論文紹介

When Are Tree Structures Necessary for Deep Learning of Representations?

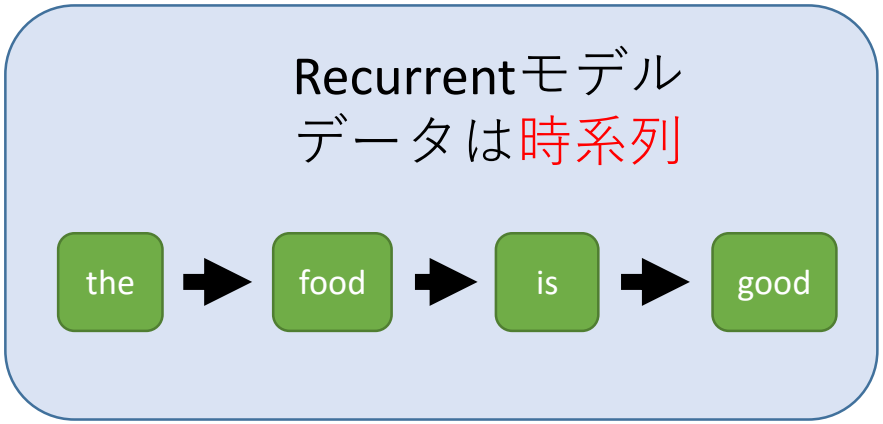
豊田工業大学　知能数理研究室
修士1年　辻村有輝

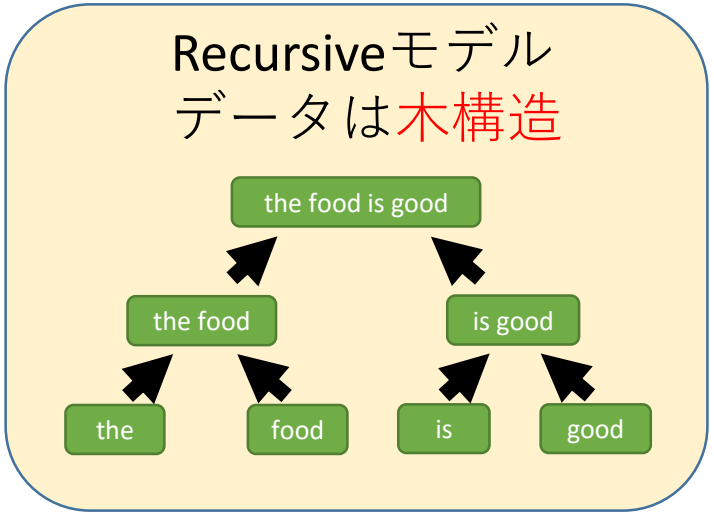
論文について

- タイトル
  - When Are Tree Structures Necessary for Deep Learning of Representations?
- 著者
  - Jiwei Li, Minh-Thang Luong, Dan Jurafsky (Stanford University)
  - Eduard Hovy (Carnegie Mellon University)
- 発表学会
  - EMNLP 2015
- 内容
  - Recurrent/Recursiveなニューラルネットワークの比較

モチベーション

- ニューラルネットワークで用いられるRecursiveモデルがいつ・なぜRecurrentモデルに勝るのかを調べたい
  - 時系列では長距離になる関係も木構造なら近い関係にできることが
  - しかしRecurrentモデルに十分処理しきる能力があるかもしれない





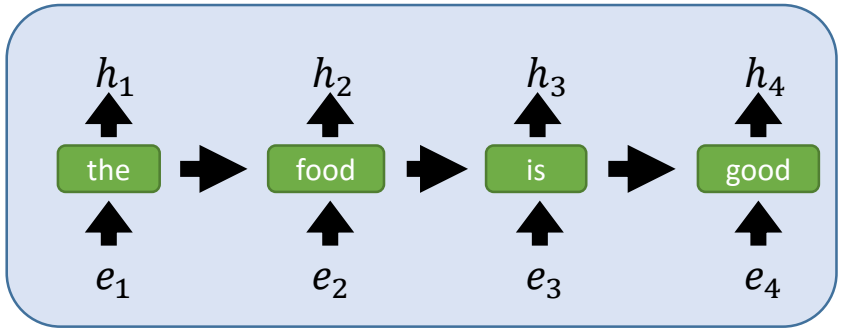
内容

- 4種のタスクについてRecurrentモデルとRecursiveモデルを出来る限り同じ条件にして性能比較
  - 学習方法を統一（AdaGradで学習など）
- 長距離間の関係を考慮する必要のあるタスク（関係抽出）だけRecursiveモデルが有意によかった

