

ニューラルネットワークの潜在表現について

— Word2Vec による単語埋込 —

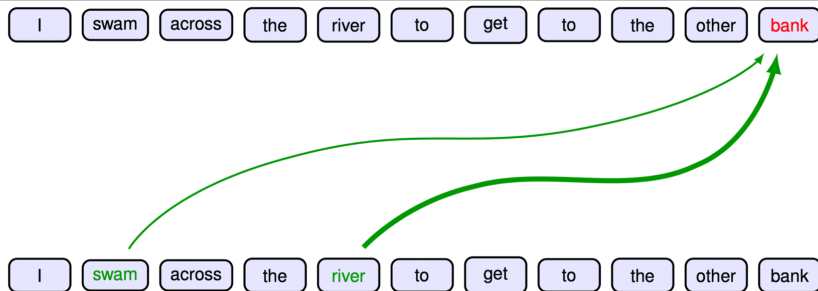
後編

本谷 秀堅

名古屋工業大学

言語データ

- シンボルの系列
- 離れたシンボル間に強い関係
- シンボル間の非対称な関係（例：対義語・類義語・包含関係）

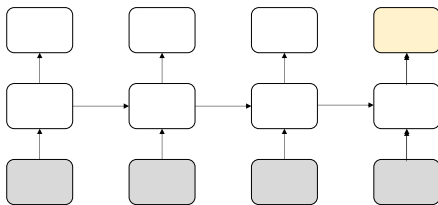


単語を特徴ベクトルへと変換

- 単語をベクトルで表現する。
- AI にとって都合の良いベクトルで表現したい

AIにとって都合の良い表現

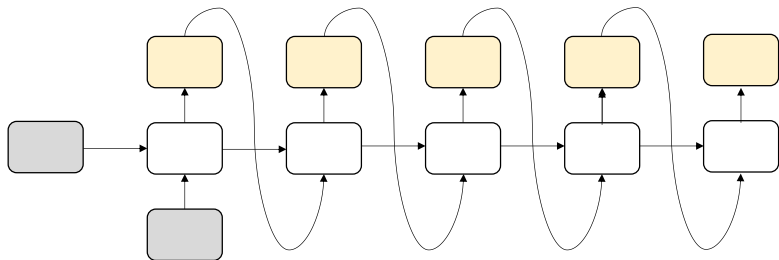
似た概念の単語は似たベクトルのほうが都合が良い



例えば RNN による品詞の推定。最後の層は MLP+Softmax 関数

AIにとって都合の良い表現

似た概念の単語は似たベクトルのほうが都合が良い



例えば RNN による会話生成。最後の層は MLP+Softmax 関数

One-hot-vector の不便さ

各単語を語彙数 K 次元の one-hot-vector で表現する
以下の例では $K = 10$

$$\mathbf{x}_{\text{pen}} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{x}_{\text{spoon}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{x}_{\text{walk}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{x}_{\text{run}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

次のことに気付く

- どのベクトル間の距離も $\sqrt{2}$
- どのベクトルも互いに直交

単語間の関係が距離にも方向にも反映されていない

AI に都合の良いベクトル

K より低次元で、方向が概念を表すベクトル
例えば

$$\mathbf{x}_{\text{pen}} = \begin{bmatrix} 0.2 \\ -0.6 \\ 0.3 \end{bmatrix}, \mathbf{x}_{\text{spoon}} = \begin{bmatrix} 0.2 \\ -0.7 \\ 0.3 \end{bmatrix}, \mathbf{x}_{\text{walk}} = \begin{bmatrix} -0.8 \\ 0.0 \\ 0.2 \end{bmatrix}, \mathbf{x}_{\text{run}} = \begin{bmatrix} -0.7 \\ 0.1 \\ 0.3 \end{bmatrix}$$

- 「pen」と「spoon」は似た方向
- 「walk」と「run」は似た方向
- pen, spoon と walk, run は互いに離れている。

単語間の関係が方向・距離に表れている

異なる単語でも概念が近いと Softmax 関数への入力も近い値に

- 都合の良いベクトルを単語ごとに学習で求める

自己教師学習 (self-supervised learning)

自己教師学習により (AI に都合の良い) 表現を獲得する

- 教師あり学習の一種
- 教師信号を自動生成する (多量のデータで学習可)
- プレテキスト タスク (pretext task)
 - 文章の一部を隠して、残りから予測
 - 文章の後半を隠して、前半から予測
 - 画像の一部を隠して、残りから予測
 - カラー画像を白黒にして、カラーを復元
 - ...

