.1	<b>67.6</b>	68.1	56.2	56.8
GloVe	68.4	68.7	76.1	75.9	61.9	62.7	50.3	51.6
FastText	40.5	45.1	19.1	24.8	58.3	61.9	48.6	52.2
ngram2vec	70.1	71.3	75.7	75.7	65.3	67.6	53.8	56.6
Dict2vec	48.5	50.5	45.1	47.4	51.4	53.1	36.5	38.9



18

18

## GloVe Analogy Visualization



19

19

## Intrinsic Evaluation: Word Similarity

- Given two representations of two words, we can compute their similarities:

$$\cos(w_x, w_y) = \frac{w_x \cdot w_y}{\|w_x\| \|w_y\|},$$

- Example dataset: WordSim353:  
<https://www.kaggle.com/datasets/julianschelb/wordsim353-crowd>
- The following is a summary of word similarity test datasets:

TABLE I  
WORD SIMILARITY DATASETS USED IN OUR EXPERIMENTS WHERE PAIRS  
INDICATE THE NUMBER OF WORD PAIRS IN EACH DATASET.

Name	Pairs	Year
WS-353 [40]	353	2002
WS-353-SIM [41]	203	2009
WS-353-REL [41]	252	2009
MC-30 [42]	30	1991
RG-65 [43]	65	1965
Rare-Word (RW) [44]	2034	2013
MEN [45]	3000	2012
MTurk-287 [46]	287	2011
MTurk-771 [47]	771	2012
YP-130 [48]	130	2006
SimLex-999 [49]	999	2014
Verb-143 [50]	143	2014
SimVerb-3500 [51]	3500	2016

Evaluating word embedding models: methods and experimental results  
[B Wang, A Wang, F Chen, Y Wang... - APSIPA transactions on ...](#), 2019 - cambridge.org  
 Extensive evaluation on a large number of word embedding models for language processing applications is conducted in this work. First, we introduce popular word embedding models and discuss desired properties of word models and evaluation methods (or evaluators). Then, we categorize evaluators into intrinsic and extrinsic two types. Intrinsic evaluators test the quality of a representation independent of specific natural language processing tasks while extrinsic evaluators use word embeddings as input features to a ...  
☆ Save 59 Cite Cited by 188 Related articles All 6 versions »

20

20

## Intrinsic Evaluation: Correlation

- Word vector distances and correlations with human judgments.

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	72.7	75.1	56.5	37.0
CBOW <sup>†</sup>	6B	57.2	65.6	68.2	57.0	32.5
SG <sup>†</sup>	6B	62.8	65.2	69.7	58.1	37.2
GloVe	6B	65.8	72.7	77.8	53.9	38.1
SVD-L	42B	74.0	76.4	74.1	58.3	39.9
GloVe	42B	75.9	83.6	82.9	59.6	47.8
CBOW*	100B	68.4	79.6	75.4	59.4	45.5

21

21

## Extrinsic Evaluation: Named Entity Recognition

- Named Entity Recognition (NER): Identify the references to a person, location, organization, time, etc.  
[https://en.wikipedia.org/wiki/Named-entity\\_recognition](https://en.wikipedia.org/wiki/Named-entity_recognition)

Philip lives in Shenzhen and Shanghai in 2024.

Person                  Location                  Time

- Use window classification with a logistic classifier for NER.
- Classify each human-labeled word using the neighboring words in its context window.

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

**Example:** Classify “Paris” as +/– location in context of sentence with window length 2:

the    museums    in    Paris    are    amazing    to    see    .

$$\mathbf{x}_{\text{window}} = [x_{\text{museums}} \quad x_{\text{in}} \quad x_{\text{Paris}} \quad x_{\text{are}} \quad x_{\text{amazing}}]^T$$

22

22

## Example: GloVe

### Analogy

Table 2: Results on the word analogy task, given as percent accuracy. Underlined scores are best within groups of similarly-sized models; bold scores are best overall. HPCA vectors are publicly available<sup>2</sup>; (i)LBL results are from (Mnih et al., 2013); skip-gram (SG) and CBOW results are from (Mikolov et al., 2013a,b); we trained SG<sup>†</sup> and CBOW<sup>†</sup> using the word2vec tool<sup>3</sup>. See text for details and a description of the SVD models.

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	<u>67.5</u>	<u>54.3</u>	60.3
SG	300	1B	61	61	61
CBOW	300	1.6B	16.1	52.6	36.1
vLBL	300	1.5B	54.2	<u>64.8</u>	60.0
ivLBL	300	1.5B	65.2	63.0	64.0
GloVe	300	1.6B	<u>80.8</u>	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 <sup>†</sup>	300	6B	63.6	<u>67.4</u>	65.7
SG <sup>†</sup>	300	6B	73.0	66.0	69.1
GloVe	300	6B	<u>77.4</u>	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	<b>81.9</b>	<b>69.3</b>	<b>75.0</b>

### Similarity

Table 3: Spearman rank correlation on word similarity tasks. All vectors are 300-dimensional. The CBOW\* vectors are from the word2vec website and differ in that they contain phrase vectors.

