

# 3-7 言語・知識

東京大学 数理・情報教育研究センター  
2021年4月1日

# 概要

- 人間の知的活動に関わる人工知能技術の中で言語・知識に関わる技術について理解する.

# 本教材の目次

1. 言語・知識	3
2. コーパス	6
3. 分かち書き・形態素解析	8
4. かな漢字変換	18
5. 構文解析	20
6. 文書単語行列・文書間類似度	23
7. 言語知識	30
8. 言語モデル	35
9. 機械翻訳	40
10. 自然言語生成	45

# 1. 言語・知識

# 言語・知識

- 自然言語処理はコンピュータによる自然言語（英語や日本語）の処理と理解を目的としています.
- 自然言語処理の技術は現在、以下のような様々な領域において活用されています.
  - 情報検索
  - 情報抽出
  - 文書自動分類
  - 文書自動要約
  - 機械翻訳
  - 質問応答システム
  - 対話システム
  - 評判分析
  - ソーシャルメディア分析 など

# 言語・知識

自然言語処理は以下のような処理を含み、これらの処理は近年、大規模なデータと機械学習技術に基づいて行われます。

- 形態素解析
  - 文を単語に分割し、各語に品詞や活用形を付与
    - 人名や地名などの単語の固有表現の認識を伴うこともある
- 構文解析（係り受け構造や句構造の解析）
  - 文の語句間の修飾関係を解析
- 格解析（述語項構造解析）
  - 文の項と述語の関係を解析
- 文脈解析
  - 文間の関係を解析
    - 照応解析（照応詞が参照する先行詞の同定）
      - 文をまたがった語句の関係を解析
    - 談話構造解析
      - 文や節の間の意味的な関係を解析

## 2. コーパス

# コーパス

- 目的に応じて収集された（加えて言語的な解釈の情報が付与された）文書の集合を **コーパス** と呼びます。自然言語処理ではコーパスを活用した解析が行われます。
  - 例：国立国語研究所の日本語コーパス
    - 話し言葉・会話コーパス
      - [https://pj.ninjal.ac.jp/corpus\\_center/](https://pj.ninjal.ac.jp/corpus_center/)
    - ウェブコーパス
      - [https://pj.ninjal.ac.jp/corpus\\_center/nwjc](https://pj.ninjal.ac.jp/corpus_center/nwjc)
  - 例：ブラウンコーパス
    - 新聞，書籍，雑誌などから集めれたコーパス
      - <http://korpus.uib.no/icame/manuals/BROWN/INDEX.HTM>
  - 例：ウェブページのコーパス
    - 7億の英語ウェブページ
      - <https://lemurproject.org/clueweb12/>
- APIを利用してデータを収集してコーパスを構築することもできます。
  - 例：国会議事録検索のAPI
    - <https://kokkai.ndl.go.jp/api.html>



# コーパス

- コーパスに対して言語的な解釈に関する情報が付与されたものを注釈付与コーパス（タグ付きコーパス）と呼びます。
- 言語的な解釈には、単語区切り、品詞、語義、固有表現、構文（句構造、依存構造）、述語項構造、照応・共参照関係、談話構造、などが含まれます。
  - 例：京都大学テキストコーパス
    - 日本語新聞記事の語に品詞情報や構文情報を付与したコーパス
    - <https://github.com/ku-nlp/KyotoCorpus>
  - 例：Penn Treebank
    - 英語新聞記事の語に品詞情報や構文情報を付与したコーパス
    - <https://catalog.ldc.upenn.edu/LDC99T42>
- 大規模な注釈付与コーパスは統計や機械学習のアプローチに基づく自然言語処理において広く活用されています。

### 3. 分かち書き・形態素解析

# 分かち書き（単語分割）

- テキストを「トークン」と呼ばれる表現要素の最小単位の集合に分割することを分かち書き（単語分割）と呼びます。単語に限らずテキストを構成する記号や数字などもトークンとなりえます。
- 英語の場合、テキストをスペースやカンマで区切れば単純な分かち書きをすることができます。

‘Beware the ides of March’

→

[‘ Beware ’, ‘ the ’, ‘ ides ’, ‘ of ’, ‘ March ’]

# 分かち書き（単語分割）

- 日本語の場合は句読点以外はスペースやカンマのような明確な区切りはテキストにないため、何らかの処理によりテキストを分割する必要があります。
- 国会図書館が作成する書誌データの標目の読みに用いるための分かち書きの基準として以下のように示されています。

“分かち書きは、検索語となる自立語を対象とする。日本語の場合は、日本語として不自然でない意味のまとまりで分かち書きを行う。外来語の場合は、当該言語の単語分割により分かち書きを行う。”

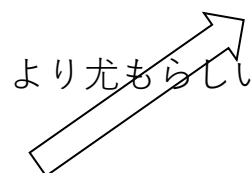
[https://www.ndl.go.jp/jp/data/catstandards/yomi/word\\_division\\_Apr2008.html](https://www.ndl.go.jp/jp/data/catstandards/yomi/word_division_Apr2008.html)

# 形態素

- 言語学では意味を持つ表現要素の最小単位は形態素と呼ばれます。より直感的には形態素は、名詞、動詞、形容詞、副詞、前置詞の品詞や語形・活用形などの文法的役割を表す語のクラスを表します。
- 語は1つ以上の形態素から構成されます。
- 形態素を表すタグを品詞（POS: Part of Speech）タグと呼びます。
  - 例えば、名詞であれば単数名詞はNN、複数名詞はNNS、固有名詞はNPというタグでそれぞれ表されます。
  - その他、VB:動詞現在形、JJ:形容詞、RB:副詞、など
    - Penn Treebankに付加されたPOSタグの例
      - [https://www.ling.upenn.edu/courses/Fall\\_2003/ling001/penn\\_treebank\\_pos.html](https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html)

