データサイエンス『実践コース』

数理工学PBL

Day 2-2：分散表現

一関高専　小池　敦

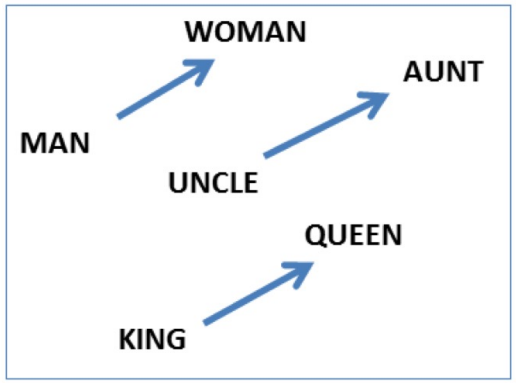
分散表現

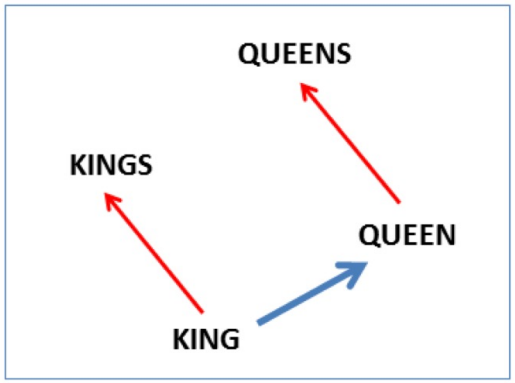
単語の分散表現とは？

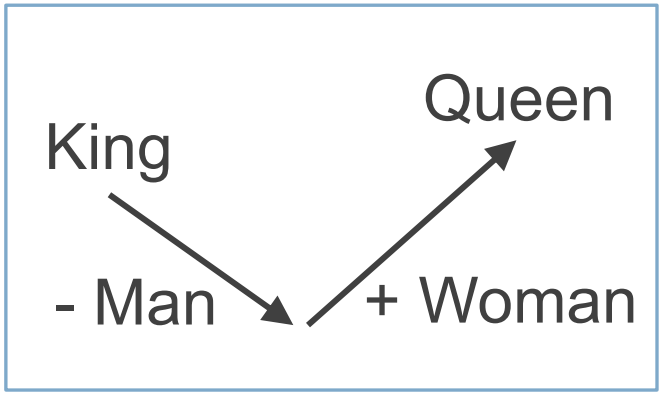
- 単語を低次元の実数値ベクトルで表したもの
  - 数10次元から数100次元
  - One-hotベクトルは語彙サイズ分の次元が必要だった
- 意味が似ている単語が近くに配置される
- 深層学習において標準的に使用されている
  - One-hot ⇒ 分散表現 の変換はword2vecと呼ばれる

分散表現の特徴

- ベクトルの足し算に意味づけができる
  - 概ね以下のような関係が成り立つ







Tomas Mikolov et al. Linguistic Regularities in Continuous Space Word Representations. NAACL-HLT 2013. https://aclanthology.org/N13-1090.pdf

線形結合

線形結合（行ベクトル版）

$$[\alpha_1 \quad \alpha_2 \quad \cdots \quad \alpha_n] \begin{bmatrix} \boldsymbol{v_1} \\ \boldsymbol{v_2} \\ \vdots \\ \boldsymbol{v_n} \end{bmatrix} = \alpha_1 [\ \boldsymbol{v_1}\ ] + \alpha_2 [\ \boldsymbol{v_2}\ ] + \vdots + \alpha_n [\ \boldsymbol{v_n}\ ]$$

$$= \sum_{i=1}^{n} \alpha_i \boldsymbol{v_i}$$

参考：線形結合（列ベクトル版）

$$\begin{bmatrix} \boldsymbol{v_1} & \boldsymbol{v_2} & \cdots & \boldsymbol{v_n} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} = \alpha_1 \begin{bmatrix} \boldsymbol{v_1} \end{bmatrix} + \alpha_2 \begin{bmatrix} \boldsymbol{v_2} \end{bmatrix} + \cdots + \alpha_n \begin{bmatrix} \boldsymbol{v_n} \end{bmatrix}$$