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 <sup>†</sup>	6B	57.2	65.6	68.2	57.0	32.5
SG <sup>†</sup>	6B	62.8	65.2	69.7	<u>58.1</u>	37.2
GloVe	6B	<u>65.8</u>	72.7	77.8	53.9	<u>38.1</u>
SVD-L	42B	74.0	76.4	74.1	58.3	39.9
GloVe	42B	<b>75.9</b>	<b>83.6</b>	<b>82.9</b>	<b>59.6</b>	<b>47.8</b>
CBOW*	100B	68.4	79.6	75.4	59.4	45.5

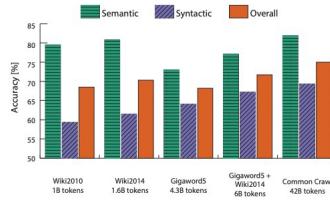


Figure 3: Accuracy on the analogy task for 300-dimensional vectors trained on different corpora. **Analogy**

Table 4: F1 score on NER task with 50d vectors. *Discrete* is the baseline without word vectors. We use publicly-available vectors for HPCA, HSMN, and CW. See text for details.

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	<b>88.7</b>	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	<b>93.2</b>	88.3	<b>82.9</b>	<b>82.2</b>

NER

[PDF] **Glove: Global vectors for word representation**

J Pennington, R Socher... - Proceedings of the 2014 ... , 2014 - aclanthology.org

... **Glove** ... **Glove** model outperforms all other methods on all evaluation metrics, except for the CoNLL test set, on which the HPCA method does slightly better. We conclude that the **Glove** ...

☆ Save 99 Cite Cited by 38846 Related articles All 27 versions 00

23

## Agenda

- Word2Vec: Continuous Bag of Words (CBOW), Skip-Gram, GloVe
- Word2Vec Evaluations
- Applications of Word2Vec in Business/Econ Research

24

24

# Application: Text Algorithms in Economics

*Annual Review of Economics*

## Text Algorithms in Economics

Elliott Ash<sup>1</sup> and Stephen Hansen<sup>2,3</sup>

<sup>1</sup>Center for Law and Economics, ETH Zurich, Zurich, Switzerland

<sup>2</sup>Department of Economics, University College London, London, United Kingdom; email: stephen.hansen@ucl.ac.uk

<sup>3</sup>Centre for Economic Policy Research, London, United Kingdom

### Keywords

text as data, topic models, word embeddings, large language models, transformer models

### Abstract

This article provides an overview of the methods used for algorithmic text analysis in economics, with a focus on three key contributions. First, we introduce methods for representing documents as high-dimensional count vectors over vocabulary terms, for representing words as vectors, and for representing word sequences as embedding vectors. Second, we define four core empirical tasks that encompass most text-as-data research in economics and enumerate the various approaches that have been taken so far to accomplish these tasks. Finally, we flag limitations in the current literature, with a focus on the challenge of validating algorithmic output.

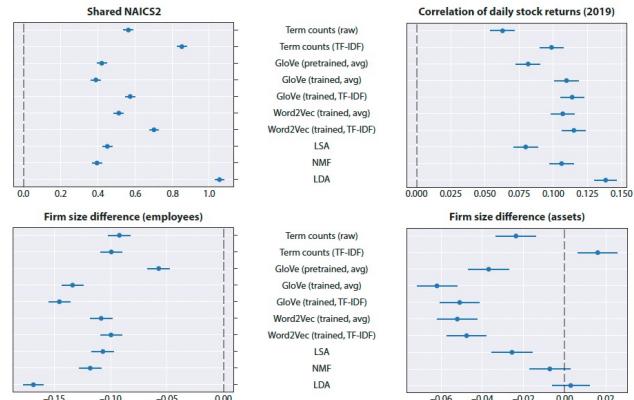
### Text algorithms in economics

E. Ash, S. Hansen - Annual Review of Economics, 2023 - annualreviews.org

... methods used for algorithmic text analysis in economics, with ... encompass most text-as-data research in economics and ... with a focus on the challenge of validating algorithmic output. ....

☆ 保存 99 引用 被引用次数: 41 相关文章 所有 6 个版本 ☰

- **Four problems:** (I) Measuring document similarity; (II) Concept detection; (III) How concepts are related; (IV) Associating text with metadata.
- **Two challenges:** (I) Validation; (II) Interpretability.
- **One future:** Large Language Models.