# 形態素解析

- テキスト（単位として文）をトークンに分割し，各トークンに品詞や語形・活用形などの情報を付与する処理を形態素解析と呼びます．
- 形態素解析では与えられたトークンの系列に対して尤もらしい形態素タグの系列を予測するという系列ラベリング問題を解きます．

より尤もらしい  名詞-動詞-前置詞-冠詞-名詞  
(光陰矢の如し)

「Time flies like an arrow」

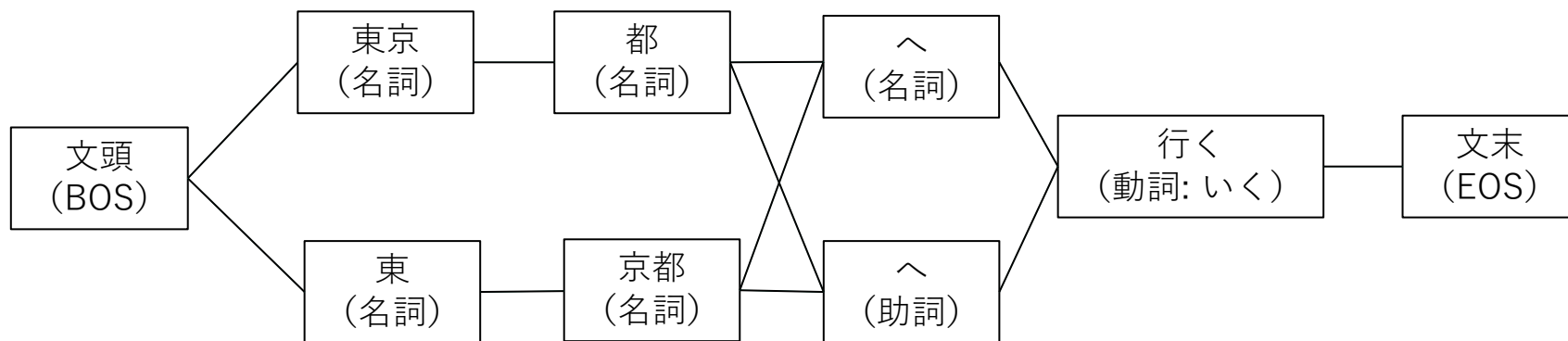
動詞-名詞-前置詞-冠詞-名詞  
(矢のようにハエを測る)

名詞-名詞-動詞-冠詞-名詞  
(時ハエは矢を好む)

# 日本語の形態素解析

- 日本語の形態素解析では辞書を参照しながら文から単語を取り出し、それらの接続の可能性を確認していくため以下の辞書を用います。
  - 単語辞書：単語の品詞，読みや活用形を定義
  - 接続可能性辞書：接続可能な2つの単語または品詞・活用のタイプを定義
- 辞書を元にして文の形態素解析の候補となるラティス構造を作ります。
  - 単語辞書にマッチする文字列に対応するノードを作る
  - 接続可能辞書を元に対象文字列の前後に接続しうる文字列のリンクを作る

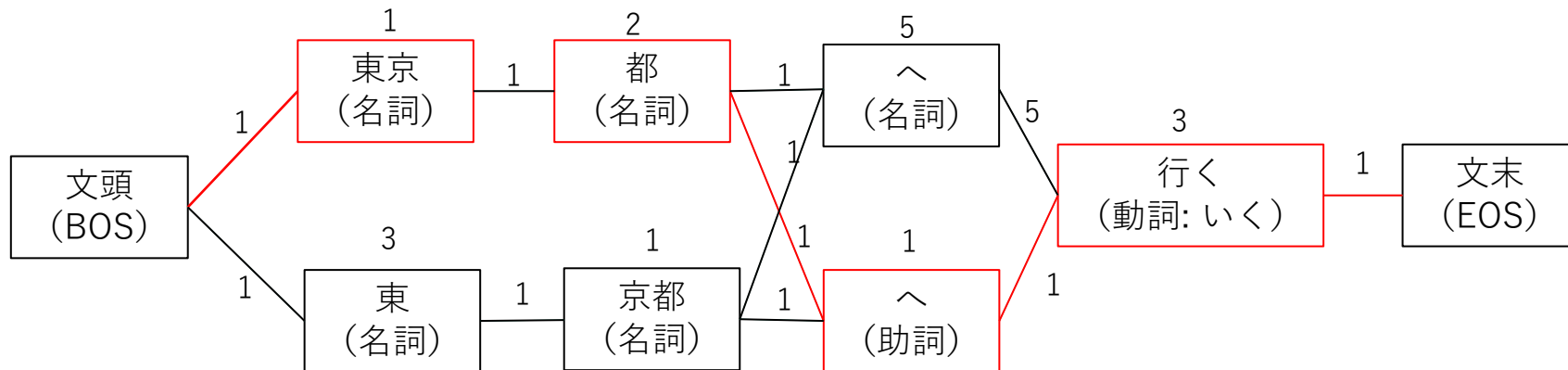
「東京都へ行く」のラティス構造



# 日本語の形態素解析

- 文頭から文末に至るリンクで接続されたノードの系列（パス）は語の並びに対応しており，文の解釈を表します。
- 形態素解析では，ノードとリンクに与えられたコストを元に，それらのコストの和が最小となるようなパス（最適パス）を求めます。
  - コストは一般に大規模なコーパスを元にした学習により与えます
- 一般にパスの組み合わせは膨大となるためビタビアルゴリズムのような動的プログラミングを元にして効率的に最適パスを求めます。