$$= \sum_{i=1}^{n} \alpha_i \boldsymbol{v_i}$$

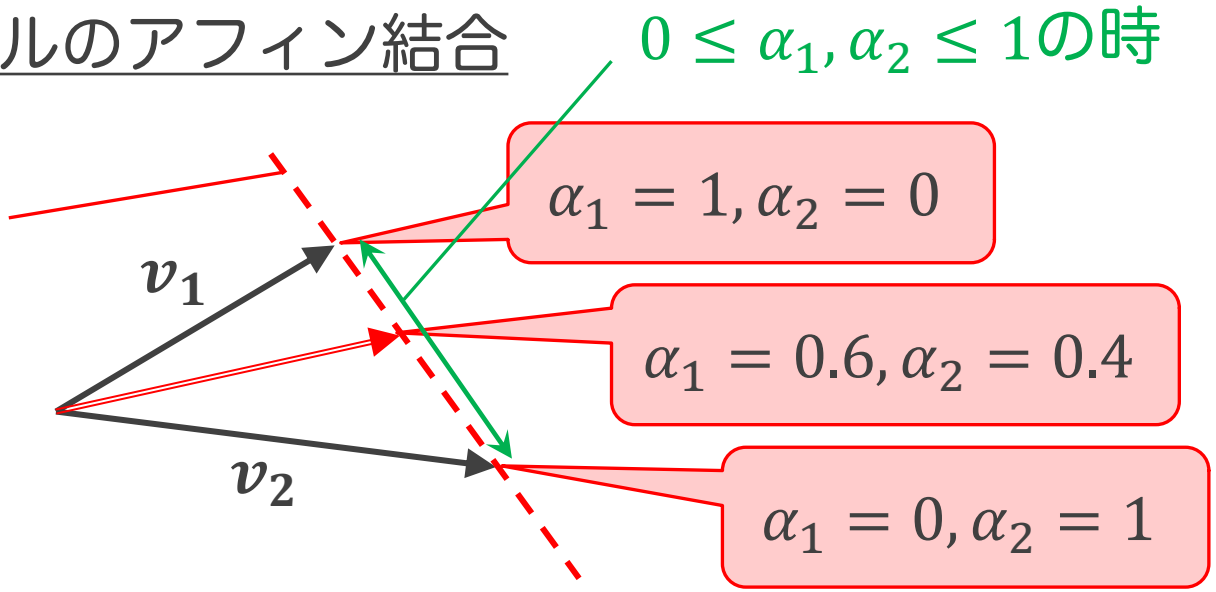
アフィン結合

- 線形結合 $\sum_{i=1}^{n} \alpha_i \boldsymbol{v_i}$ において $\sum_{i=1}^{n} \alpha_i = 1$ の場合

  をアフィン結合と言う

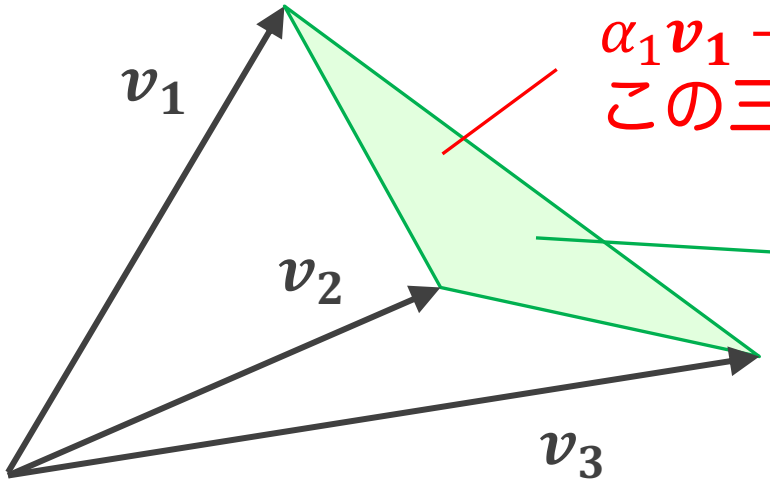
ふたつのベクトルのアフィン結合

アフィン結合
$\alpha_1 \boldsymbol{v_1} + \alpha_2 \boldsymbol{v_2}$は
必ずこの直線上
になる



アフィン結合

3つのベクトルのアフィン結合



アフィン結合

$\alpha_1\boldsymbol{v_1} + \alpha_2\boldsymbol{v_2} + \alpha_3\boldsymbol{v_3}$は
この三角形を含む平面上になる