想定する AI のタスク：識別や回帰

Word2Vec (2013)

- CBoW (Continuous Bag-of-Words)
 - 文中で前後の単語から真ん中の単語を予測
- Skip-Gram
 - 文中の各単語について前後の単語を予測

以下の説明は"Rong, word2vec Parameter Learning Explained, 2012, <https://arxiv.org/abs/1411.2738> による

CBoW: Continuous Bag of Words

I swam across the river to get to the other bank.



プレテキストタスク
前後2単語から、中央の単語を予測せよ

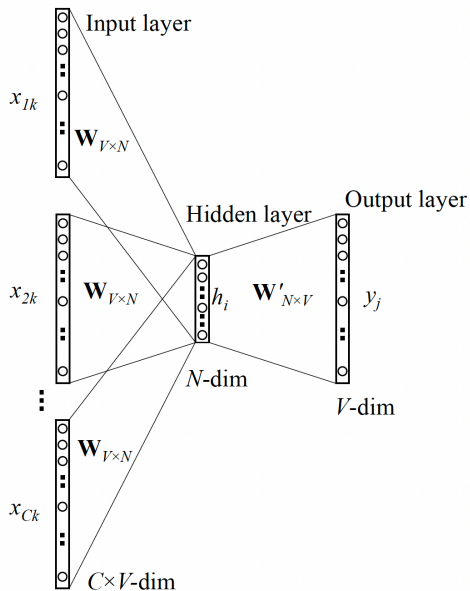
学習データ 1

across the [REDACTED] to get
 x_1 x_2 t x_3 x_4

学習データ 2

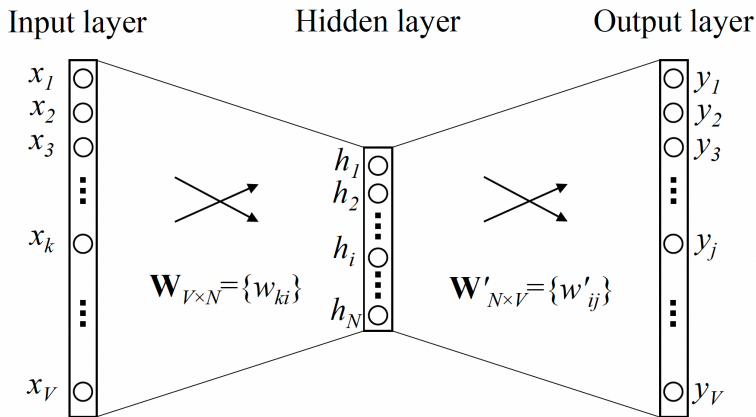
the river [REDACTED] get to
 x_1 x_2 t x_3 x_4

CBoW のアーキテクチャ



CBoW のアーキテクチャ (シンプル版)

入力が1単語だけだと、こうなる



- 1層目も2層目も全結合 (MLP)
- V : 語彙数, D : 中間層のユニット数

式表現（シンプル版）

1 層目から 2 層目

$$\mathbf{h} = \mathbf{W}\mathbf{x}$$

2 層目から 3 層目

$$\mathbf{u} = \mathbf{W}'\mathbf{h}$$

$$y_j = \text{SM}_j(\mathbf{u}) = \frac{\exp(u_j)}{\sum_{j'=1}^V \exp(u_{j'})}$$

前段の全結合層

結線の重みの行列表現

$$W = \begin{bmatrix} \mathbf{w}_1^\top \\ \mathbf{w}_2^\top \\ \vdots \\ \mathbf{w}_D^\top \end{bmatrix} \quad \text{where} \quad \mathbf{w}_i^\top = [w_{i1}, w_{i2}, \dots, w_{iV}]$$

