# A Recursive Information Flow Gated Model for RST-style Text-level Discourse Parsing

Longyin Zhang, Xin Tan, Fang Kong(✉), and Guodong Zhou

School of Computer Science and Technology, Soochow University, Suzhou, China
{lyzhang7, xtan}@stu.suda.edu.cn, {kongfang, gdzhou}@suda.edu.cn

**Abstract.** Text-level discourse parsing is notoriously difficult due to the long-distance dependency over the document and the deep hierarchical structure of the discourse. In this paper, we attempt to model the representation of a document recursively via shift-reduce operations. Intuitively, humans tend to understand macro and micro texts from different perspectives, so we propose a recursive model to fuse multiple information flows and strengthen the representation of text spans. During parsing, the proposed model can synthetically grade each information flow according to the granularity of the text. Experimentation on the RST-DT corpus shows that our parser can outperform the state-of-the-art in nuclearity detection under stringent discourse parsing evaluations.

**Keywords:** RST · Discourse Parsing · Tree-LSTM

## 1 Introduction

A document usually consists of many related text units organized in the form of constituency and dependency. As a representative linguistic theory about discourse structures, Rhetorical Structure Theory (RST) [11] describes a document as a discourse constituency tree. As shown in Fig. 1, each leaf node of the tree corresponds to an Element Discourse Unit (EDU) and relevant leaf nodes are connected by rhetorical relations to form bigger text spans recursively until the final tree is built. In particular, a label (either *nucleus* or *satellite*) is assigned according to the nuclearity of two neighboring document units, where the *nucleus* is considered more important.

In this paper, we address the task of RST-style discourse parsing, which aims to identify the overall rhetorical structure of the entire document. With the release of RST Discourse Treebank (RST-DT), text-level discourse parsing has been attracting more and more attention. However, the RST-DT corpus is limited in size since corpus annotation is time consuming and labor extensive. In this condition, most of previous studies heavily rely on manual feature engineering [5, 8]. With the increasing popularity of deep learning in NLP, some researchers turn to DNNs [9, 10, 3]. For example, Li *et al.* [9] propose a recursive deep model to compute the representation for each text span based on its subtrees. However, recursive deep models usually suffer from gradient vanishing.

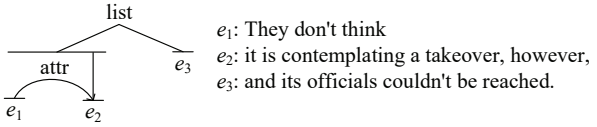


**Fig. 1.** An example of RST discourse tree, where $e_1$, $e_2$, $e_3$ and $e_4$ are EDUs, *list* and *attr* are discourse relation labels, and arrows indicate the nucleuses of relations.

In addition, humans tend to understand macro and micro texts from different perspectives. Li *et al.* [10] propose a hierarchical Bi-LSTM model to learn representations of text spans, which can store document-level information for a long period of time. However, text spans in RST-DT are formed recursively in nature, it is hard for sequence LSTMs to capture discourse structure information.

To address above challenges, we propose the recursive information flow gated model (R-IFGM) to learn representations of text spans. The proposed model can well store information for a long time and thus avoid gradient vanishing. In particular, we introduce two additional information flows, i.e., the state transition information for long distance dependency and the principal component of the text span for low-latitude representation of the text. Specially, the R-IFGM model can grade these information flows according to the granularity of each text span automatically. Compared with the state-of-the-art, the resulting parser obtains competitive performance under stringent discourse parsing evaluations and can strongly outperform the corresponding best parser in nuclearity detection with an absolute gain of 3.2% in F1-measure due to the modeling of multiple information flows in a recursive way.

## 2 Parsing Model

A shift-reduce discourse parser maintains two data structures: a stack of partially completed subtrees and a buffer of EDUs yet to be parsed. The parser is initialized with the stack empty and the buffer contains all EDUs ($e_1$, $e_2$, ..., $e_N$) of the document in order. When the action *reduce* is performed, the parser will choose both the type of nuclearity and relation between the two popped elements. The shift-reduce system transforms the structure building problem into a labeling decision problem. During parsing, the parser consumes transitions ($a_1$, $a_2$, ..., $a_k$, ..., $a^{2N-1}$) constantly according to the state of buffer and stack, where $a_k \in \{shift, reduce\}$. Mutually, the state of buffer and stack will change according to the predicted action label. When the buffer becomes empty and the stack has only one element, the shift-reduce process is ended. The last element in the stack is the target discourse constituency tree.