$0 \leq \alpha_1, \alpha_2, \alpha_3 \leq 1$の時
この領域

アフィン結合 ⇒ ベクトルのブレンド

線形結合 ⇒ アフィン結合

- 線形結合の係数をソフトマックスで変換すると，アフィン結合になる（係数の和が1になる）

線形結合

$$[\alpha_1 \quad \alpha_2 \quad \cdots \quad \alpha_n] \begin{bmatrix} \boldsymbol{v_1} \\ \boldsymbol{v_2} \\ \vdots \\ \boldsymbol{v_n} \end{bmatrix}$$

⇓

アフィン結合

$$\mathrm{softmax}([\alpha_1 \quad \alpha_2 \quad \cdots \quad \alpha_n]) \begin{bmatrix} \boldsymbol{v_1} \\ \boldsymbol{v_2} \\ \vdots \\ \boldsymbol{v_n} \end{bmatrix}$$

One-hotベクトルによるアフィン結合

$$\begin{matrix}\alpha_1 & \cdots & \alpha_k & \cdots & \alpha_n\end{matrix}$$
$$\begin{bmatrix}0 & \cdots & 1 & \cdots & 0\end{bmatrix}$$
$$\begin{bmatrix} \boldsymbol{v_1} \\ \vdots \\ \boldsymbol{v_k} \\ \vdots \\ \boldsymbol{v_n} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v_k} \end{bmatrix}$$

値が1のindexに対応するベクトルが取り出される

分散表現の求め方

分散表現学習の方針

- ニューラルネットワークでOne-hotベクトルから分散表現への変換方法を学習させる
- 分散表現を用いて別の問題を解かせるようなネットワークにする
  - 文中のある単語を隠して周りの単語から予測させる（CBOW）
  - 文中のある単語から，周りの単語を予測させる（skip-gram）

Tomas Mikolov et al. Efficient Estimation of Word Representations in Vector Space. https://arxiv.org/pdf/1301.3781.pdf

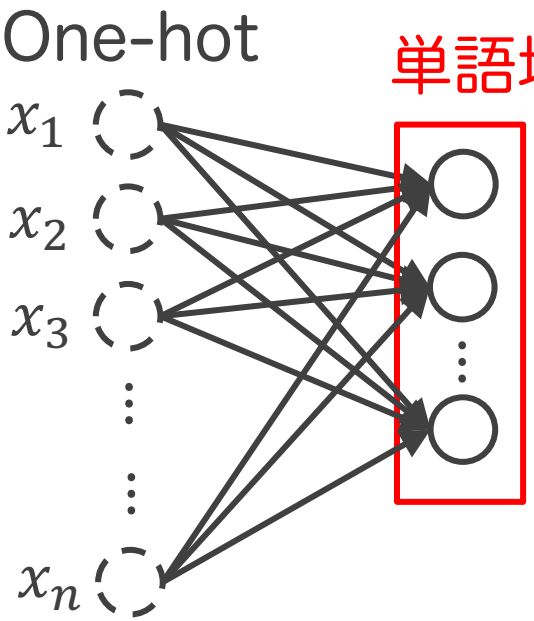
単語埋め込み（embedded）層

- 単語IDに対応するOne-hotベクトルを分散表現に変換する層
- 活性化関数のない全結合層となっている(ただし入力は単語ID)
- 複数の単語IDを入力すると複数の分散表現が得られる