実験で使用するモデル

Recurrentモデル

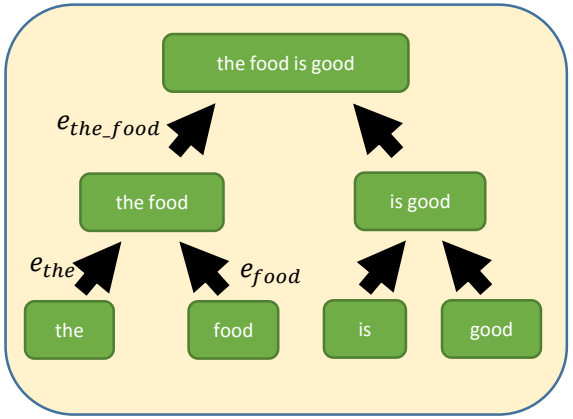
時系列データが対象



- モデルは2種類＋各Bidirectionalモデル
  - Standard：$h_t = f(\mathrm{W}h_{t-1} + \mathrm{V}e_t)$
  - LSTM
  - Bidirectional：$h_t = f(\mathrm{W}_L[h_t^{\leftarrow}, h_t^{\rightarrow}])$

Recursiveモデル

木構造データが対象



- モデルは2種類でボトムアップ方式
  - Standard：$e_\eta = f\left(\mathrm{W}e_{\eta_{left}} + \mathrm{V}e_{\eta_{right}}\right)$
  - 木構造LSTM

実験を行うタスク

4種類5つのデータセット

- **Sentiment Classification**
  - Stanford Sentiment Treebank (Fine-Grained and Binary)
  - Pang et al. (2002) dataset (Binary)
- **Phrase Matching**
  - UMDQA dataset
- **Semantic Relation Classification**
  - SemEval-2010
- **Discourse parsing**
  - RST-DT corpus

実験結果

Sentiment Classification on the Stanford data set

構文木中の各ノード（句だったり節だったり文全体だったり）に
与えられている極性を予測
全データ：11,855文（215,154フレーズ）
うちテストデータ：82,600フレーズ＋2,210のルートノード

Standardモデル

	Fine-Grained	Binary
Tree	0.433	0.815
Sequence	0.420 (-0.013)	0.807 (-0.007)
P-value	0.042*	0.098
Bi-Sequence	0.435 (+0.08)	0.816 (+0.002)
P-value	0.078	0.210

ルートノードに対する識別結果

	Fine-Grained	Binary
Tree	0.820	0.860
Sequence	0.818 (-0.002)	0.864 (+0.004)
P-value	0.486	0.305
Bi-Sequence	0.826 (+0.06)	0.862 (+0.002)
P-value	0.148	0.450

フレーズに対する識別結果

ルートノードに対しては単方向SequenceよりはTreeの方が極僅かによかった

実験結果

Sentiment Classification on the Stanford data set

LSTMモデル

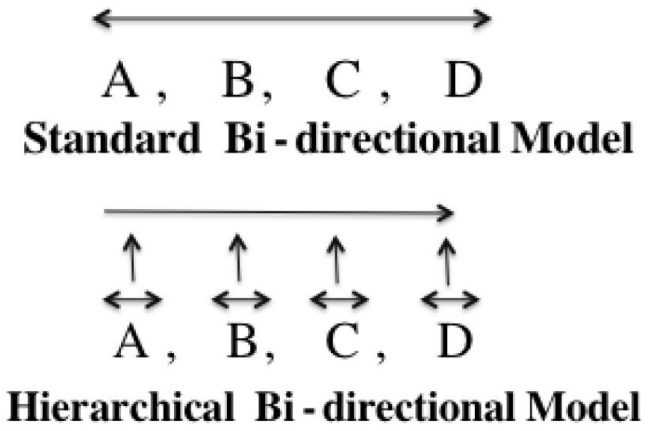
Tree LSTMはルートに対しての識別で良いといわれている

Model	all-fine	root-fine	root-coarse
Tree LSTM	**83.4** (0.3)	50.4 (0.9)	86.7 (0.5)
Bi-Sequence	83.3 (0.4)	49.8 (0.9)	86.7 (0.5)
Hier-Sequence	82.9 (0.3)	**50.7** (0.8)	**86.9** (0.6)