コストが最小（ $1+1+1+2+1+1+1+3+1$ ）のパス：「東京 | 都 | へ | 行く」





# 補：機械学習と言語処理

- 形態素解析の系列ラベリング問題における系列のような構造を学習することを構造学習と呼びます。構造には木やグラフなども考えられます。
  - 確率モデルを用いた構造学習では、観測 $x$ が与えられた時の構造 $y$ の条件付き確率 $p(y|x)$ が最大となる構造を求める問題を解きます
  - 生成モデルを用いる場合は、観測 $x$ と予測する構造 $y$ の同時確率 $p(x,y)$ が最大となる構造を求める問題を解きます
- 系列ラベリング問題では、隠れマルコフモデル(HMM: Hidden Markov Model)，条件付き確率場(CRF: Conditional Random Field)，などの機械学習手法が用いられます。

# 日本語の形態素解析

形態素解析の例：

入力：「テキストもデータであることを理解する。」

形態素解析の結果：

テキスト	名詞,一般,*,*,*,テキスト,テキスト,テキスト
も	助詞,係助詞,*,*,*,も,モ,モ
データ	名詞,一般,*,*,*,データ,データ,データ
で	助動詞,*,*,*,特殊・ダ,連用形,だ,デ,デ
ある	助動詞,*,*,*,五段・ラ行アル,基本形,ある,アル,アル
こと	名詞,非自立,一般,*,*,*,こと,コト,コト
を	助詞,格助詞,一般,*,*,*,を,ヲ,ヲ
理解	名詞,サ変接続,*,*,*,理解,リカイ,リカイ
する	動詞,自立,*,*,サ変・スル,基本形,する,スル,スル
.	記号,句点,*,*,*,.,.,.

例えば、形態素解析の結果から

「テキスト」，「データ」，「こと」，「理解」  
を抽出して、文の名詞を対象に解析することができます。

# 未知語とユーザ定義辞書

- 形態素解析の単語辞書に登録されていない単語を未定義（または未知）語と呼びます。未定義語は多くの場合、固有名詞（人名，地名，組織名など），専門用語，新語や造語です。
- 単語辞書に新たに単語を追加する，または独自にユーザ定義辞書を作成する場合は，以下のように単語の情報を定義します。

見出し語	読み	品詞	活用型	活用形	基本形
人工知能	じんこうちのう	名詞	-	-	-
走る	はしる	動詞	五段・ラ行	終止形	走る

- 未知語をすべて手動で単語辞書に追加していくのは現実的でないため，例えばウェブから収集した情報を元に辞書を自動的に拡充することも行われます。

# 日本語形態素解析のツール

- MeCab
  - <https://taku910.github.io/mecab/>
- Chasen
  - <https://chasen-legacy.osdn.jp/>
- JUMAN
  - <https://nlp.ist.i.kyoto-u.ac.jp/?JUMAN>

## 4. かな漢字変換

# かな漢字変換

- **かな漢字変換**では、入力された「かな」のべた書きテキストを解析して、単語の区切りを見つけ出し、必要な部分を漢字に変換します。
- かな漢字変換では単語の区切りを見つけるのに形態素解析の考え方を応用することができます。
- 一方、かなから漢字へ変換する際はテキストの意味や文脈に応じて漢字を選択する必要があります。
  - 単純な漢字変換はあらかじめ決められた規則を元に漢字候補の優先度を計算することで行います。
  - 単語の使用頻度、単語間の共起関係などの情報を元に、よりテキストの意味や文脈を考慮した漢字変換を行うこともできます。

# かな漢字変換

- かな漢字変換を行う日本語入力システムとしてIME（インプットメソッドエディタ）があります。最近では入力を予測しながら支援する入力予測機能を利用することもできます。

IMEによるかな漢字変換 いって

行って  
言って  
言っていた  
行っていた  
言っている

Tabキーで予測された単語を選択

入力予測機能

🔍 マスク

🔍 マスク

🔍 マスク販売在庫あり

🔍 マスク手作り

## 5. 構文解析

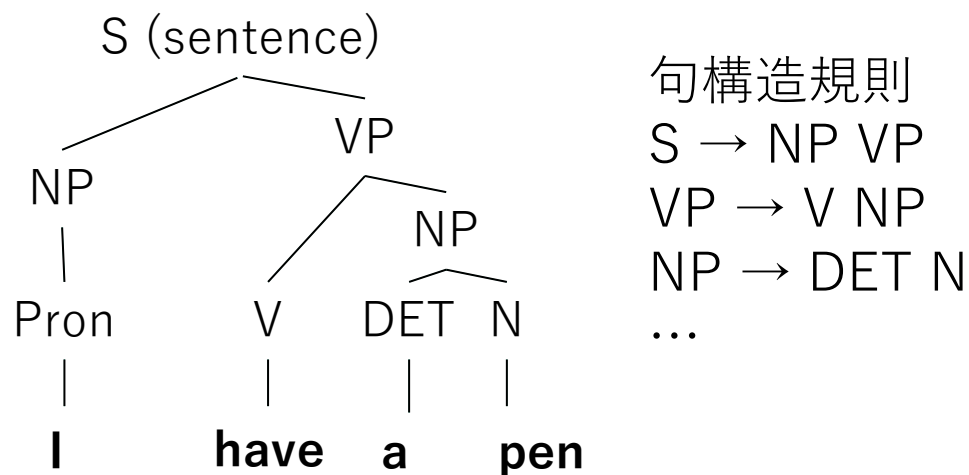


# 構文解析

- 文の構造を構文と呼びます。構文は一般に木構造によって表されます。
- 文節の係り受けは基本的な構文となります。
- 係り受け構造は一般には依存構造と呼ばれ、依存構造を表す木を依存構造木と呼びます。
  - 依存構造木は語や文節をノード、それらの間の依存関係をリンク、として表した木です。
- 複数の語・文節をまとめて句として、また、複数の句をまとめて新たな句とすることで文の構造を表したものを句構造と呼びます。
  - 句構造を表す木では、語は木の葉に対応し、葉と根の中間のノードは品詞や句の種類に対応します。
  - 句構造を表す木のノードの親子関係やノードの分岐は文法（文の構造を規定するもの）の規則を表しています。