- 中間層の活性化関数は口頭写像 $f(u) = u$
- 中間層の i 番目のユニットの出力 h_i ($i=1, 2, \dots, D$)

$$h_i = \mathbf{w}_i^\top \mathbf{x}$$

V 次元の one-hot ベクトル \mathbf{x} が D 次元のベクトル \mathbf{h} に変換された ($D \ll V$)

$$\mathbf{h} = [h_1, h_2, \dots, h_D]^\top$$

後段の全結合層

結線の重みの行列表現

$$W' = \begin{bmatrix} (\mathbf{w}'_1)^\top \\ (\mathbf{w}'_2)^\top \\ \vdots \\ (\mathbf{w}'_V)^\top \end{bmatrix} \quad \text{where} \quad (\mathbf{w}'_j)^\top = [w'_{j1}, w'_{j2}, \dots, w'_{jD}]$$

最終層の j 番目のユニットの内積計算

$$u_j = (\mathbf{w}'_j)^\top \mathbf{h}$$

最終層の活性化関数はソフトマックス関数（多クラス識別）

$$y_j = \frac{\exp(u_j)}{\sum_{j'=1}^V \exp(u_{j'})}$$

y_j は隠された単語が、辞書の j 番目である事後確率の推定値

$$y_j = P(t_j = 1 | \mathbf{x})$$

学習の損失関数

各問題に対する尤度 $L(\mathbf{W}, \mathbf{W}')$

$$L(\mathbf{W}, \mathbf{W}') = \prod_{j=1}^V P(t_j = 1 | \mathbf{x})^{t_j}$$

辞書の j^* 番目の単語が正解だったとする。

$$t_j = \begin{cases} 1, & j = j^*, \\ 0, & j \neq j^*. \end{cases}$$

損失関数は交差エントロピー（下記を全学習データについて足す）

$$E(\mathbf{W}, \mathbf{W}') = -\log L = -\sum_{j=1}^V t_j \left(u_j - \log \sum_{j'=1}^V \exp(u_{j'}) \right)$$

損失関数の微分 (1/4)

(再掲) 損失関数は交差エントロピー

$$E(\mathbf{W}, \mathbf{W}') = -\log L = -\sum_{j=1}^V t_j \left(u_j - \log \sum_{j'=1}^V \exp(u_{j'}) \right)$$

正解は $j = j^*$

$$= - \left(u_{j^*} - \log \sum_{j'=1}^V \exp(u_{j'}) \right)$$

損失関数の微分 (2/4)

最終層の内積で微分

$$\frac{\partial E}{\partial u_j} = -(t_j - y_j) \equiv e_j$$

最終の結線の重み： $(\mathbf{w}'_j)^\top = [w'_{j1}, w'_{j2}, \dots, w'_{jD}]$

$$\frac{\partial E}{\partial w'_{ji}} = \frac{\partial E}{\partial u_j} \frac{\partial u_j}{\partial w'_{ji}} = e_j \cdot h_i$$

学習率を η で表すと

$$(w'_{ji})^{(\text{new})} = (w'_{ji})^{(\text{old})} - \eta \cdot e_j \cdot h_i$$

後段のパラメータの更新式

$$(\mathbf{w}'_j)^{(\text{new})} = (\mathbf{w}'_j)^{(\text{old})} - \eta \cdot e_j \cdot \mathbf{h}$$

損失関数の微分 (3/4)

中間層の出力 h_i で微分（中間層から最終層は全結合）

$$\frac{\partial E}{\partial h_i} = \sum_{j=1}^V \frac{\partial E}{\partial u_j} \frac{\partial u_j}{\partial h_i} = \sum_{j=1}^V e_j w'_{ji}$$