### 2.1 EDU Representation

The LSTM model is meant to maintain a rough summary of the portion of the sentence has been processed so far. Since an EDU is a sequence of words, we

employ a Bi-LSTM to encode each sequence constituted by the concatenation of word embeddings and POS embeddings, obtaining $H = (H_1, H_2, \ldots, H_n)$. Then we use the unweighted average to get a summarization of the hidden states, obtaining $\bar{H}$. Previous studies on discourse parsing have proved that words at the beginning and end of EDUs provide quite useful information [5, 7, 9]. Following their works, we formulate the representation of an EDU as:

$$x_e = [\bar{H}; w_1; w_n; p_1] \tag{1}$$

$$[h_0; c_0] = W_e x_e + b_e \tag{2}$$

where $[h_0; c_0]$ is a transformed representation of the EDU (leaf node) for LSTM-style computation.

### 2.2 Composition Function for Text Span Representation

The representations for non-leaf nodes serve as structure and relation classification. When the action *reduce* is performed, the top two elements in the stack are entered into a *composition function* to form the representation for a bigger text span. The proposed R-IFGM is also a recursive *composition function* that can generate the representation for each text span in a bottom-up fashion. Fig. 2 shows the architecture of R-IFGM, which is also a variant of the Tree-LSTM [14]. The architecture illustrates the input ($x$), cell ($c$) and hidden ($h$) nodes at the time $t$. It extends the sequence LSTM by splitting the previous hidden state vector $h_{t-1}$ into a left child's state vector $h^L_{t-1}$ and a right child's state vector $h^R_{t-1}$, splitting the previous cell vector $c_{t-1}$ into $c^L_{t-1}$ and $c^R_{t-1}$.

Different from other tree-structured LSTMs, we bring two additional information flows as external inputs. For the state transition information, we employ a state tracker here to track the changing states of the stack and buffer to capture the long distance dependency. For the text span's principal component, we introduce the *smooth inverse frequency* (SIF) [1] into this paper to supply a low dimension representation of the text span to form. Additionally, the hidden states of the left and right child nodes supply the structure information in nature. Generally, we expect the model to have the ability to automatically assign weights on the three gates to choose which information is more important

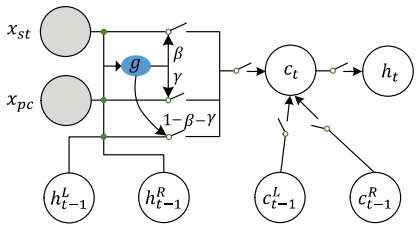


**Fig. 2.** Topology of the proposed recursive information flow gated model (R-IFGM).

in representing the parent node. Therefore, we use a mutually-exclusive gating mechanism to balance the three information flows.

When the action *reduce* is performed, the representations of two text spans are popped from the stack and fed into the *composition function* to compute the representation for the new parent node at the time $t$. The *composition function* calculates the parent node's cell vector $c_t$ and the hidden state $h_t$ as follows:

$$I_h = W_h h_{t-1} + b_h \tag{3}$$

$$I_s = W_s x_{st} + b_s, I_{pc} = W_{pc} x_{pc} + b_{pc} \tag{4}$$

$$\beta = \frac{e^{I_s}}{\sum_{I \in D} e^I}, \gamma = \frac{e^{I_{pc}}}{\sum_{I \in D} e^I} \tag{5}$$

$$g_t = W_g(\beta x_{st} \oplus \gamma x_{pc} \oplus (1 - \beta - \gamma) h_{t-1}) + b_g \tag{6}$$

$$c_t = \sum_{N \in \{L,R\}} f_{t-1}^N \odot c_{t-1}^N + i_t \odot g_t \tag{7}$$

$$h_t = o_t \odot \tanh(c_t) \tag{8}$$

where $h_{t-1}$ refers to the concatenation of the top two hidden states popped from the stack, $D = \{I_h, I_{pc}, I_s\}$, $f_{t-1}^N$ is the forget gate, $i_t$ is the input gate, $o_t$ is the output gate, $\oplus$ is the concatenation, and $\odot$ is the element-wise product. The new pair $(h_t, c_t)$ that represents a new text span is then pushed onto the stack. The external inputs $x_{pc}$ and $x_{st}$ are detailedly described in Section 2.3.