# 構文解析

- 句構造のような文の構造を（複数の構文の可能性の中から）決めることを構文解析と呼びます。



- 構文解析では文法を満たすような文の構造を求めます。
  - 例えば、チョムスキー標準形の文脈自由文法に基づく句構造規則を考えるとCKY法により構文解析を行うことができます。

# 構文解析

- CKY法では、文の構造に曖昧性（複数の構文の可能性）がある場合、文法を満たす構文の候補が求められます。
- 構文の候補の中から妥当な構文を選ぶ際には、例えば修飾の表現など語や句の結びつきを手がかりとして実際の言語の使われ方に即した構文を選択することが考えられます。
- 例えば、大規模な言語データを元に、機械学習を用いて主辞（係り受け構造における係り先）と修飾語（係り受け構造における係り元）の依存関係のスコアを計算することで、構文の候補の中から妥当な構文を選択することができます。
- 日本語の構文・係り受け解析器
  - KNP: <https://nlp.ist.i.kyoto-u.ac.jp/index.php?KNP>
  - Cabocha: <https://taku910.github.io/cabocha/>

## 6. 文書単語行列・文書間類似度

# 文書単語行列

- 形態素解析などの処理によりテキストを分割すると，以下のように複数のテキスト（文書）を表（テーブル）の形で表すことができます．この表を文書単語行列と呼び，行はテキスト，列は単語に対応します．
- 文書単語行列の値は，あるテキスト（行）にある単語（列）が現れる（1）・現れない（0）の2値，またはテキスト中の単語の頻度（出現頻度）で表します．

	テキスト	AI	解析	データ
テキストを解析するAI	1	1	1	0
テキストデータを解析する	1	0	1	1
テキストもデータである	1	0	0	1

\*各テキストの名詞のみを抽出

# 文書単語行列

- 文書単語行列の列に対応する単語は以下に留意しながら，なるべくテキストで重要と考えられる単語を用います.
  - 助詞や助動詞などの機能語を用いるか
  - 高頻度で出現する一般的な単語を用いるか
  - 極端に出現頻度が低い単語を用いるか
- 文書単語行列の値は，出現頻度だけでなくテキストにおけるその単語の重要度を数値化したもので表すこともあります.
  - 代表的な重要度としてtf-idfがあります.
  - tf-idfは単語の出現頻度 (tf) にその単語を含むテキスト数の逆数 (idf) を掛け合わせた値です. idfは具体的には以下で定義されます.
    - $\text{idf} = \log_2(\text{すべてのテキスト数} / \text{単語を含むテキスト数}) + 1$   
\*値が大きくなりすぎないようにlogをとります

# 文書単語行列

- 例：以下の3つの文が与えられた時のtf-idfの例
  - 「テキストを解析するAI」の単語「AI」のtf-idf値は、tfが1, idfが $\log_2(3/1)+1$ のため、 $1 \times (\log_2(3/1)+1)=2.585$
  - 「テキストもデータである」の単語「データ」のtf-idf値は、tfが1, idfが $\log_2(3/2)+1$ のため、 $1 \times (\log_2(3/2)+1)=1.585$

	テキスト	AI	解析	データ
テキストを解析するAI	1	2.585	1.585	0
テキストデータを解析する	1	0	1.585	1.585
テキストもデータである	1	0	0	1.585

# 文書ベクトル

- 文書単語行列の行は，その行に対応する文書をその文書における各単語の重要度を要素とするベクトル（文書ベクトル）で表したものとみることができます。
  - 例：テキストにおける各単語の出現頻度を重要度とすると，「テキスト」，「AI」，「解析」，「データ」の単語のベクトル空間において，各文のベクトルは以下のようになります。

	テキスト	AI	解析	データ
テキストを解析するAI	1	1	1	0
テキストデータを解析する	1	0	1	1
テキストもデータである	1	0	0	1

- テキスト1:「テキストを解析するAI」 →  $d_1 = (1,1,1,0)$
- テキスト2:「テキストデータを解析する」 →  $d_2 = (1,0,1,1)$
- テキスト3:「テキストもデータである」 →  $d_3 = (1,0,0,1)$



# 文書間類似度

- 文書間の類似度は、各文書を表すベクトル間の類似度として計算することができます。
  - ベクトル間の類似尺度として、**コサイン類似度**を用いることができます。
  - コサイン類似度は2つのベクトルの内積をそれぞれの長さで割ったものです。
  - コサイン類似度が高いほどそれらの文書は関連度が高いと考えられます。

ベクトル

$$x = (x_1, x_2, \dots, x_n) \quad y = (y_1, y_2, \dots, y_n)$$

内積

$$x \cdot y = x_1 y_1 + x_2 y_2 + \dots + x_n y_n$$

ベクトルの大きさ

$$\|x\|_2 = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$

コサイン類似度

$$\cos\theta = \frac{x \cdot y}{\|x\|_2 \|y\|_2} \quad (\theta \text{ は2つのベクトルのなす角})$$