One-hotベクトル

$[x_1 \quad x_2 \quad \cdots \quad x_v]$

n：語彙数



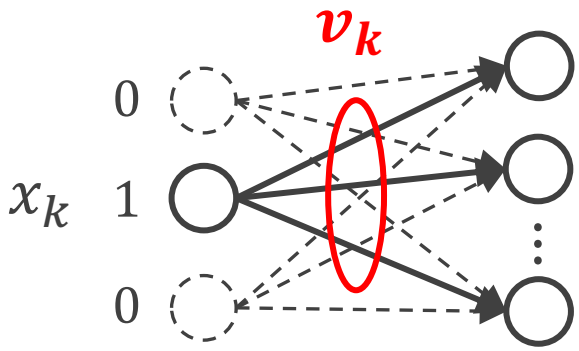
単語 k の分散表現

$[\quad \boldsymbol{v_k} \quad]$

単語埋め込み（embedded）層の行列表現

One-hotベクトル　　辺の重みW　　分散表現

$$\begin{bmatrix} x_1 & \cdots & x_k & \cdots & x_n \\ 0 & & 1 & & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v_1} \\ \vdots \\ \boldsymbol{v_k} \\ \vdots \\ \boldsymbol{v_n} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v_k} \end{bmatrix}$$

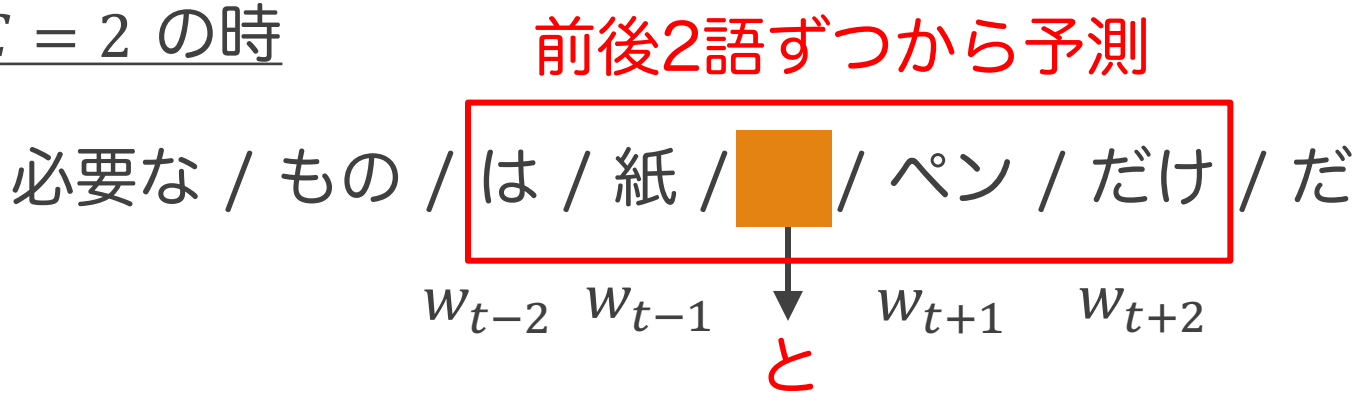


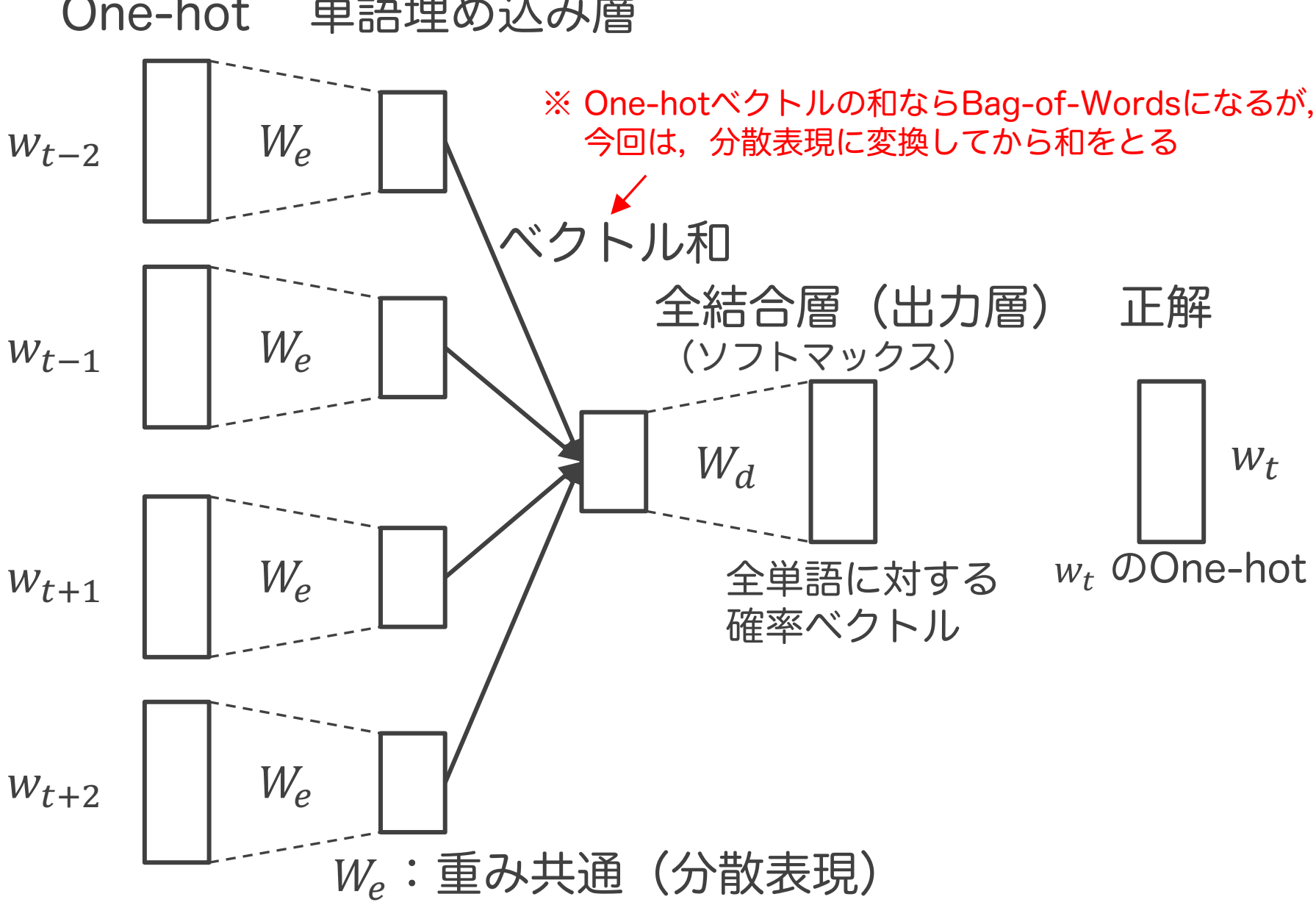
単語 k の分散表現は，
辺の重み行列 W の
k 行目の行ベクトルとなる

CBOW (Continuous Bag-of-Words)

- 文中の単語を前後 C 単語ずつから予測する
  - 自己教師あり学習となっている
    （教師あり学習だがラベルを用意する必要がない）
  - 予測の正解率は100%にならないが問題ない
    （学習の過程で分散表現が良くなれば十分）