### 2.3 The Principal Component of Text Span & State Transition

Getting a precise representation of the text span is difficult as many documents are quite long sequences, so auxiliary information is urgently needed. In this section, we give an explanation about the two external inputs, i.e., the text span's principal component and the state transition information.

**The Principal Component of Text Span** Usually, it is difficult to capture semantic information recursively when the document tree is extremely deep. So, a method to abstract the most principal component of each text span is necessary. In a discourse tree, each inner node can uniquely identify a text area, which contains a set of EDUs that provide complete semantic information for the text area. We aim to translate each text span into a lower latitude representation. The SIF proposed by Arora *et al.* [1] is an unsupervised sentence embedding method. It achieves significantly better performance than certain supervised methods including LSTMs on most of textual similarity tasks. For a start, we borrow the SIF [1] to get unsupervised representation of each EDU in the text span. Then, we get a summarization of this text span by the unweighted average on the representations of EDUs. When the action *reduce* is performed, the top two text spans in the stack are popped to form a bigger text span and we formulate

the principal component of the text span as:

$$s = [s_1, \ldots, s_m] = \text{SIF}(E, p(w), a) \tag{9}$$

$$x_{pc} = average(s) \tag{10}$$

where $E$ is the set of EDUs in the text span, $p(w)$ denotes the estimated probabilities of words in the Wall Street Journal articles used in Treebank-2, and $a$ is the weighting parameter mentioned in the paper [1].

**State Transition** Our proposed discourse parser is a shift-reduce discourse parser, which maintains a stack and a buffer. During parsing time, the stack and buffer generate a series of changing states according to the transition actions. We aim to use a state tracker to capture the long distance dependency. The state tracker is a sequence tracking LSTM [2] and it is meant to maintain a low-resolution summary of the portion of the discourse has been processed so far. We choose the first element in the buffer and the top two elements in the stack to represent the state at the $t^{th}$ step, obtaining $state_t = [h^0_{t\_b}; h^{-1}_{t\_s}; h^{-2}_{t\_s}]$. Then, we take those changing states as a series of inputs to the state tracker. The output at the time $t$ will supply state transition information for the *composition function* (see Section2.2) and it also supplies features for the transition classifier to predict the next transition action at the time $t$.

$$x_{st}, c_t = \text{LSTM}(c_{t-1}, state_t) \tag{11}$$

## 3 Classification & Training

In this work, we combine the nuclearity and relation together, obtaining a set of NR labels. We build two classifiers in this paper for transition and NR classification. For the first classifier, we use the hidden state $i = x_{st}$ of the state tracker as its input at the time $t$. For the second, we use the hidden states of the two text spans encoded by R-IFGM as additional inputs for the NR tags are assigned directly between them. And the overall input is defined as $i' = [x_{st}; h_{span_l}; h_{span_r}]$. For each classifier, we feed these vectors into respective output layers:

$$y_s = \tanh(W_s i + b_s) \tag{12}$$

$$y_{nr} = \tanh(W_{nr} i' + b_{nr}) \tag{13}$$

where $W_s \in \mathbb{R}^l$, $W_{nr} \in \mathbb{R}^{m \times l'}$, $b_s \in \mathbb{R}$, $b_{nr} \in \mathbb{R}^m$ are model parameters, $m = 41$ is the number of combinations of nuclear and relation tags in RST-DT.

For transition classification, we expect the output score of the correct label to be much larger than the wrong one. And, we set a max-margin value as a threshold for this difference. For the NR classifier, we train it to maximize the conditional log-likelihood of gold labels. During training, the two classifiers are trained according to their respective goals and update the shared parameters alternatively. We add an additional max-margin objective to strengthen the

nuclearity classification. And the general training objective is defined as:

$$\mathcal{L}(\Theta) = -\log(p_{nr_g}) + \frac{\lambda}{2}\|\Theta\|^2 + \frac{\Sigma_{i=0}^{l_{y_s}}\max(0, M - y_{s_g} + y_{s_i})}{l_{y_s}} + \zeta\,\frac{\Sigma_{i=0}^{l_{y_n}}\max(0, M - y_{n_g} + y_{n_i})}{l_{y_n}} \quad (14)$$

where $p_{nr_g}$ is the softmax probability of the gold relation label, $l_{y_s} = 2$, $l_{y_n} = 3$ are the sizes of transition and nuclearity prediction, and $M$ is the margin value.