# 文書間類似度

- 文書間類似度の例

- テキスト1: 「テキストを解析するAI」  $\rightarrow d_1 = (1, 1, 1, 0)$
- テキスト2: 「テキストデータを解析する」  $\rightarrow d_2 = (1, 0, 1, 1)$
- テキスト3: 「テキストもデータである」  $\rightarrow d_3 = (1, 0, 0, 1)$

- テキスト1とテキスト2のコサイン類似度

$$\frac{d_1 \cdot d_2}{\|d_1\| \|d_2\|} = (1 \times 1 + 1 \times 0 + 1 \times 1 + 0 \times 1) / (\sqrt{1^2 + 1^2 + 1^2} \times \sqrt{1^2 + 1^2 + 1^2}) = \frac{2}{3} = 0.666...$$

- テキスト1とテキスト3のコサイン類似度

$$\frac{d_1 \cdot d_3}{\|d_1\| \|d_3\|} = (1 \times 1 + 1 \times 0 + 1 \times 0 + 0 \times 1) / (\sqrt{1^2 + 1^2 + 1^2} \times \sqrt{1^2 + 1^2}) = \frac{1}{\sqrt{6}} = 0.408...$$

- 「テキスト」, 「AI」, 「解析」, 「データ」の単語のベクトル空間においてはテキスト1はテキスト3よりもテキスト2と類似していると言えます。

# 補：語の類似度

- 語の類似度は、同じような文脈に現れる語同士は類似している，という分布仮説の考えに基づいて求められます。
- 語が現れる文脈はその語と関連する（よく共起するような）語で表すことができます。
  - 関連語は相互情報量などの尺度を元に求めることができます。
  - 単語とそれらの関連語を行列で表したものを単語文脈行列と呼びます。
- 分布仮説に従えばお互いの関連語（単語文脈行列の行同士）が類似していればそれらの語は類似しているといえます。
  - $\cos$ 類似度やJaccard係数，Simpson係数，Dice係数などの類似尺度を元に語の類似度を求めることができます。
- 単語文脈行列を行列分解して獲得した語のより低次元表現を元に語の類似度を求めることも行われます。

## 7. 言語知識

# 言語知識

- 言語データから言語処理に利用可能な知識を獲得することを言語知識獲得と呼びます。またそれらの知識は知識表現で記述されます。
- 言語知識には、語彙、文法、語の意味（語が表現する概念）の関係、同義・類義性、多義性、格フレーム、文間の含意関係、文間の因果関係、文の極性、常識的知識 などがあります。
- 近年は大規模な言語データからこれらの言語知識を自動で獲得し、またそれらを言語処理に活用することも行われています。
- 特に語や概念の関係を整理した代表的な言語知識として以下が挙げられます。
  - シソーラス・語彙的オントロジー
  - 意味ネットワーク
  - 知識グラフ

# シソーラス

- 語（語によって表現される概念）の関係を体系的に整理した辞書をシソーラスと呼びます。
  - 上位・下位関係，類義・反義関係，全体・部分関係 など
- 概念のさまざまな関係の記述も含めて体系的に整理したものを特に語彙的（言語的）オントロジーと呼びます。
- 代表的なシソーラスとしてWordNetがあります。
  - WordNetではsynsetと呼ばれる同義語の集合を基本単位として，そのsynsetと上位・下位関係，全体・部分関係がある他のsynsetが記述されています。
    - 英語：<https://wordnet.princeton.edu/>
    - 日本語：<http://compling.hss.ntu.edu.sg/wnja>
      - 日本語のシソーラスとして，この他に，EDR概念辞書，日本語語彙体系などがあります。

# 意味ネットワーク

- 知識をノード（概念やそのインスタンス，実体）とリンク（概念間の関係）からなるネットワーク（有向グラフ）で表したものを意味ネットワークと呼びます.
  - 例えば，概念が持つ属性や概念間の関係として上位・下位関係，全体・部分関係などが定義されます.
- コンピュータが処理可能なように知識を表現する意味ネットワークは対象世界の知識の表現，文の意味の表現，連想関係の表現，などに利用されています.
  - 意味ネットワークを文の意味表現に用いることで，例えば「誰が」（主格），「いつ」（時間格），「どこで」（場所格），「何を」（目的格），といった格の関係を動詞を中心として表現することができます.
  - このような意味ネットワーク（格フレーム）は言語の処理に活用することができます.

# 知識グラフ

- 実体（エンティティ）とそれらの関係を、＜主語，述語，目的語＞の3つ組（トリプル）を用いて表現したものを知識グラフ（ナレッジグラフ）と呼びます。
  - 知識グラフは意味ネットワークに制約を付加した知識表現として元々は提案されていましたが，現在は構造化された大規模な知識ベースを指すことが多いです。
- 現在の知識グラフを支える技術として，情報をトリプルとして記述するための枠組みであるRDF（resource description framework）やそのような構造化された情報を公開・活用する仕組みであるLOD（Linked Open Data）があります。
- 知識グラフの例としてWikipediaの情報を構造化しLODにしたDBPediaやWikipediaのように共同編集可能なWikidataなどがあります。
  - DBPedia: <http://ja.dbpedia.org/>
- 知識グラフは検索エンジンや質問応答などで活用されています。



# 知識グラフ

## Wikidataの「natural language processing」のページ

Wikidataの「natural language processing」のページ

“subclass of”という関係にある概念

Statements

subclass of		
artificial intelligence	1 reference	edit
computer science	0 references	edit
computational linguistics	0 references	edit
industry	0 references	edit
academic discipline	0 references	edit

