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 —

前編

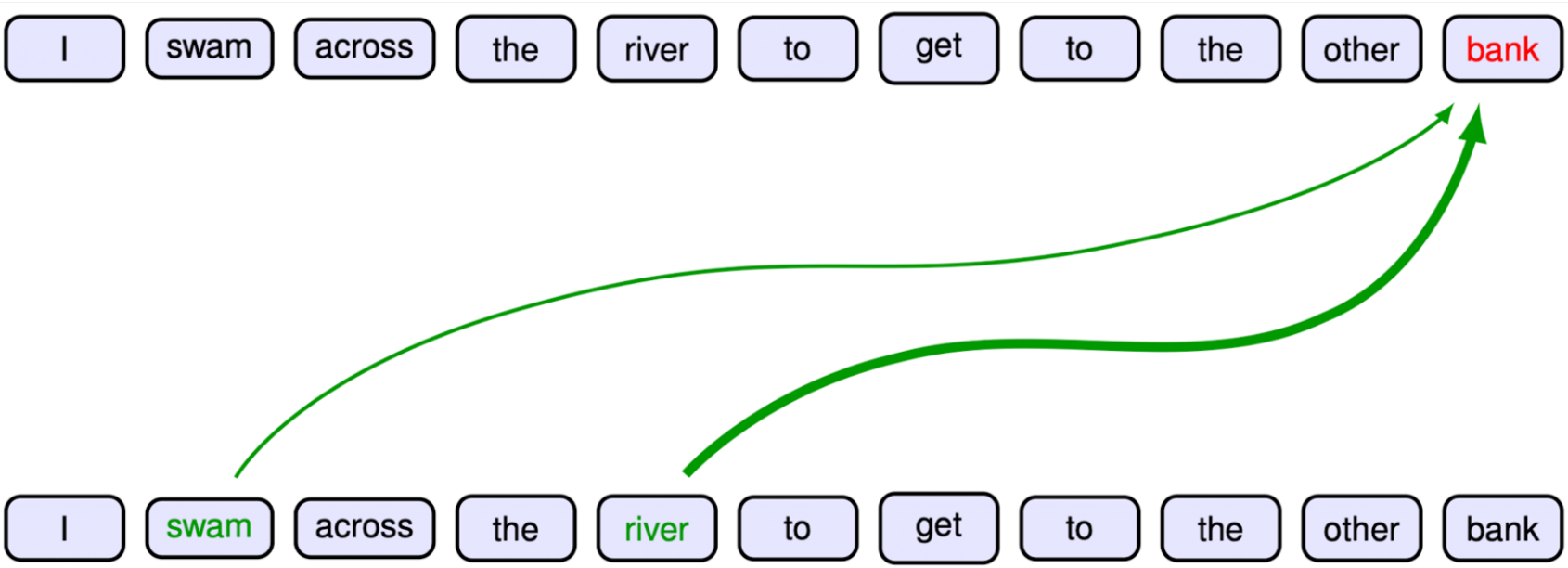
本谷 秀堅

名古屋工業大学

再掲：言語データ

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）

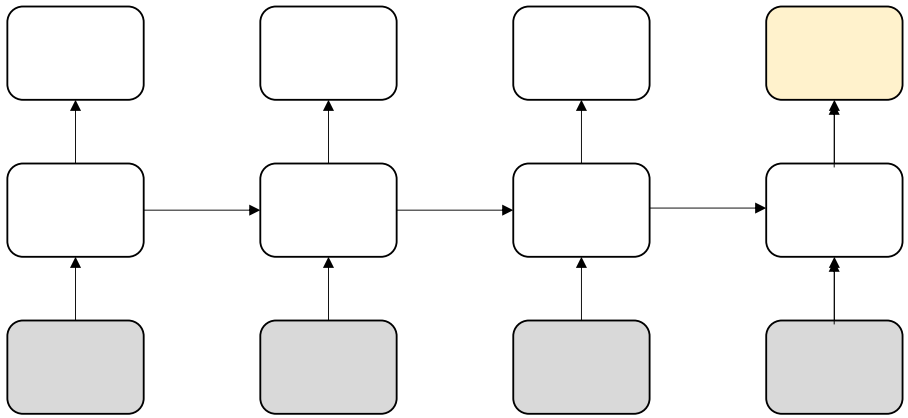


単語を特徴ベクトルへと変換

- 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

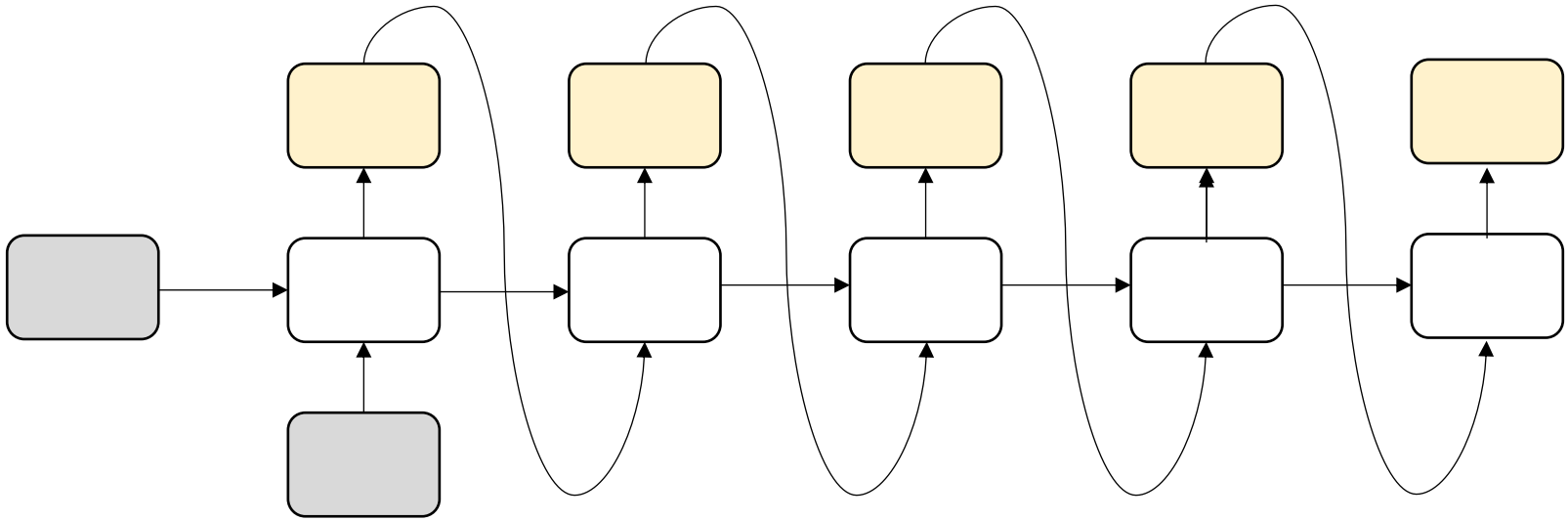
似た概念の単語は似たベクトルのほうが都合が良い



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 $K = 10$

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

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

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