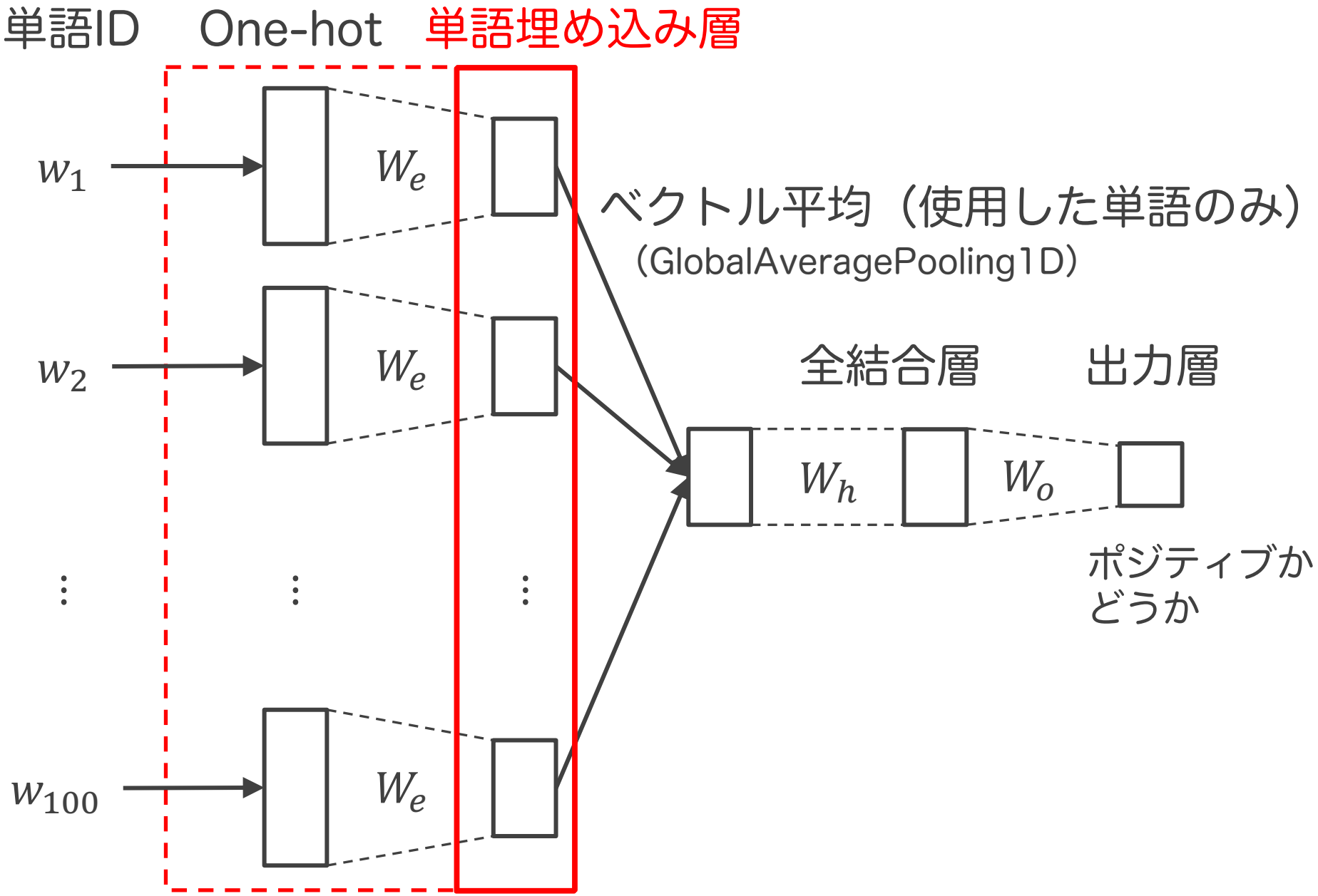
$C = 2$ の時




One-hot　単語埋め込み層
※ One-hotベクトルの和ならBag-of-Wordsになるが，
今回は，分散表現に変換してから和をとる
w_{t-2}
W_e
w_{t-1}
W_e
w_{t+1}
W_e
w_{t+2}
W_e
ベクトル和
全結合層（出力層）
（ソフトマックス）
W_d
全単語に対する
確率ベクトル
正解
w_t
w_t のOne-hot
W_e：重み共通（分散表現）

実習で作成するモデル

- 映画のレビュー文から，内容がポジティブかネガティブかを推定する
  - 入力：文の各単語（最大100単語）
  - 出力：論理値（ポジティブならTrue）
- CBOWの出力を単語でなく論理値に変える
  - CBOWはソフトマックス計算が重く工夫が必要


単語ID
One-hot
単語埋め込み層
w_1
w_2
w_{100}
W_e
W_e
W_e
ベクトル平均（使用した単語のみ）
(GlobalAveragePooling1D)
全結合層
出力層
W_h
W_o
ポジティブか
どうか