中間層は入力 \mathbf{W} による線形変換： $h_i = \mathbf{w}_i^\top \mathbf{x}$

$$\frac{\partial E}{\partial w_{ik}} = \frac{\partial E}{\partial h_i} \frac{\partial h_i}{\partial w_{ik}} = \sum_{j=1}^V e_j w'_{ji} \cdot x_k$$

損失関数の微分 (4/4)

行列の形で表すことにすると

$$\frac{\partial E}{\partial \mathbf{W}} = \frac{\partial E}{\partial \mathbf{h}} \mathbf{x}^\top \in \mathbb{R}^{V \times D}$$

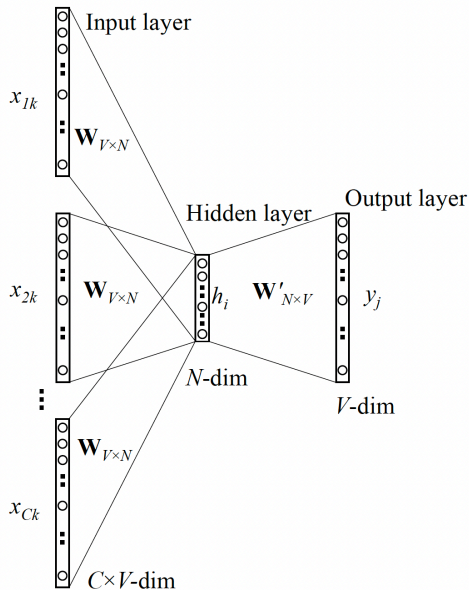
ただし、 $\mathbf{x} \in \mathbb{R}^V$, $\partial E / \partial \mathbf{h} \in \mathbb{R}^D$ 。 \mathbf{x} は一つの成分だけが1であとはゼロ。

前段のパラメータの更新式

$$\mathbf{W}^{(\text{new})} = \mathbf{W}^{(\text{old})} - \eta \frac{\partial E}{\partial \mathbf{W}}$$

$D \times V$ の行列 \mathbf{W} のうち、 \mathbf{x} が非ゼロの成分に対応する列ベクトルだけが更新される。

CBoW のアーキテクチャ (正式版: C 単語入力)



式表現 (正式版: C 単語入力)

1 層目から 2 層目

$$\mathbf{h} = \frac{1}{C} \mathbf{W}(\mathbf{x}_1 + \mathbf{x}_2 + \dots, \mathbf{x}_C)$$

2 層目から 3 層目

$$\begin{aligned} \mathbf{u} &= \mathbf{W}'\mathbf{h} \\ y_j &= \text{SM}_j(\mathbf{u}) = \frac{\exp(u_j)}{\sum_{j'=1}^V \exp(u_{j'})} \end{aligned}$$

y_j は隠された単語が、辞書の j 番目である事後確率の推定値

$$y_j = P(t_j = 1 | \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_C)$$

学習の損失関数

各問題に対する尤度 $L(\mathbf{W}, \mathbf{W}')$

$$L(\mathbf{W}, \mathbf{W}') = \prod_{j=1}^V P(t_j = 1 | \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_C)^{t_j}$$

以下は入力が1単語のときと同じ

$$E(\mathbf{W}, \mathbf{W}') = - \left(u_{j^*} - \log \sum_{j'=1}^V \exp(u_{j'}) \right)$$

損失関数の微分 (3'/4)

中間層の出力 h_i で微分 (中間層から最終層は全結合)

$$\frac{\partial E}{\partial h_i} = \sum_{j=1}^V \frac{\partial E}{\partial u_j} \frac{\partial u_j}{\partial h_i} = \sum_{j=1}^V e_j w'_{ji}$$

中間層は入力の \mathbf{W} による線形変換: $h_i = \mathbf{w}_i^\top (\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_C)/C$

$$\frac{\partial E}{\partial w_{ik}} = \frac{\partial E}{\partial h_i} \frac{\partial h_i}{\partial w_{ik}} = \frac{1}{C} \sum_{j=1}^V e_j w'_{ji} \cdot (x_{1k} + x_{2k} + \cdots + x_{Ck})$$

