Compositional Vector Space Models for Knowledge Base Completion

最先端 NLP **勉強会** 2015

Saku Sugawara

東京大学相澤研究室

August 29, 2015

論文

Compositional Vector Space Models for Knowledge Base Completion
Arvind Neelakantan, Benjamin Roth, Andrew McCallum
Univ. Massachusetts
ACL2015

要点

- Knowledge Base Completion **の論文**
- relation **の含意関係の合成計算を** Reccurent NN **で行う**
- multi-hop relations(=path) **の推論を** non-atomically **に行う**
- **既存手法よりも** 7-10%**程度高い精度を達成**

Overview

Knowledge Base における知識とは

- (Barack Obama, presidentOf, USA)
- (Brad Pitt, marriedTo, Angelina Jolie)

のように、

$$(entity1, relation, entity2)$$

の形をしているもの

Knowledge Base Completion

- 既存の知識を使って新しい知識を補完するタスク
- 既存の知識ベースは不完全であり、必要な情報が欠けている
  - 個人の情報の国籍が欠けている
    $\Rightarrow$ 出生地や居住地から推測できるのでは？
  - ベンガルトラはしっぽを持つか？
    $\Rightarrow$ ベンガルトラはトラ、トラはネコ科、ネコ科は……
- 必要に応じて推論して補完してあげる必要がある

単純には……

- relation でグラフを作りパスをたどるだけで新しい知識になる

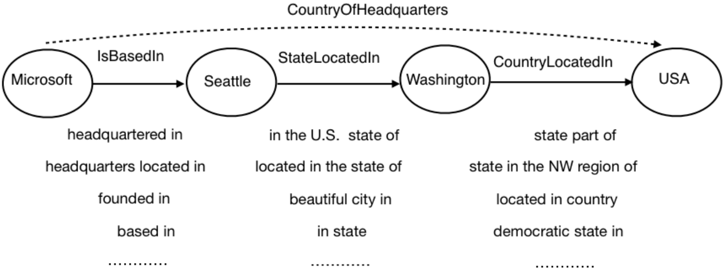

Figure 1: Semantically similar paths connecting entity pair (Microsoft, USA).

- 膨大な数になってしまう

先行研究 1 - Path Ranking Algorithm (PRA)

- Lao et al., 2010-2012
- **パスを全列挙せず、**
- **ランダムウォークで探索して、**
- **スコア付けして学習（雰囲気です）**
  $\Rightarrow$ **計算が早くても、結局は特徴の数が多くなってしまう**

先行研究 2 - Vector representations of relations

- (Bordes+, 2013), (Nickel+, 2011), (Socher+, 2013) **など**
- relation **をベクトル表現にして、類似度を測れるようにする**
  $\Rightarrow$ **ただし、推論の一単位が個々の** relation **のみ**
- **行列やテンソルを使ったものもある**

先行研究 3 - Cluster PRA

- Gardner et al., 2013-2014
- **訓練済みのベクトル表現を用いる**
- **しかし** relation type **のクラスタリングは** atomic **なパスの特徴で済ませている**
  $\Rightarrow$ relation type **に依存しない汎用的なモデルに到達できない**

本研究の貢献

手法として：

- **パスである** relation **のつながりを推論する**
- **ベクトル表現を用いて一般化**
- non-atomically **かつ** compositionally **な推論でさらに一般化**

さらに：

- **新しいデータセットを構築** (52 million triples)
  http://iesl.cs.umass.edu/downloads/inferencerules/release.tar.gz

とのこと

Recurrent Neural Networks for KB Completion

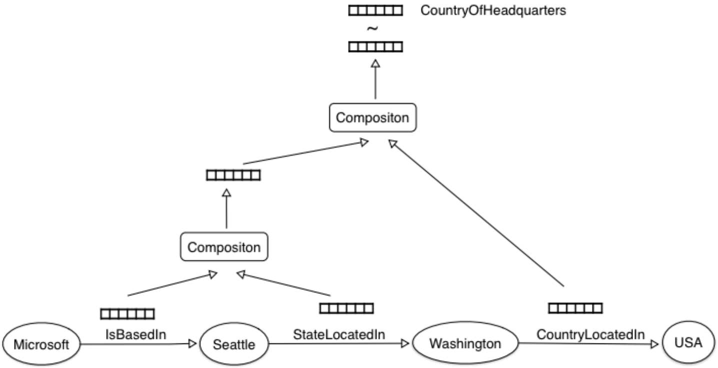

Figure 2: Vector Representations of the paths are computed by applying the composition function recursively.

Composition

relation $\delta = r_1 \rightarrow r_2 \rightarrow \ldots \rightarrow r_n$ **に対して、次のように計算する**

$$v(r_{1 \ldots i}) = f(W_\delta[v(r_{1 \ldots i-1}); v(r_i)])$$

- $v(\_) \in \mathbb{R}^d$ **は** relation **のベクトル表現**
- f **はシグモイド関数**
- $W_\delta \in \mathbb{R}^{d \times (2d+1)}$ **は** relation δ **の合成行列** (to be trained)
- semicolon $=$ concatenation^{2d} $+$ bias feature

Model Training

ひとつの relation δ **に対して、次のパラメータを学習する：**

$$\Theta = \{W_\delta, v_r(\omega) \forall \omega \in \Delta\}$$

目的関数は、対数尤度に L2 **正則化項を加えて、**

$$\Theta^* = \arg\max_{\Theta} \sum_{\lambda=(\gamma,\delta)\in\Lambda_\delta^+} P(y_\lambda = 1; \Theta) + \sum_{\lambda=(\gamma,\delta)\in\Lambda_\delta^-} P(y_\lambda = 0; \Theta) - \rho||\Theta||^2$$