## 4 Experimentation

The RST-DT corpus annotates 385 documents from the WSJ which is mainly divided into two data sets (347 for training and 38 for testing). Following previous works, we binarize those non-binary subtrees with right-branching and use the 18 coarse-grained relations to evaluate our parser with gold-standard EDUs.

### 4.1 Experimental Settings

The metrics of RST discourse parsing evaluation include span, nuclearity and relation indication. Moery *et al.* [12] prove that most gains reported in recent publications are an artefact of implicit difference in evaluation procedures, and the original RST-Parseval overestimates the performance of discourse parsing. In this work, we randomly select 34 documents from the training set as the development set and employ a more stringent metric [12] to evaluate our parser. For fair comparison, scores we report are micro-averaged $F_1$ scores in default.

We optimized following parameters during training: the learning rate is 0.001, the dropout rate is 0.2, the dimension of POS tags is 50, the hidden size of R-IFGM is 640. The $l_2$ regularization parameter and the parameter $\zeta$ in the loss function are set by $10^{-5}$ and 0.6 respectively. We use word representation based on the 1024D vectors provided by ELMo [13] and we do not update the vectors during training. The model is trained with Adam to minimize the loss objective. We trained the model iteratively on the training set by 10 rounds. The development corpus was used to choose the best model and the final model was evaluated on the test set after parameter tuning.

### 4.2 Experimental Results

We show three groups of comparisons here to illustrate the results of our experiments with respect to span (S), nuclearity (N), relation (R) and full (F).

**Comparison with the baseline model** Inspired by Li *et al.* [9] and Li *et al.* [10], we select the Tree-LSTM as our baseline model. This is done by manually setting $\lambda$ and $\gamma$ equal to zero, and other settings are consistent with R-IFGM. Then, we let the model to decide the value of $\lambda$ and $\gamma$ automatically. From the

**Table 1.** Performance comparison for different settings of our system.

| System Setting | S | N | R | F |
|---|---|---|---|---|
| Baseline | 67.9 | 55.3 | 43.5 | 43.2 |
| R-IFGM | 68.2 | 57.7 | 46.1 | 45.7 |

**Table 2.** Performance comparison for different approaches (borrowed from [12]).

| Parser | S | N | R | F |
|---|---|---|---|---|
| Feng and Hirst [5] | **68.6** | 55.9 | 45.8 | 44.6 |
| Ji and Eisenstein [7] | 64.1 | 54.2 | **46.8** | **46.3** |
| Joty *et al.* [8] | 65.1 | 55.5 | 45.1 | 44.3 |
| Hayashi *et al.* [6] | 65.1 | 54.6 | 44.7 | 44.1 |
| Braud *et al.* [4] | 59.5 | 47.2 | 34.7 | 34.3 |
| Li *et al.* [10] | 64.5 | 54.0 | 38.1 | 36.6 |
| Braud *et al.* [3] | 62.7 | 54.5 | 45.5 | 45.1 |
| Ours | 68.2 | **57.7** | 46.1 | 45.7 |

comparison in Table 1, the information flows of us is useful and the performance on nuclearity and relation indication is greatly improved by using R-IFGM to compute the representation of each text span. In particular, we analyze how the model assigns weights for these information flows in Section 4.3.

**Comparison with other systems** We compare our parser with seven parsers mentioned in [12] using the same evaluation metrics, and one can refer to the paper for more details. As shown in Table 2, we divide these systems into systems based on traditional methods and neural networks. From the results, systems in the first group are competitive with systems in the second group. Our parser is also based on neural network methods and it can strongly outperform other systems in nuclearity detection. The scores of ours in span and relation are relatively lower than the corresponding best results but better than most parsers. It is worth emphasizing that we exclude these complicated features others use in the aim to show the effect of the deep learning model itself.