ただし、 $\mathbf{x}_\ell = [x_{\ell 1}, x_{\ell 2}, \dots, x_{\ell V}]^\top$

損失関数の微分 (4'/4)

行列の形で表すことにすると

$$\frac{\partial E}{\partial \mathbf{W}} = \frac{1}{C} \frac{\partial E}{\partial \mathbf{h}} (\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_C)^\top \in \mathbb{R}^{V \times D}$$

ただし、 $\mathbf{x}_\ell \in \mathbb{R}^V$, $\partial E / \partial \mathbf{h} \in \mathbb{R}^D$ 。 \mathbf{x}_ℓ は一つの成分だけが1であとはゼロ。

前段のパラメータの更新式

$$\mathbf{W}^{(\text{new})} = \mathbf{W}^{(\text{old})} - \eta \frac{\partial E}{\partial \mathbf{W}}$$

$D \times V$ の行列 \mathbf{W} のうち、 \mathbf{x}_ℓ ($\ell = 1, 2, \dots, C$) が非ゼロの成分に対応する列ベクトルだけが更新される。(入力された C 個の単語に対応する \mathbf{W} の列ベクトルそれぞれが、同じ量だけ $(\partial E / \partial \mathbf{h})$ 更新される。)

CBow で得られた単語の埋込

$$h = Wx$$

単語間の類似度

単語 1 と単語 2 の埋め込みベクトルへの変換

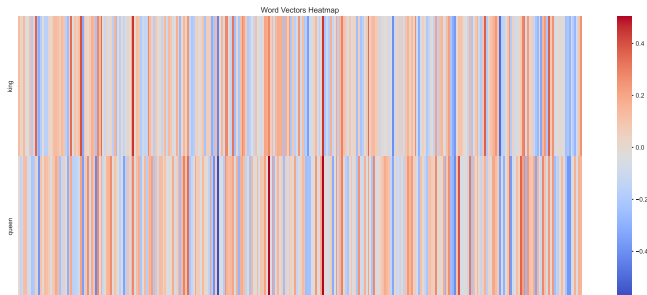
$$\mathbf{h}_1 = \mathbf{W}\mathbf{x}_1, \quad \mathbf{h}_2 = \mathbf{W}\mathbf{x}_2$$

単語 1 と単語 2 のコサイン類似度

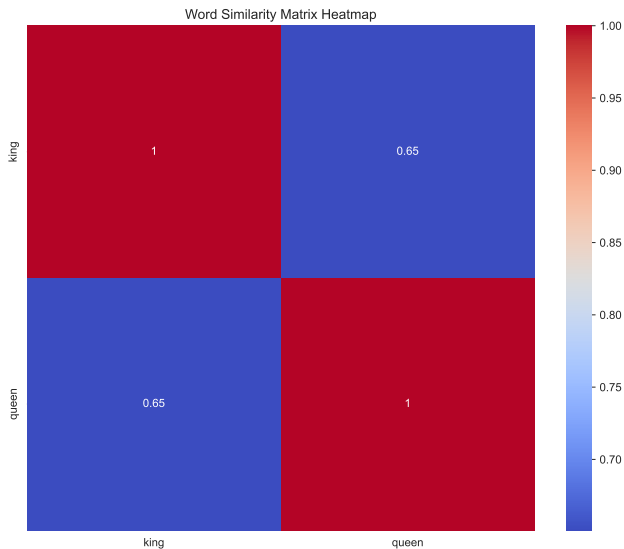
$$S(\text{word}_1, \text{word}_2) = \cos \theta_{12} = \frac{\mathbf{h}_1^\top \mathbf{h}_2}{\|\mathbf{h}_1\| \|\mathbf{h}_2\|}$$

単語の埋込ベクトル (h)