25

25

# Application: Corporate Culture Measurement

The Review of Financial Studies



## Measuring Corporate Culture Using Machine Learning

Kai Li

Sauder School of Business, University of British Columbia

Feng Mai

School of Business, Stevens Institute of Technology

Rui Shen

Shenzhen Finance Institute, School of Management and Economics  
The Chinese University of Hong Kong, Shenzhen

Xinyan Yan

School of Business Administration, University of Dayton

We create a culture dictionary using one of the latest machine learning techniques—the word embedding model—and 209,480 earnings call transcripts. We score the five corporate cultural values of *innovation, integrity, quality, respect, and teamwork* for 62,664 firm-year observations over the period 2001–2018. We show that an innovative culture is broader than the usual measures of corporate innovation – R&D expenses and the number of patents. Moreover, we show that corporate culture correlates with business outcomes, including operational efficiency, risk-taking, earnings management, executive compensation design, firm value, and deal making, and that the culture-performance link is more pronounced in bad times. Finally, we present suggestive evidence that corporate culture is shaped by major corporate events, such as mergers and acquisitions. (JEL C45, G34, M14)

Received February 15, 2019; editorial decision May 26, 2020 by Editor Itay Goldstein.  
Authors have furnished an Internet Appendix, which is available on the Oxford University Press Web site next to the link to the final published paper online.

- Train **word embeddings** using word2vec on **earnings call transcripts**.
- Create a **culture dictionary** based on the **associations** of different words/phrases and the "**value words**" (*innovation, integrity, quality, respect, and teamwork*) under the trained word embeddings.
- **Business implications:** A strong corporate culture **correlates** with greater operational efficiency, more risk-taking, less earnings management, and higher firm value.

### Measuring corporate culture using machine learning

K. Li, F. Mai, R. Shen, X. Yan  
The Review of Financial Studies, 2021 academic.oup.com

### Abstract

We create a culture dictionary using one of the latest machine learning techniques—the word embedding model—and 209,480 earnings call transcripts. We score the five corporate cultural values of *innovation, integrity, quality, respect, and teamwork* for 62,664 firm-year observations over the period 2001–2018. We show that an innovative culture is broader than the usual measures of corporate innovation – R&D expenses and the number of patents. Moreover, we show that corporate culture correlates with business

展开 ▾

☆ 保存 99 引用 被引用次数: 412 相关文章 所有 14 个版本 Web of Science: 119 ☰

26

26



# Other Applications of Product2Vec

## P2V-MAP: Mapping market structures for large retail assortments

S Gabel, D Guhl, D Klapper

Journal of Marketing Research, 2019 · journals.sagepub.com

The authors propose a new, exploratory approach for analyzing market structures that leverages two recent methodological advances in natural language processing and machine learning. They customize a neural network language model to derive latent product attributes by analyzing the co-occurrences of products in shopping baskets. Applying dimensionality reduction to the latent attributes yields a two-dimensional product map. This method is well-suited to retailers because it relies on data that are readily

SHOW MORE ▾

[☆ Save](#) [引用](#) [Cited by 61](#) [Related articles](#) [All 5 versions](#) [Web of Science: 24](#) [»](#)

## Scalable bundling via dense product embeddings

M Kumar, D Eckles, S Aral

arXiv preprint arXiv:2002.00100, 2020 · arxiv.org

Bundling, the practice of jointly selling two or more products at a discount, is a widely used strategy in industry and a well examined concept in academia. Historically, the focus has been on theoretical studies in the context of monopolistic firms and assumed product relationships, e.g., complementarity in usage. We develop a new machine-learning-driven methodology for designing bundles in a large-scale, cross-category retail setting. We leverage historical purchases and consideration sets created from clickstream data to

展开 ▾

[☆ 保存](#) [引用](#) [被引用次数 : 10](#) [相关文章](#) [所有 3 个版本](#) [»](#)

## Item2vec: neural item embedding for collaborative filtering

O Barkan, N Koenigstein

2016 IEEE 26th International Workshop on Machine Learning for ..., 2016 · ieeexplore.ieee.org

Many Collaborative Filtering (CF) algorithms are item-based in the sense that they analyze item-item relations in order to produce item similarities. Recently, several works in the field of Natural Language Processing (NLP) suggested to learn a latent representation of words using neural embedding algorithms. Among them, the Skip-gram with Negative Sampling (SGNS), also known as word2vec, was shown to provide state-of-the-art results on various linguistics tasks. In this paper, we show that item-based CF can be cast in the

SHOW MORE ▾

[☆ Save](#) [引用](#) [Cited by 600](#) [Related articles](#) [All 13 versions](#) [»](#)

29