In this work, we propose the R-IFGM model aiming to improve the model of Li *et al.* [9] and Li *et al.* [10]. Therefore, we compare with their works to further evaluate the performance of our method. Unfortunately, Moery *et al.* [12] replicate nine parsers in their paper except the parser of Li *et al.* [9]. Therefore, we use the same evaluation metrics used in [9] and [10] to examine the effect of our model, and the performance of their parsers is borrowed from published results. For fair comparison, we employ the same word embeddings as Li *et al.* [10], and the overall performance is reported in Table 3. From the results, our proposed model can outperform the two parsers in span, nuclearity and relation detection, which further proves the effectiveness of our model.

**Table 3.** Performance comparison with models of Li *et al.* [9] and Li *et al.* [10].

| Parser | S | N | R |
|---|---|---|---|
| Li *et al.* [9] | 82.4 | 69.2 | 56.8 |
| Li *et al.* [10] | 84.2 | 70.0 | 56.3 |
| Ours | 85.1 | 71.4 | 58.1 |

### 4.3 Analysis

Why the R-IFGM can help improve RST discourse parsing? To answer this question, we count the distribution of text spans' EDU numbers and find that about a quarter of text spans contain more than 7 EDUs. In addition, the longest document in RST-DT contains 304 EDUs and each document contains about 56 EDUs on average. In this condition, we give another statistic about the averaged proportion of each information flow with respect to the EDU number, as shown in Fig. 3. We assume that the more EDUs a text span has, the higher level of granularity the text span has. The figure shows that when the EDU number becomes larger, the proportion of structure information gets lower and the model will pay more attention to the state transition information and the text span's principal component.

General recursive deep models usually suffer from gradient vanishing. The baseline model of us works like a traditional Tree-LSTM model. It itself can store long distance information and thus effectively avoid gradient vanishing. To show the difference between the proposed R-IFGM and the baseline model, we give another statistic about the precision of span, nuclearity and relation detection for text spans with respect to the number of EDUs, as shown in Fig. 4. In the prediction of text spans with less than 5 EDUs, the superiority of the proposed R-IFGM is not obvious and it is even worse than the baseline model in span detection. The statistic shows that when the number of EDUs becomes

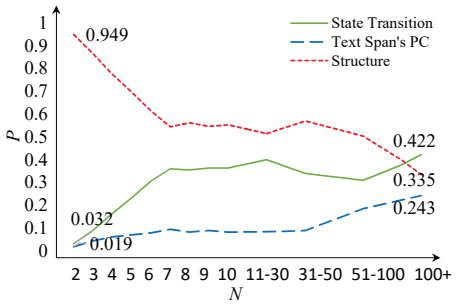


**Fig. 3.** The averaged proportion ($P$) of each information flow with respect to the EDU number ($N$) of each text span.

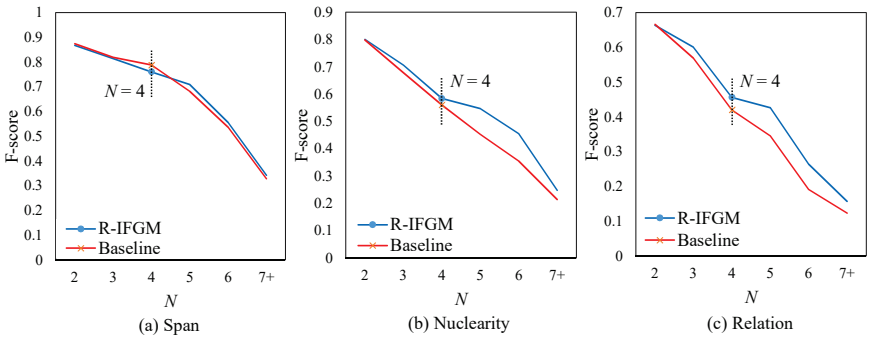


**Fig. 4.** The performance of our parser over text spans with different EDU numbers.

larger, the proposed R-IFGM performs relatively better. This superiority is evident in nuclearity and relation detection. From the statistic, we can draw an experimental conclusion that the proposed model has the ability to grade the information flows according to the granularity of each text span, and thus enable it to understand a document from different perspectives.