- 「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

単語ベクトルの自己教師学習

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

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 ■ to get

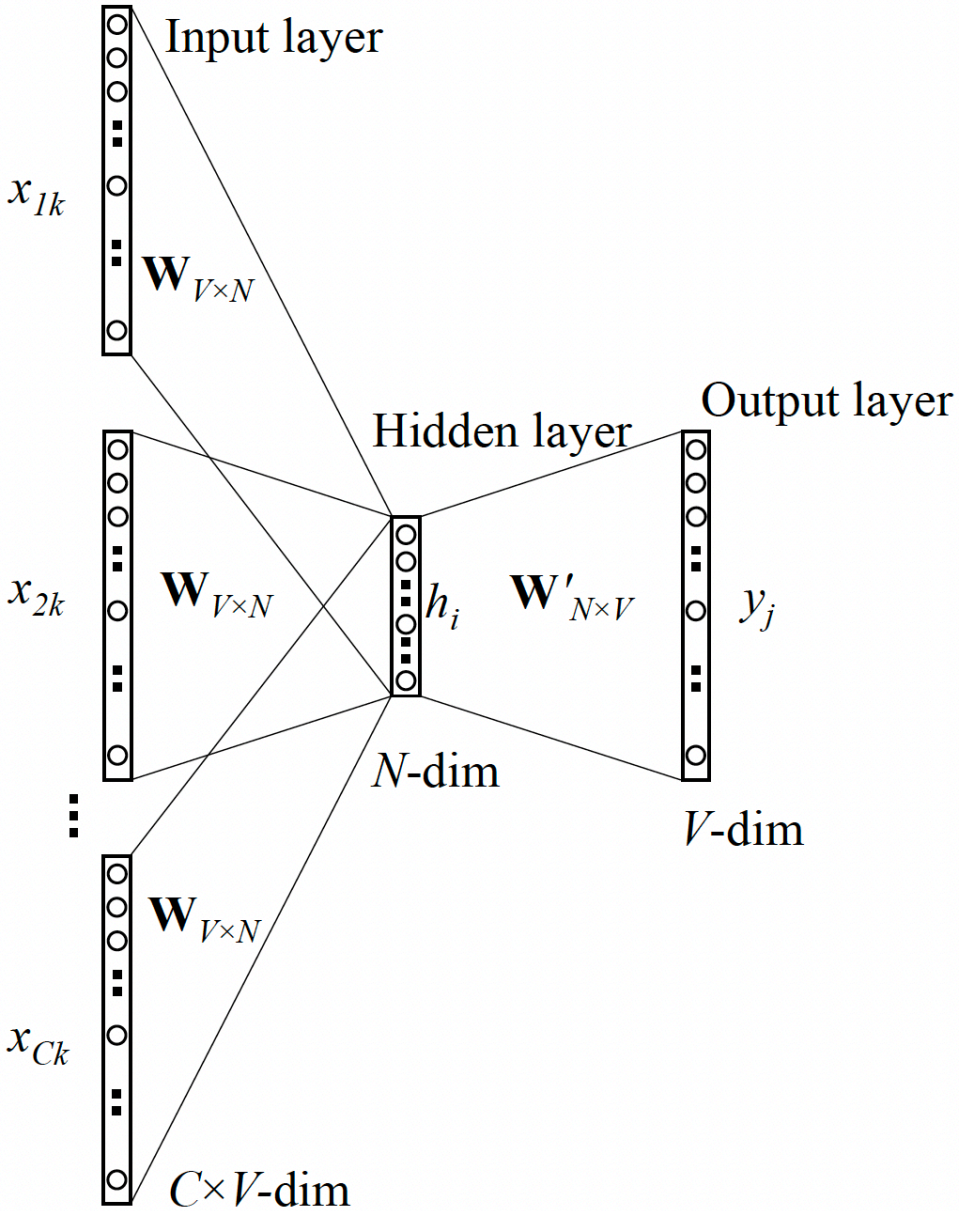
x_1 x_2 t x_3 x_4

学習データ 2

the river ■ get to

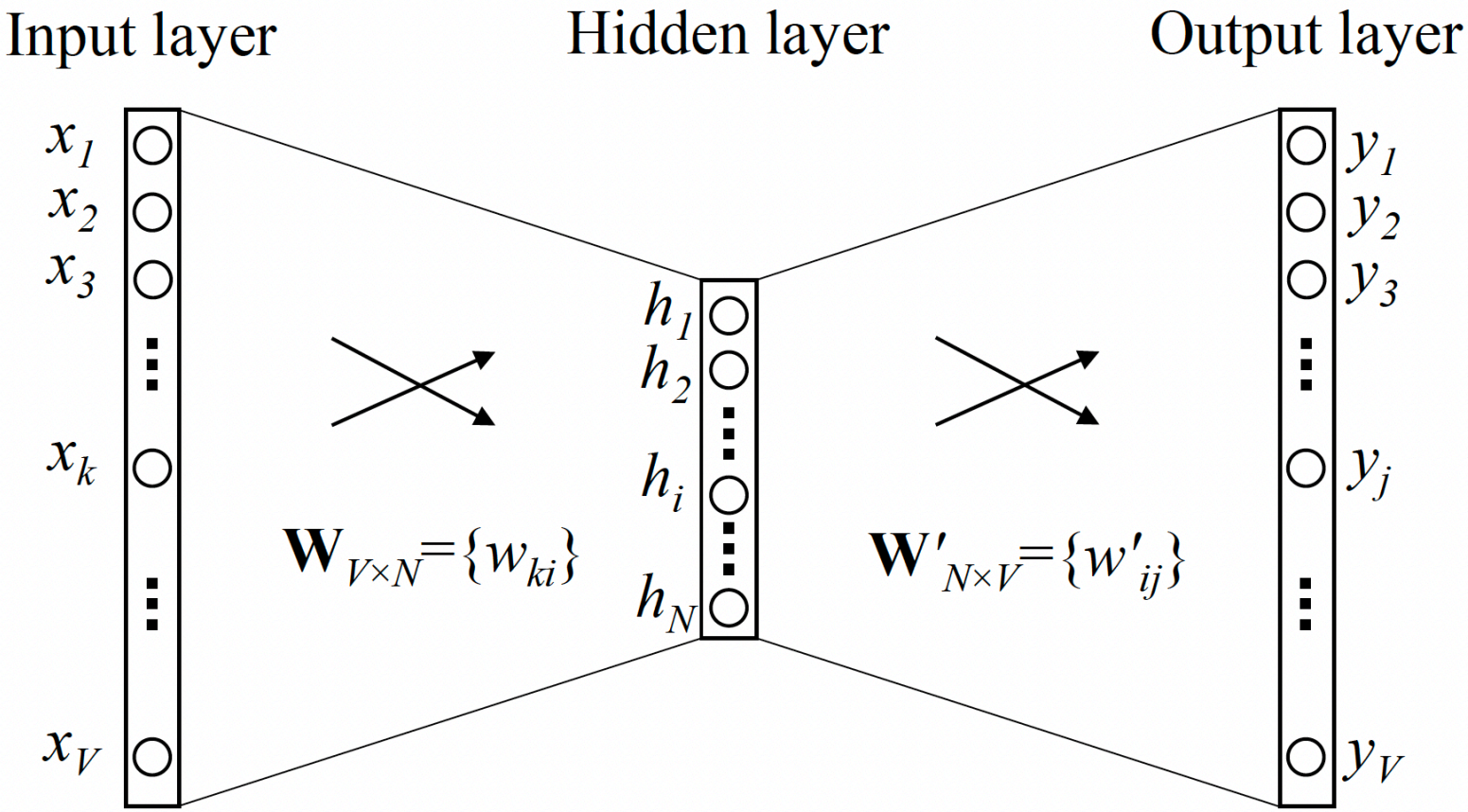
x_1 x_2 t x_3 x_4

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, N: 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

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

2 層目から 3 層目

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

前段の全結合層

結線の重みの行列表現

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

- 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, . . . , N)

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

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

$$\boldsymbol{h} = [h_1, h_2, \ldots, h_N]^\top$$

後段の全結合層

結線の重みの行列表現

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

最終層の j 番目のユニットの内積計算

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

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

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

式表現（正式版: C 単語入力）

1 層目から 2 層目

$$\boldsymbol{h} = \boldsymbol{W}(\boldsymbol{x}_1 + \boldsymbol{x}_2 + \ldots, \boldsymbol{x}_C)$$

2 層目から 3 層目

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

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

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

学習の損失関数

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

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

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）

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