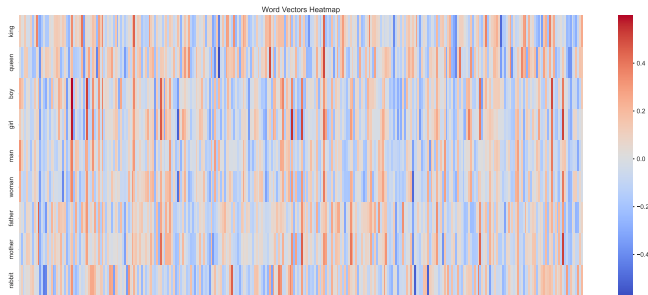
- この例では h の次元は $D = 500$



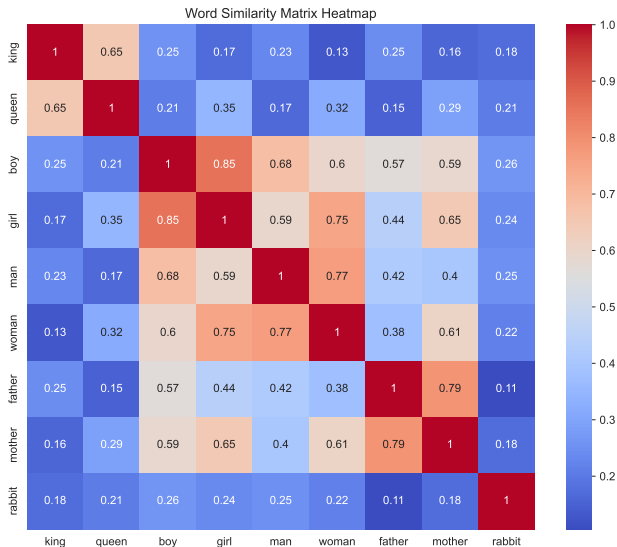
単語間の類似度



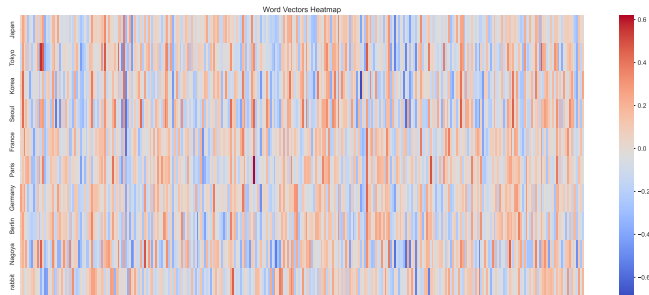
単語の埋込ベクトル (h)



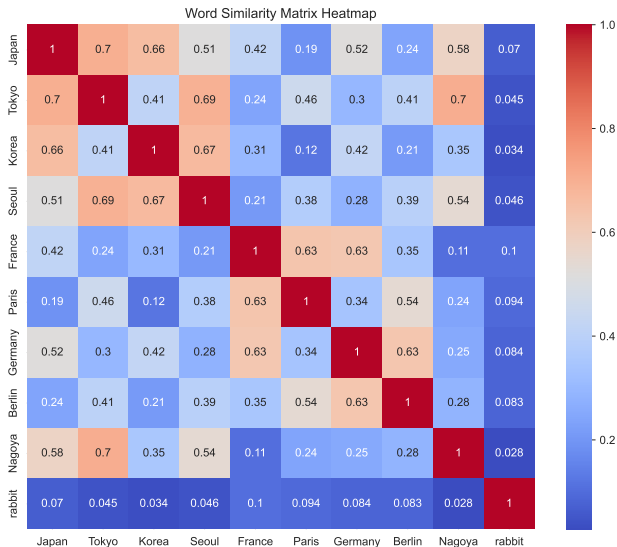
単語間の類似度



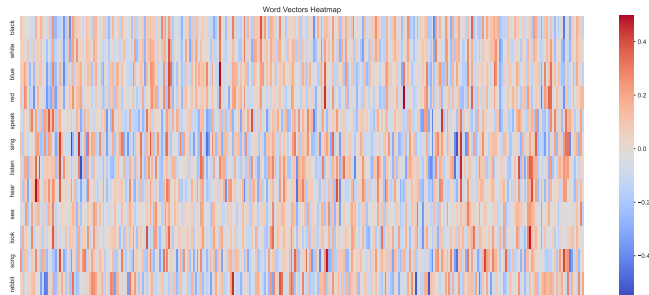
単語の埋込ベクトル (h)