## 5 Related Work

Discourse parsing has largely concentrated on the RST-DT, which is the largest corpus of documents annotated with discourse structures. Early works on discourse parsing rely mainly on hand-crafted features [8, 7, 5]. Among these studies, SVM and variants of CRF models are mostly employed, and lexical, syntactic and semantic features are heavily used. With the increasing popularity of DNNs, there exist some neural network-based discourse parsers. Li *et al.* [9] firstly propose to build a recursive deep model in discourse parsing. Braud *et al.* [4] use a heuristic method to constrain their proposed seq2seq parser to build trees. Li *et al.* [10] build a CKY chart parser with the attention-based hierarchical Bi-LSTM. The parser proposed by Braud *et al.* [3] is a variant of the transition-based parser with a number of sophisticated features.

Inspired by Li *et al.* [9], we build a recursive deep model to generate representations for text spans. This recursive framework computes the representation for each parent based on its children recursively in a bottom-up fashion. However, recursive deep models are known to suffer from gradient vanishing. Li *et al.* [10] propose to employ an attention-based hierarchical Bi-LSTM to alleviate this problem. Nevertheless, it is hard for sequence LSTMs to provide discourse structure information. In this work, we absorb the advantages of both Li *et al.* [9] and Li *et al.* [10], and propose a TreeLSTM-style discourse parser. In particular, humans tend to understand macro and micro texts from different perspectives and the documents in RST-DT are known to have quite deep structures. Therefore, we introduce the R-IFGM to synthetically grade the information flows of us automatically according to the granularity of each text span.

## 6 Conclusion

We propose a transition-based discourse parser based on a recursive deep model. The R-IFGM of us can fuse multiple information flows according to the granularity of each text span to enrich the representation for text spans. Compared with previous works, our parser obtains competitive performance under stringent discourse parsing evaluations. Our future work will focus on extending text-level discourse parsing to related tasks like macro-sentiment analysis.

**Acknowledgements.** This work is supported by Artificial Intelligence Emergency Project 61751206 under the National Natural Science Foundation of China, and Project 61876118 under the National Natural Science Foundation of China.

## References

1. Arora, S., Liang, Y., Ma, T.: A simple but tough-to-beat baseline for sentence embeddings. In: Proceedings of ICLR (2016)
2. Bowman, S.R., Gauthier, J., Rastogi, A., Gupta, R., Manning, C.D., Potts, C.: A fast unified model for parsing and sentence understanding. arXiv preprint arXiv:1603.06021 (2016)
3. Braud, C., Coavoux, M., Søgaard, A.: Cross-lingual rst discourse parsing. arXiv preprint arXiv:1701.02946 (2017)
4. Braud, C., Plank, B., Søgaard, A.: Multi-view and multi-task training of rst discourse parsers. In: Proceedings of COLING 2016. pp. 1903–1913 (2016)
5. Feng, V.W., Hirst, G.: A linear-time bottom-up discourse parser with constraints and post-editing. In: Proceedings of the 52nd ACL. vol. 1, pp. 511–521 (2014)
6. Hayashi, K., Hirao, T., Nagata, M.: Empirical comparison of dependency conversions for rst discourse trees. In: Proceedings of the 17th SIGDIAL. pp. 128–136 (2016)
7. Ji, Y., Eisenstein, J.: Representation learning for text-level discourse parsing. In: Proceedings of the 52nd ACL. vol. 1, pp. 13–24 (2014)
8. Joty, S., Carenini, G., Ng, R.T.: Codra: A novel discriminative framework for rhetorical analysis. Computational Linguistics **41**(3), 385–435 (2015)
9. Li, J., Li, R., Hovy, E.: Recursive deep models for discourse parsing. In: Proceedings of EMNLP 2014. pp. 2061–2069 (2014)
10. Li, Q., Li, T., Chang, B.: Discourse parsing with attention-based hierarchical neural networks. In: Proceedings of EMNLP 2016. pp. 362–371 (2016)
11. Mann, W.C., Thompson, S.A.: Rhetorical structure theory: Toward a functional theory of text organization. Text-Interdisciplinary Journal for the Study of Discourse **8**(3), 243–281 (1988)
12. Morey, M., Muller, P., Asher, N.: A dependency perspective on rst discourse parsing and evaluation. Computational Linguistics pp. 198–235 (2018)
13. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)
14. Zhu, X., Sobihani, P., Guo, H.: Long short-term memory over recursive structures. In: ICML 2015. pp. 1604–1612 (2015)