Hierarchical Model

文をコンマやクエスチョンマーク等で分割し，
それぞれを別々に双方向LSTMに入力して，
その後さらに単方向LSTMに入力

- 文の内部構造を捉える
- 各単語のタイムステップ数が少なくなり誤差が伝播しやすくなる



実験結果

Sentiment Classification on the Pang's data set

文ごとにラベルが与えられている
学習データ8101件　　開発データ500件　　テストデータ2000件

	Standard	LSTM
Tree	0.745	0.774
Sequence	0.733 (-0.012)	0.783 (+0.008)
P-value	0.060	0.136
Bi-Sequence	0.754 (+0.09)	0.790 (+0.016)
P-value	0.058	0.024*

構文木を使ってもうまくいかなかった
↑学習データが少なく，識別も文全体にしか行わないせい？

実験結果

Phrase Matching on the UMD-QA dataset

	Standard	LSTM
Tree	0.523	0.558
Sequence	0.525 (+0.002)	0.546 (-0.012)
P-value	0.490	0.046*
Bi-Sequence	0.530 (+0.007)	0.564 (+0.006)
P-value	0.075	0.120

質問文の解答になるフレーズを
あらかじめ決められた
解答プール中から選ぶ

学習

質問文をRNNに入力し各タイミングの出力e_η, e_tを正解フレーズの
Embedding $\vec{c}$に近づけランダムに選んだ不正解$\vec{z}$から遠ざける

木構造

$$L = \sum_{\eta \in [\text{parse tree}]} \sum_{z} max(0, 1 - \vec{c} \cdot e_\eta + \vec{z} \cdot e_\eta)$$

時系列

$$L = \sum_{t \in [1, N_s]} \sum_{z} max(0, 1 - \vec{c} \cdot e_t + \vec{z} \cdot e_t)$$

Recurrentモデルの構文的に正しくないタイミングでの出力も
正解に近いそれらしい出力となっている？

実験結果

Semantic Relation Classification on the SemEval 2010

文中の指定された2つのエンティティ間の関係を予測

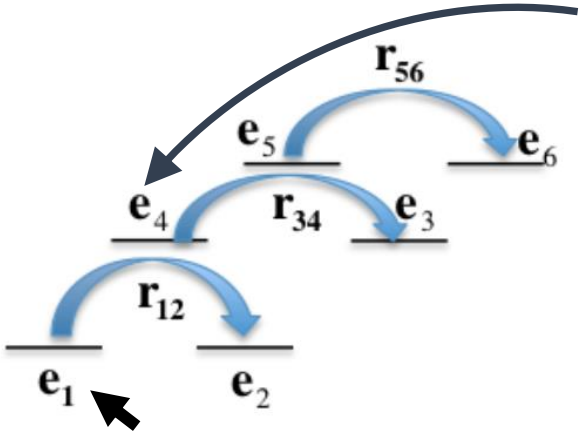
	Standard	LSTM
Tree	0.748	0.767
Sequence	0.712 (-0.036)	0.740 (-0.027)
P-value	0.004*	0.020*
Bi-Sequence	0.730 (-0.018)	0.752 (-0.014)
P-value	0.017*	0.041*

木構造モデルの方が性能が良い

- 他のタスクと比べて考慮すべき関係が時系列でみると遠い
- 時系列としては遠いが木構造上では近い関係になる

実験結果

Discourse Parsing on the RST-DT corpus



elementary discourse units（EDU）

各EDUは節を構成する単語列から成る

ここではEDU同士の関係（r_{xy}）予測の性能を比較

子EDUをマージして親EDUのベクトル表現を生成

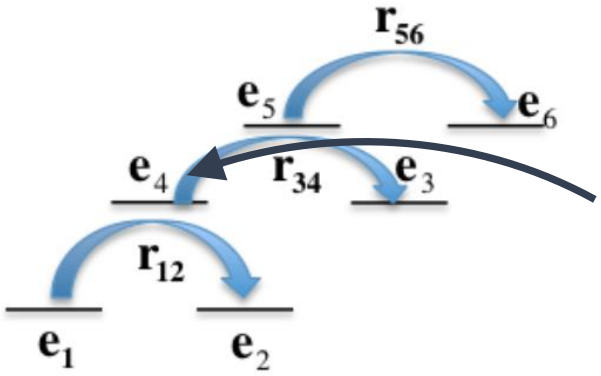
	Standard	LSTM
Tree	0.568	0.564
Sequence	0.572 (+0.004)	0.563 (-0.002)
P-value	0.160	0.422
Bi-Sequence	0.578 (+0.01)	0.575 (+0.012)
P-value	0.054	0.040*

あまり性能差は見られなかった

- 葉のEDUが短いため処理順がほぼ変わらず木構造の効果がなかった？
- 各EDUのマージ処理がフィルタの役割をした？（？？）
  - マージ処理でRecurrentでも木構造を考慮してしまったということ？