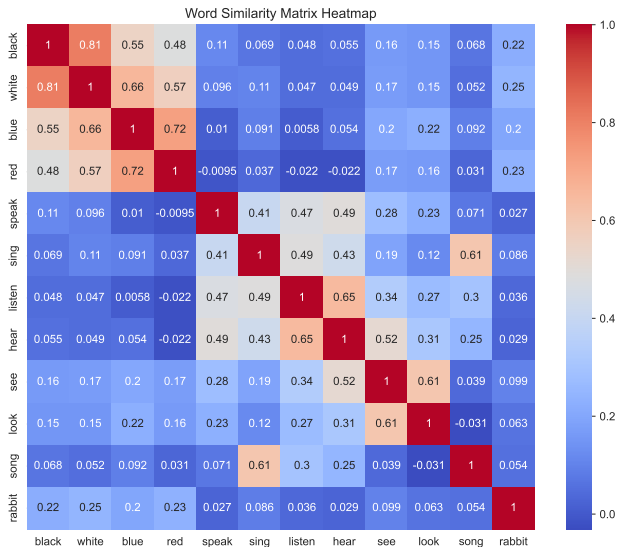
単語間の類似度



単語の埋込ベクトル (h)



単語間の類似度



考察：Continuous Bag of Wordsは何をしたのか

- **コンテキスト（文脈）**：与えられた文章全体
 - CBoW では前後あわせて C 単語 (x_1, x_2, \dots, x_C) がコンテキスト

分布仮説

単語の意味はその単語の周辺に現れる単語、すなわちコンテキストによって決まっている。

- I swam across the river to get to the other **bank**.
- I walked across the road to get cash from the **bank**.

「コンテキスト」もベクトルで表現して、単語の予測に使った。

脱線：ベクトルの表現能力

1 本の 500 次元のベクトルが「コンテキスト」を表現できるのか。

- 500 次元の 2 値ベクトル 1 本で表現できる量

2^{500} 通り

- 参考：地球を構成する原子の総数の概算

8.2×2^{51} 個

考察：Continuous Bag of Wordsは何をしたのか

- CBoW では全単語ベクトルの平均でコンテキストを表現していた
 - Bag of Words: 単語群を入れたカバン (bag)
 - 単語の順序は気にしない

$$\mathbf{h} = \frac{1}{C} \mathbf{W}(\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_C) = \frac{1}{C} (\mathbf{W}\mathbf{x}_1 + \mathbf{W}\mathbf{x}_2 + \cdots + \mathbf{W}\mathbf{x}_C)$$

- 各単語の埋め込みベクトルの平均でコンテキストを表現

$$= \frac{1}{C} (\mathbf{h}_1 + \mathbf{h}_2 + \cdots + \mathbf{h}_C)$$

- 単語の one-hot-vector の平均でコンテキストを表現する場合と比較せよ。

考察：Continuous Bag of Wordsは何をしたのか

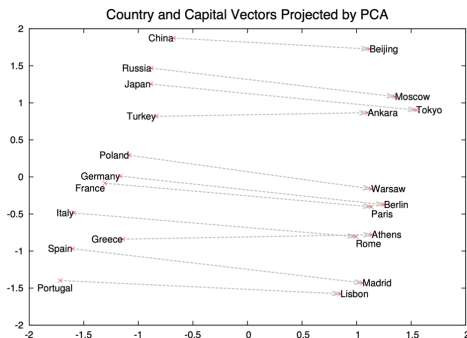
- CBoW はマスクされた単語をコンテキストから予測
 - コンテキストは埋め込みベクトルの和で表現
- 予測は線形多クラス識別

$$y_j = \text{SM}_j(\mathbf{u}) = \frac{\exp((\mathbf{w}'_j)^\top \mathbf{h})}{\sum_{j'=1}^V \exp((\mathbf{w}'_{j'})^\top \mathbf{h})}$$