<https://www.wikidata.org/wiki/Q30642>

## 8. 言語モデル

# 言語モデル

- コーパスにおける文や表現の出現確率（文や表現が使われる確からしさ）を与える統計的なモデルを言語モデルと呼びます。
- 特に確率的な言語モデルではコーパスを元に言語モデルを学習し，文字あるいは単語列に対して確率を与えることでその言語らしさを推定することができます。
  - 単語列の場合は文や文書（文字列の場合は単語）について，対象言語においてより自然なものに高い確率を与えます。
- 単純な確率的言語モデルとしてn-gram言語モデルがあります。
  - 単語n-gram言語モデルでは単語列の各単語は直前のn-1語に依存して現れるとして，コーパスにおける頻度情報を元に単語の生起確率を計算します。
  - n-gram言語モデルは文や表現の出現確率（文や表現が使われる確からしさ）の計算に使われます。

# 言語モデル

- 例えば、単語n-gram言語モデルでは、1単語（ユニグラム）、2単語（バイグラム）、3単語（トライグラム）、といった連続するn単語を単位としてコーパス内のそれらの頻度を数えます。

文「データを解析する」のバイグラムによる出現確率の計算の例

$$P(\text{データを解析する}) = P(\text{データ}|\text{BOS})P(\text{を}|\text{データ})P(\text{解析}|\text{を})P(\text{する}|\text{解析})$$

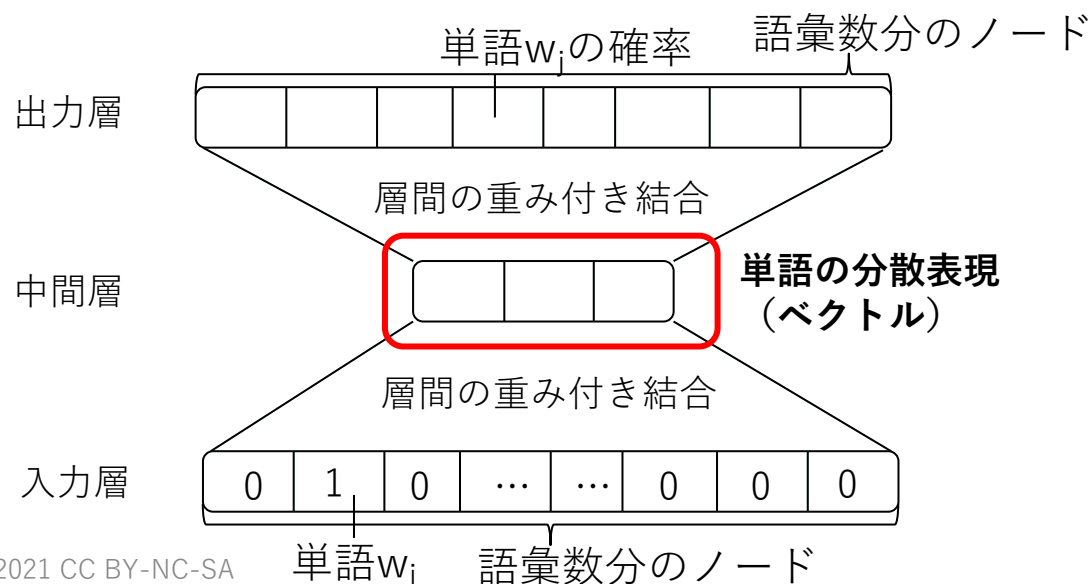
$P(\text{単語}i|\text{単語}i-1)$ はコーパスにおける単語*i*の頻度に対して単語*i*と単語*j*が隣接して出現する頻度の割合を元に計算できる

- 以下ではGoogle Booksをコーパスとしたn-gramの統計情報を見ることができます。
  - <https://books.google.com/ngrams>

# ニューラル言語モデル

- ニューラル言語モデルでは単語列を時系列とみなし，単語を分散表現（ $n$ 個の実数の組みからなるベクトル）として表し，時系列上で単語を予測することで言語モデルを学習します。
  - この時，単語の分散表現はニューラルネットワークのパラメータを学習することで獲得されます（**表現学習**）。
  - 分散表現の学習によって，言語表現（その背後にある情報）を低次元のベクトルとして表すことができます。
  - 単語のほか，句，文，段落，文書レベルの分散表現の学習も可能です。