マージ方法（本当にこれ？）

Discourse Parsing on the RST-DT corpus



子EDUをマージして親EDUのベクトル表現を生成

Adaptive recursive neural network for target-dependent twitter sentiment classification（Li et al. 2014）によれば

標準的なRecursive NN

$$\boldsymbol{v} = f\big(g(\boldsymbol{v}_l, \boldsymbol{v}_r)\big) = f(\mathrm{W}[\boldsymbol{v}_l, \boldsymbol{v}_r] + \boldsymbol{b})$$

AdaRNN

$$\boldsymbol{v} = f\left(\sum_h P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r)\, g_h(\boldsymbol{v}_l, \boldsymbol{v}_r)\right)$$

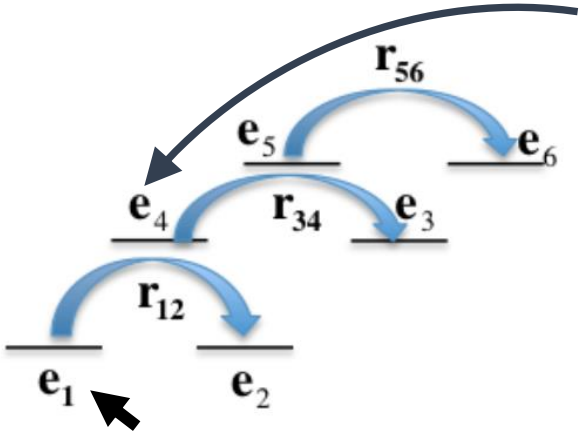
$$P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r) = softmax(\mathrm{W}_h^s[v_l, v_r])$$

$$g_h(\boldsymbol{v}_l, \boldsymbol{v}_r) = \mathrm{W}_h[v_l, v_r] + \boldsymbol{b}_h$$

ただし$\boldsymbol{v}$は親EDUの，$\boldsymbol{v}_l$と$\boldsymbol{v}_r$は子EDUのベクトル表現で，W，W_h^s，W_hは重み行列，$\boldsymbol{b}$，$\boldsymbol{b}_h$はバイアス，$P(g_h|\boldsymbol{v}_l, \boldsymbol{v}_r)$は子$\boldsymbol{v}_l$，$\boldsymbol{v}_r$の親が関係$g_h$である確率を表す

実験結果

Discourse Parsing on the RST-DT corpus



子EDUをマージして親EDUのベクトル表現を生成

	Standard	LSTM
Tree	0.568	0.564
Sequence	0.572 (+0.004)	0.563 (-0.002)
P-value	0.160	0.422
Bi-Sequence	0.578 (+0.01)	0.575 (+0.012)
P-value	0.054	0.040*

elementary discourse units（EDU）

各EDUは節を構成する単語列から成る

ここではEDU同士の関係（r_{xy}）予測の性能を比較

あまり性能差は見られなかった

- 葉のEDUが短いため処理順がほぼ変わらず木構造の効果がなかった？
- 各EDUのマージ処理がフィルタの役割をした？（？？）
  - マージ処理でRecurrentでも木構造を考慮してしまったということ？

まとめ

- Recursiveモデルが有利かもしれない状況
  - 時系列としては遠いが木構造上では近い関係になるとき
  - ルートノードに対する識別
- RecurrentをRecursiveと同等の性能にできるかもしれない方法
  - Bidirectionalにする
  - 元の長い文を，短い部分文に分割
- Recurrentモデルの構文的に正しくないタイミングでの出力もそれらしい出力となっている？

- もっと深いモデルにした時に同じ結果になるかは不明
- 平等のために学習アルゴリズムを統一したが実際はモデルごとの特性を考えればむしろ不平等だったかも

紹介者の感想

- 結局木構造は使わなくても大抵性能に影響がない？
- もうちょっと詳細な比較等が見てみたかった
- 意外に読むのが大変だった

プレゼン中に元論文の図や表を引用させていただきました.

補足資料：モデル

- 基本的には単語ベクトルをRecurrent/Recursive層に入力しその上に出力層が積まれる
  - Sentiment Classification中にはRecurrent層が二層のモデル（Hierarchical Model）がある
  - Phrase MatchingではRecurrent/Recursive層からの出力がそのまま識別・学習に使われる
  - Discourse parsingではRecurrent/Recursive層で文中の各フレーズのベクトル表現を作りそれらをマージしていき他クラス分類
- Recurrent/Recursiveの両モデルを出来る限り同じ条件にして実験
  - 全てAdaGradで学習
  - 調整は開発データor交差検定
  - 調整するパラメータには学習率とミニバッチ数，正則化項が含まれる