ここで、y_λ **は** λ **が正しいとき** 1**、そうでないとき** 0
AdaGrad(Duchi et al., 2011) **で更新**

Model Training

また、λ の確率を表す P は、

$$P(y_\lambda = 1; \Theta) = sigmoid(v_p(\mu_\lambda) \cdot v_r(\delta))$$
$$P(y_\lambda = 0; \Theta) = 1 - P(y_\lambda = 1; \Theta)$$
$$\text{where} \mu_\lambda = \arg\max_{\pi \in \Phi_\delta(\gamma)} (v_p(\mu_\pi) \cdot v_r(\delta)$$

Φ_δ **は、**entity pair γ **に対して** δ **に等しいパスの集合** (PRA **で与えられる**)

Zero-shot KB Completion

- relation vectors **を訓練済みのものに固定する**
  - from (Riedel et al., 2013)
- **一般化された合成行列のみを学習する**
- **すなわち、すべての** relation **を同一の行列で学習**

Task

- PRA で重みのないパスを事前に列挙しておく
- 与えられた relation と entity の片割れに対して、適切なパスを推測する
- 欲しい答えが出せれば正解

Data

- ClueWeb dataset(Orr+, 2013) で Freebase の relation を拡張
- entity は Freebase にあるもののみを利用し、entity のペアを含む文章を ClueWeb から抜き出して新しい知識とする
- relation は次のような形式で表現される；
    - /people/person/place_of_birth
    - /government/polled_entity/poll_scores
    - /music/artist/genre

Data

Entities	18M
Freebase triples	40M
ClueWeb triples	12M
Relations	25,994
Relation types tested	46
Avg. paths/relation	2.3M
Avg. training facts/relation	6638
Avg. positive test instances/relation	3492
Avg. negative test instances/relation	43,160

Results - example1

Relation: /book/written_work/original_language/ *(book "x" written in language "y")*
Seen paths:
/book/written_work/previous_in_series → /book/written_work/author → /people/person/nationality → /people/person/nationality^{-1} → /people/person/languages
/book/written_work/author → /people/ethnicity/people^{-1} → /people/ethnicity/languages_spoken
Unseen paths:
"in"$^{-1}$ - "writer"$^{-1}$ → /people/person/nationality^{-1} → /people/person/languages
/book/written_work/author → addresses → /people/person/nationality^{-1} → /people/person/languages

Relation: /people/person/place_of_birth/ *(person "x" born in place "y")*
Seen paths:
"was,born,in" → /location/mailing_address/citytown^{-1} → /location/mailing_address/state_province_region
"from" → /location/location/contains^{-1}
Unseen paths:
"born,in" → /location/location/contains → "near"$^{-1}$
"was,born,in" → commonly,known,as^{-1}

Results - example2

Relation: /geography/river/cities/ *(river "x" flows through or borders "y")*
Seen paths:
"at" $\longrightarrow$ /location/location/contains^{-1}
"meets,the" $\longrightarrow$ /transportation/bridge/body_of_water_spanned^{-1} $\longrightarrow$ /location/location/contains^{-1} $\longrightarrow$ "in"
Unseen paths:
/geography/lake/outflow^{-1} $\longrightarrow$ /location/location/contains^{-1}
/geography/lake/outflow^{-1} $\longrightarrow$ /location/location/contains^{-1} $\longrightarrow$ "near"

Relation: /people/family/members/ *(person "y" part of family "x")*
Seen paths:
/royalty/monarch/royal_line^{-1} $\longrightarrow$ /people/person/children $\longrightarrow$ /royalty/monarch/royal_line $\longrightarrow$ /royalty/royal_line/monarchs_from_this_line
/royalty/royal_line/monarchs_from_this_line $\longrightarrow$ /people/person/parents^{-1} $\longrightarrow$ /people/person/parents^{-1} $\longrightarrow$ /people/person/parents^{-1}
Unseen paths:
/royalty/monarch/royal_line^{-1} $\longrightarrow$ "leader"$^{-1}$ $\longrightarrow$ "king" $\longrightarrow$ "was,married,to"$^{-1}$
"of,the"$^{-1}$ $\longrightarrow$ "but,also,of" $\longrightarrow$ "married" $\longrightarrow$ "defended"$^{-1}$

Results - separate RNN

	train with top 1000 paths	train with all paths
Method	MAP	MAP
PRA Classifier	43.46	51.31
Cluster PRA Classifier	46.26	53.23
Composition-Add	40.23	45.37
RNN-random	45.52	56.91
RNN	46.61	56.95
PRA Classifier-b	48.09	58.13
Cluster PRA Classifier-b	48.72	58.02
RNN + PRA Classifier	49.92	58.42
RNN + PRA Classifier-b	51.94	**61.17**

-b では、PRA で bigram features を用いている

Results - Zero-shot

	train with top 1000 paths	train with all paths
Method	MAP	MAP
RNN	43.82	50.10
zero-shot	19.28	20.61
Random	7.59	

Table 4: Results comparing the zero-shot model with supervised RNN and a random baseline on 10 types. RNN is the fully supervised model described in section 3 while zero-shot is the model described in section 4. The zero-shot model without explicitly training for the target relation types achieves impressive results by performing significantly ($p < 0.05$) better than a random baseline.

問題点と今後

- RNN **が、局所的な特徴を保持できていない**
  - PRA **の** bigram features **でスコアが上がったのと同じように記憶の保持が要る**

  $\Rightarrow$ memory module **の実装**
- entity **にもベクトル表現を導入し、動詞句の多義性** (polysemy) **の問題に取り組む**

まとめ (要点)

- relation **の含意関係の合成計算を** Reccurent NN **で行う**
- multi-hop relations(=path) **の推論を** non-atomically **に行う**
- **既存手法よりも** 7-10%**程度高い精度を達成**