ニューラルネットワークによる  
言語モデル（バイグラム）の学習  
（単語 $w_i$ から単語 $w_j$ の予測）

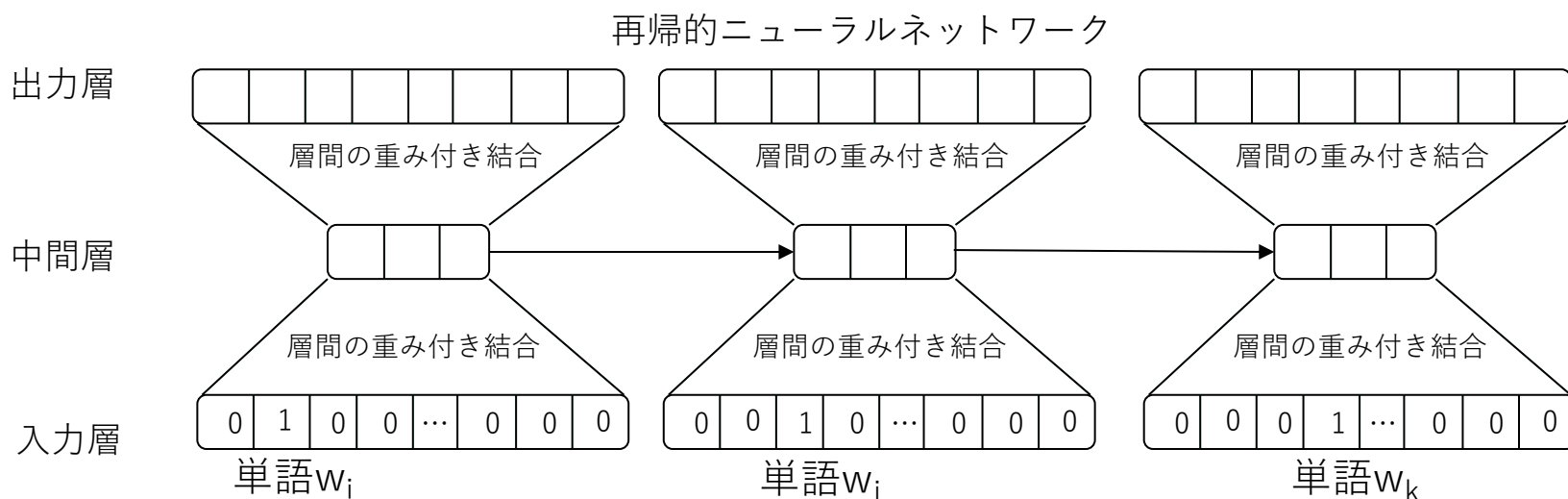


# 補：表現学習と単語の分散表現

- 事象や概念などの離散的な対象，具体的にはテキスト，画像，時系列などのデータ，の特徴（表現）を学習することを表現学習と呼びます
  - 表現学習では特徴を分散表現 (distributed representation) と呼ばれるベクトル（ベクトルの各次元は一般に実数値をとる）として学習します.
- ニューラル言語モデルでは単語の予測を通して単語の分散表現を学習しています.
  - 入力単語に対応した入力層のノードから中間層への結合重みとして，入力単語の分散表現を獲得することができます.
- 単語の分散表現をより効率的に求めるツールとして「word2vec」があります.
  - word2vecでは単語の文脈から単語を予測（または単語からその文脈の単語を予測）するようなニューラルネットワークを学習します.
    - より具体的にはCBow と skip-gram という2種類のモデルがあります.

# ニューラル言語モデル

- ニューラル言語モデルのニューラルネットワークの構造として自己ループを導入した再帰的なニューラルネットワーク（RNN: Recurrent Neural Network）を用いることで、より長い系列の言語モデルを扱うことが可能になりました。ニューラル言語モデルは例えば機械翻訳や自然言語生成などへ応用される言語モデルとなりました。
- 単語（あるいは文）の分散表現，再帰的なニューラルネットワークを用いたニューラル言語モデルは現在さまざまな自然言語処理のタスクにおいて活用されています。



## 9. 機械翻訳



# 機械翻訳

- ある原言語から別の目的言語への機械的処理による翻訳を機械翻訳と呼びます。
- 機械翻訳の研究は半世紀にわたる歴史がありますが、90年代以降は大規模な対訳データ（対訳コーパス）を元に、原言語の文と目的言語の文の語の対応や語の順番を統計的に学習する方法である統計的機械翻訳が主流となりました。
- 統計的機械翻訳は雑音のある通信路モデルに基づいています。
  - ある原言語の文 $s$ に対して可能なさまざまな目的言語の翻訳文 $t$ を列挙し、それらへ翻訳される確率を求めます。この確率を最大化する（誤りを最小化する）ような翻訳文 $t^*$ を求めます。

$$t^* = \operatorname{argmax}_t P(t|s) = \operatorname{argmax}_t P(s|t)P(t)/P(s) = \operatorname{argmax}_t P(s|t)P(t)$$

