

# パーザを作ろう

(依存構造解析 編)

---

2013/03/05  
Hiroyasu Yamada  
dyama.h@gmail.com

# 発表の目的

依存構造解析器を実際に作てみたい！

その最初の一歩のお手伝い

「作る」という観点で基本事項を説明

# 参考資料

## 読むべき文献や資料

- ◆ [1] MacDonald & Nivre (2011 CL)
- ◆ [2] 海野さんのスライド

<http://www.slideshare.net/unnonouno/ss-5724050>

- ◆ [3] COLING06のチュートリアルのスライド

<http://robot.cmpe.boun.edu.tr/robsem/dependency%20parsing.pdf>

僕のtumblrのページにも情報あげときます

<http://dyama1813.tumblr.com/post/44608439429/dependency-parser>

# 統計的依存構造解析器作成に必要な3要素

## ■入出力

□解析アルゴリズム

□機械学習との関連

# 依存構造解析の入出力

入力：文を構成する単語列（※今回は品詞付と仮定）

出力：文の依存構造

単語間の修飾関係（とその種類）

- 句構造との違い

- 記述の種類

※[3]COLING06チュートリアルのスライド参照

# 統計的依存構造解析器作成に必要な3要素

□入出力

■解析アルゴリズム

□機械学習との関連

# 主要な依存構造解析アルゴリズム

## Transition Based Algorithm

- ◆ Arc-standard
- ◆ Arc-eager (Malt Parser)

## Graph Based Algorithm

- ◆ Eisner's Algorithm
- ◆ MST Algorithm (MST Parser)

今日は arc-eager を紹介

# 用語 (1/3)

## Dependency Graph

- ◆ 単語がノード、依存関係がエッジに対応
- ◆ 非循環有向グラフ
- ◆ 依存関係：親ノードから子ノードの向き
  - ◆ 依存関係には主語、述語のような統語的な役割をラベル付けることも可能
  - ◆ 親：非修飾語 子：修飾語  
(親をhead, 子をdependentと呼ぶこともある)

# 用語 (2/3)

## Dependency Treeの重要な性質1

(Treeとしての性質)

- ◆ ルートが1文で1つ（文の主辞に相当）
- ◆ Single head : ルート以外のノードは必ず1つの親ノードを持つ

# 用語 (3/3)

## Dependency Tree の重要な性質2

- ◆ Projective : エッジ同士は非交差
- ◆ Non-Projective : エッジ同士が交差する場合がある

※英語や日本語の依存関係は原則 Projective

## Projective / Non-Projective

→ 解析アルゴリズムに大きく影響

(※今回は Projective な依存構造を対象)

# Transition Based Algorithm

状態遷移を繰り返しながら依存構造を構築

句構造解析のShift-Reduce 法に類似した解析手法

- ◆ arc-eager
- ◆ arc-standard

上記 2 手法は類似、いずれもとてもシンプル

今回は arc-eager を紹介

# arc-eager : 概要

状態は3つ組：(S, Q, A)で表現

- ◆ S: stack : 解析途中のノードを格納
- ◆ Q: queue : 未解析のノードを格納
- ◆ A: 構築済みの依存構造を格納

状態遷移アクションを繰り返し適用

SとQが空になれば解析終了

# arc-eager : 状態遷移アクション

## 4種類の状態遷移アクション

StackトップをSt、Queue の先頭をQ[0]

依存関係: \$parent→\$child

- ◆ Shift : Q[0]の単語を 1 つにSにプッシュ
- ◆ Reduce : Stをポップ
- ◆ LeftArc : Stをポップ。Q[0]→StをAに追加
- ◆ RightArc : St→Q[0]をAに追加。Q[0]をSにプッシュ

※LeftArc-subjやRightArc-objなど統語的な役割を解析することも可能（今回は省略）

# arc-eager : 状態遷移アクション適用条件

## 正しい依存構造が既知の場合

以下の順序で適用条件に当てはまるアクションを実行

- ◆ LeftArc :  $Q[0] \rightarrow St$  という依存関係がある場合
- ◆ RightArc :  $St \rightarrow Q[0]$  という依存関係がある場合
- ◆ Reduce :  $St$  の親が決まっていて部分木が完成している場合
- ◆ Shift : 上記以外

# arc-eager (0/16)

Action:

[ ]S [Economic news had little effect on financial markets]Q

[ ]A

# arc-eager (1/16)

Action: Shift

[ ]S [Economic news had little effect on financial markets]Q

[ ]A



[Economic ]S [news had little effect on financial markets]Q

[ ]A

# arc-eager (2/16)

Action: **LeftArc**

[Economic ]S [news had little effect on financial markets]Q

[ ]A



[ ]S [news had little effect on financial markets]Q

[news→Economic ]A

# arc-eager (3/16)

Action: **Shift**

[ ]S [news had little effect on financial markets]Q

[news→Economic ]A



[news]S [had little effect on financial markets]Q

[news→Economic ]A

# arc-eager (4/16)

Action: **LeftArc**

[news ]S [had little effect on financial markets]Q

[news→Economic ]A



[ ]S [had little effect on financial markets]Q

[news→Economic, had→news ]A

# arc-eager (5/16)

Action: **Shift**

[ ]S [had little effect on financial markets]Q

[news→Economic, had→news ]A



[had ]S [little effect on financial markets]Q

[news→Economic, had→news ]A

# arc-eager (6/16)

Action: Shift

[had ]S [**little** effect on financial markets]Q

[news→Economic, had→news ]A



[had **little** ]S [effect on financial markets]Q

[news→Economic, had→news ]A

# arc-eager (7/16)

Action: **LeftArc**

[had **little**]S [**effect** on financial markets]Q

[news→Economic, had→news]A



[had]S [effect on financial markets]Q

[news→Economic, had→news, **effect**→**little**]A

# arc-eager (8/16)

Action: **RightArc**

[had ]S [effect on financial markets]Q

[news→Economic, had→news, effect→little]A



[had **effect** ]S [on financial markets]Q

[news→Economic, had→news, effect→little, **had→effect**]A

# arc-eager (9/16)

Action: **RightArc**

[had **effect**]S [on financial markets]Q

[news→Economic, had→news, effect→little, had→effect]A



[had effect **on**]S [financial markets]Q

[news→Economic, had→news, effect→little, had→effect,

**effect→on**]A

# arc-eager (10/16)

Action: Shift

[had effect on ]S [financial markets]Q

[news→Economic, had→news, effect→little, had→effect

effect→on ]A



[had effect on financial ]S [markets]Q

[news→Economic, had→news, effect→little, had→effect

effect→on ]A

# arc-eager (11/16)

Action: **LeftArc**

[had effect on **financial**]S [**markets**]Q

[news→Economic, had→news, effect→little, had→effect

effect→on ]A



[had effect on ]S [**markets**]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, **markets**→**financial** ]A

# arc-eager (12/16)

Action: RightArc

[had effect **on**]S [markets]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial ]A



[had effect on **markets**]S [ ]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial, **on→markets** ]A

# arc-eager (13/16)

Action: **Reduce**

[had effect on **markets**]S [ ]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial, on→markets ]A



[had effect ]S [ ]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial, on→markets ]A

# arc-eager (14/16)

Action: **Reduce**

[had effect **on** ]S [ ]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial, on→markets ]A



[had effect ]S [ ]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial, on→markets ]A

# arc-eager (15/16)

Action: **Reduce**

[had **effect**]S [ ]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial, on→markets ]A



[had]S [ ]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial, on→markets ]A

# arc-eager (16/16)

Action: Reduce

[had]S [ ]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial, on→markets ]A



[ ]S [ ]Q

[news→Economic, had→news, effect→little, had→effect

effect→on, markets→financial, on→markets ]A

# arc-eager:まとめ

**left-to-right** で依存構造を構築

- ◆ LeftArc ボトムアップ
- ◆ RightArc トップダウン（親が先に決定）

※入力単語nに対し $2n$ の状態遷移で解析可能

**疑問：依存構造が未知の文**

状態遷移アクションは誰が決めるのか？

# 統計的依存構造解析器作成に必要な3要素

- 入出力
- 解析アルゴリズム
- 機械学習との関連

# 状態遷移アクションの学習

平文に対するarc-eagerの実現

→各状態で状態遷移アクションの推定が必要

訓練時は訓練データそのものをarc-eagerで解析

- ◆  $x$ : 状態の素性ベクトル
- ◆  $y$ : 状態遷移アクション

各状態遷移で( $x, y$ )を獲得

SVMなど分類器系の学習アルゴリズムを利用可能

# 状態の素性ベクトル化

## 素性ベクトル化の例

Action: Shift

[had ]S [little effect on financial markets]Q  
VBD ADJ NN IN ADJ NNS

[news→Economic, had→news ]A

素性: {St.p: Stの品詞、 St.w: Stの単語、

Q[0].p: Q[0]の品詞、 Q[0].w: Q[0]の単語}

x = {St.p:VBD, St.w.had, Q[0].p:ADJ, Q[0].w:little}

※実際は（精度をあげるには）もう少し複雑な素性が必要

# まとめ

依存構造解析の基本事項の説明

入出力：依存グラフ、依存木、etc

解析アルゴリズム：arc-eagerアルゴリズムの紹介

機械学習との関連：状態からアクションへの分類

練習がてらarc-eagerをまずは作ってみよう！

(もちろん他のアルゴリズムでもOK)