- コンテキストの埋め込みベクトル \mathbf{h} を用いれば単語予測が線形識別可となるように学習

考察：Continuous Bag of Words で何ができたのか

- 関係の強い単語どうしが同じ方向を向く
- アナロジーがベクトルの加減算で解けるように配置



例：日本における東京はフランスにおける何か？

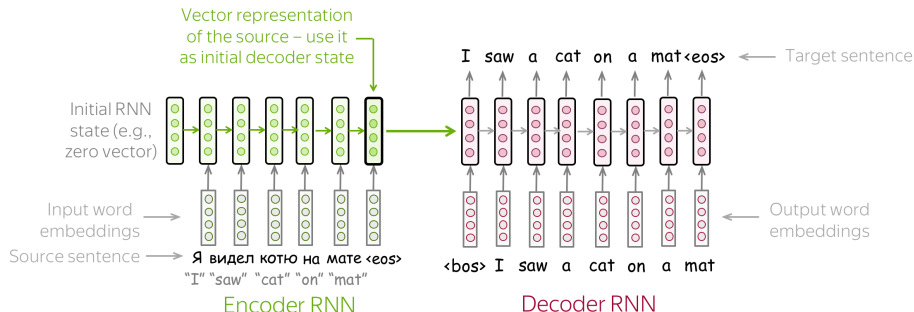
$$h_{\text{Paris}} = h_{\text{Tokyo}} - h_{\text{Japan}} + h_{\text{France}}$$

seq2sec

RNN を使ったコンテキストの表現

Seq2Seq: RNN で系列変換

- 前段の RNN: エンコーダ
 - 入力文を特徴ベクトルに
- 後段 RNN: デコータ
 - 入力文の特徴ベクトルから異なる系列を自己復号



復習：系列変換

例題：自動翻訳 「This is a pen」 → 「これはペンです」

- 入力：日本語の単語系列: $\mathbf{x}_1, \mathbf{x}_2, \dots$
- 出力：英語の単語系列: $\mathbf{y}_1, \mathbf{y}_2, \dots$

翻訳のために $p(\mathbf{y}|\mathbf{x})$ を求めたい。Seq2Seq で実現

$$\hat{\mathbf{y}} = \arg \max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}, \theta)$$

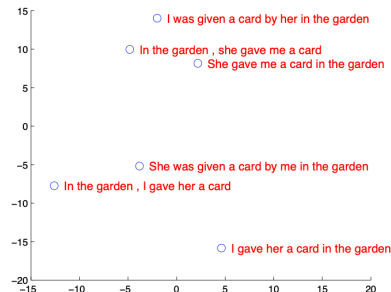
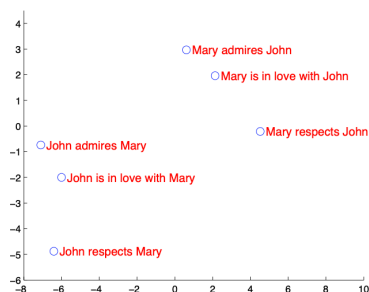
条件付き言語モデル：

$$p(\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T | \mathbf{x}) = \prod_{t=1}^T p(\mathbf{y}_t | \mathbf{y}_{<t}, \mathbf{x})$$

- 交差エントロピーを最小にするように学習
- エンコーダは入力文 \mathbf{x} を1つのベクトル \mathbf{z} へと変換する
- デコーダは \mathbf{z} を入力とし、出力文 \mathbf{y} を出力する

RNN による文章の符号化

隠れ変数 z は入力された文章のベクトルによる表現



<https://arxiv.org/pdf/1409.3215.pdf> (Fig.2)

意味の似た文章は似た方向のベクトルへと変換される

RNNによる文章=コンテキスト符号化の懸念事項

- 文章が長くなると、出力特徴ベクトルに文冒頭の影響が、なかなか及ばない