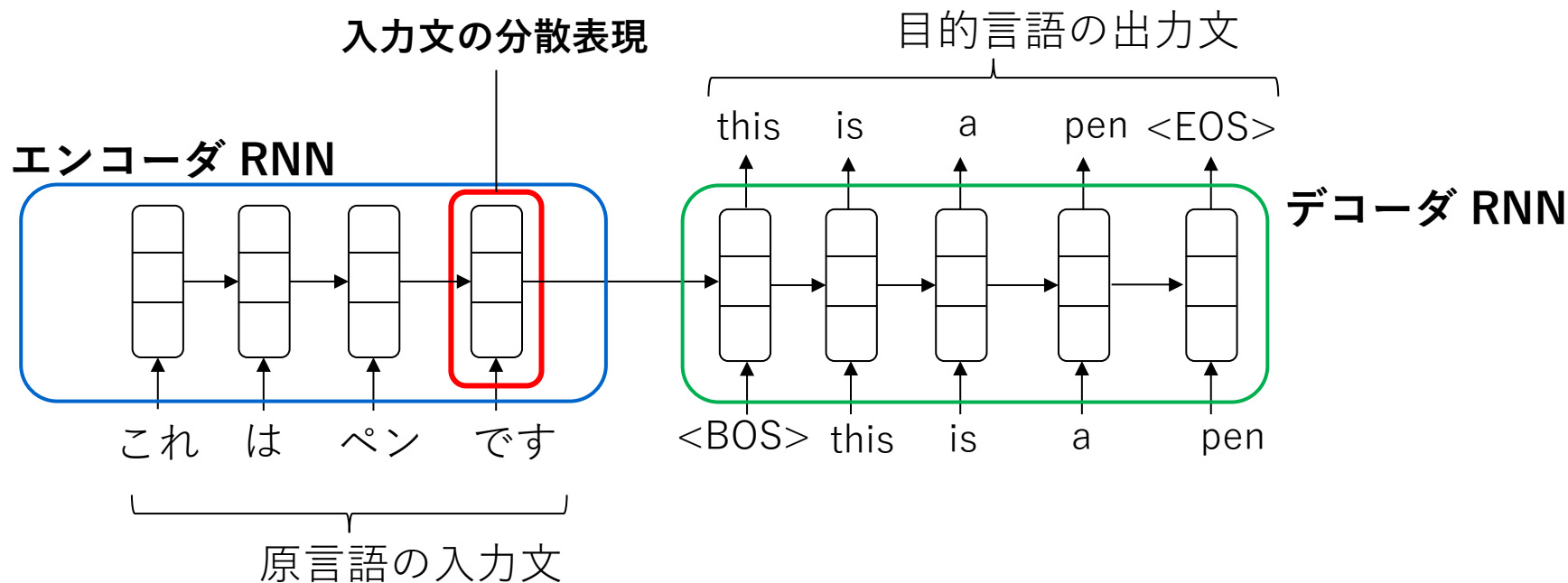
# 機械翻訳

- 統計的機械翻訳は雑音のある通信路モデルに基づいています。
  - 原言語の文sは目的言語tに雑音がのって観測されたと考えます.
  - この時, 文sから元の文tへ復元 (暗号を解読) することが翻訳となります
$$t^* = \operatorname{argmax}_t P(t|s) = \operatorname{argmax}_t P(s|t)P(t)/P(s) = \operatorname{argmax}_t P(s|t)P(t)$$
    - $P(t)$ : 目的言語の文tが生成される確率 (文tの尤もらしさ) (言語モデルで計算できます) .
    - $P(s|t)$ : 文tが文sに翻訳される確率. 原言語と目的言語の語の統計的な対応を元に近似します. より一般的にはこの確率を最大化するように対訳コーパスを元に教師なし学習を行います.
- 文sが文tに翻訳される確率 $P(t|s)$ を複数の関数の組み合わせによってモデル化することもできます. この時, 翻訳品質を数値化した評価尺度 (BLEU) を元に, 評価尺度が最大となるように学習を行うことができます.
  - あらかじめ原文を人間が翻訳した参照訳を数種類用意しておき, 翻訳文との類似度をはかることによって翻訳品質を数値化したものをBLEUと呼びます.

# ニューラル機械翻訳

- 深層学習技術の発展により，ニューラルネットワークを用いた機械翻訳手法（ニューラル機械翻訳）は，それまでの翻訳精度を大きく向上させ，機械翻訳の実用化につながりました。
- ニューラル機械翻訳では対訳ペアなどの明示的な対応関係を表現する代わりに，2言語の対応付けをベクトルによって表現し，この表現をニューラルネットワークによりデータから学習することで単一のモデルで機械翻訳を実現します。
  - モデルはエンコーダ・デコーダモデル（あるいは系列・系列モデル）と呼ばれ，エンコーダを通して原言語の入力文を分散表現で表現し，デコーダにより目的言語の出力文（長さは逐次可変）を生成します。
  - エンコーダ，デコーダにはそれぞれ再帰的ニューラルネットワーク（RNN）を用い，大規模な対訳データを元にして入力・出力となる単語の系列の表現を学習します。

# ニューラル機械翻訳



より実用的には

- 文頭から文末だけでなく文末から文頭という逆方向のモデル化も行います。
- RNNのほか、系列の情報の流れをより詳細に制御するLSTMやGRUなどのモデルも用いられます。
- 入力文の単語列のどの部分に着目するかを考慮する注意 (attention) 機構も併せて用いられます。

# ニューラル機械翻訳

## Google翻訳の例

英語 - 自動検出

英語

日本語

韓国語

▼

↔

日本語

英語



韓国語

▼

Machine translation, sometimes referred to by the abbreviation MT (not to be confused with computer-aided translation, machine-aided human translation or interactive translation), is a sub-field of computational linguistics that investigates the use of software to translate text or speech from one language to another.

×

319 / 5000




 

機械翻訳は、MTという略語（コンピューター支援翻訳、機械支援人間翻訳、インタラクティブ翻訳と混同しないでください）と呼ばれることもあり、テキストや音声を翻訳するためのソフトウェアの使用を調査する計算言語学のサブフィールドです。ある言語から別の言語へ。

☆

Kikai hon'yaku wa, MT to iu ryakugo (konpyūtā shien hon'yaku, kikai shien ningen hon'yaku, intarakutibu hon'yaku to kondō shinaide kudasai) to yoba reru koto mo ari, tekisuto ya onsei o hon'yaku suru tame no sofutō~ea no shiyō o chōsa suru keisan

[さらに表示](#)

## 10. 自然言語生成

# 自然言語生成

- 機械翻訳では原言語を入力として目的言語を出力しています。入力を言語に限らず、ある情報（例えば画像）を元に自然言語に変換して出力することを**自然言語生成**と呼びます。
  - 例えば、画像を入力として言語を出力とする自然言語生成では入力 of 画像のキャプションを生成することができます
- 自然言語生成したテキストは目的に応じて、自然さ・簡潔さ、可読性、正確性、結束性、一貫性、ドメインや目的への適合性、などを備えている必要があります。
- 自然言語生成を含むさまざまな言語処理タスクにおいて近年、注意機構に基づいたモデルであるTransformerが利活用されています。
- 巨大な事前学習TransformerモデルのGPT-3 (Generative Pre-trained Transformer)は自然なテキストを生成できることでさまざまな応用が期待される一方でフェイクニュースなど悪意ある利用について議論が起こっています。