A Neural Graph-based Local Coherence Model

Mohsen Mesgar and **Leonardo F. R. Ribeiro** and **Iryna Gurevych**
Ubiquitous Knowledge Processing Lab (UKP Lab)
Department of Computer Science
Technical University of Darmstadt
www.ukp.tu-darmstadt.de

Abstract

Entity grids and entity graphs are two frameworks for modeling local coherence. These frameworks represent entity relations between sentences and then extract features from such representations to encode coherence. The benefits of convolutional neural models for extracting informative features from entity grids have been recently studied. In this work, we study the benefits of Relational Graph Convolutional Networks (RGCN) to encode entity graphs for measuring local coherence. We evaluate our neural graph-based model for two benchmark coherence evaluation tasks: sentence ordering (SO) and summary coherence rating (SCR). The results show that our neural graph-based model consistently outperforms the neural grid-based model for both tasks. Our model performs competitively with a strong baseline coherence model, while our model uses 50% fewer parameters. Our work defines a new, efficient, and effective baseline for local coherence modeling[1].

1 Introduction

Local coherence is a discourse property that distinguishes a high-quality text from a random sequence of sentences. Modeling local coherence is crucial for various downstream NLP applications, e.g., summary evaluation and generation (Barzilay and Lapata, 2008; Parveen et al., 2016), readability assessment (Barzilay and Lapata, 2008; Mesgar and Strube, 2014), essay scoring (Burstein et al., 2010; Mesgar and Strube, 2016), dialogue evaluation and generation (Mesgar et al., 2020, 2021), and machine translation (Born et al., 2017; Kuang et al., 2018).

Motivated by the Centering theory (Joshi and Weinstein, 1981), many approaches to local coherence modeling rely on entity relations between sentences. The entity grid (Barzilay and Lapata, 2005, 2008) and the entity graph (Guinaudeau and Strube, 2013) are two well-studied frameworks for representing entity relations in a text. Entity grid-based models use grids while entity graph-based models use graphs to capture entity relations between sentences. Several methods have been proposed to enrich these representations and also to extract features from these representations to model local coherence. Recent work shows the effectiveness of convolutional neural networks (CNNs) for extracting features from entity grids to encode coherence (Tien Nguyen and Joty, 2017; Joty et al., 2018). Pre-trained transformer-based encoders can also capture relations between tokens in a text (Devlin et al., 2019). However, these encoders are potentially incapable of capturing long-distance relations (Martins et al., 2021), specifically where the text length is greater than the maximum input length in these encoders.

In this work, we revisit graph-based coherence assessment by introducing a neural graph-based coherence model. To do so, we represent a text via a graph (Figure 1) since a graph can capture long-distance relations in a text. Such a graph contains two types of edges: (1) Edges that capture entity-based relations between sentences, and (2) edges that capture the linear order of sentences in the text. To encode such graphs, we adapt *Relational Graph Convolutional Networks (RGCNs)* (Schlichtkrull et al., 2018). RGCNs encode nodes of a graph into vectors using the graph's connectivity structure and any feature information captured in the graph, such as edge types. We then apply a self-attention layer to these node vectors to capture to what extent each sentence of the text is crucial for estimating the coherence of the entire text. We finally use an output layer to transform the outputs of the self-attention layer to a score, which estimates the coherence degree of the text. Figure 2 depicts an overview of our model.

We evaluate our model for two benchmark co-

[1] https://github.com/UKPLab/emnlp2021-neural-graph-based-coherence-model

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 2316–2321
November 7–11, 2021. ©2021 Association for Computational Linguistics

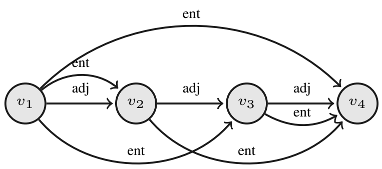
s_1: ***LDI** Crop., Cleveland, said it will offer \$50 million in commercial **paper** backed by lease-rental **receivables**.*

s_2: *The **program** matches **funds** raised from the **sale** of the commercial **paper** with small to medium-sized **leases**.*

s_3: ***LDI** leases and sells data-processing **telecommunications** and other high-tech **equipment**.*

s_4: ***LDI** termed the **paper** 'non-resource financing', meaning that **investors** would be repaid from the **lease** receivables, rather than directly by **LDI** Corp.*

(a)



(b)

Figure 1: A sample text in which entity mentions shown by bold (a), and its corresponding graph (b).

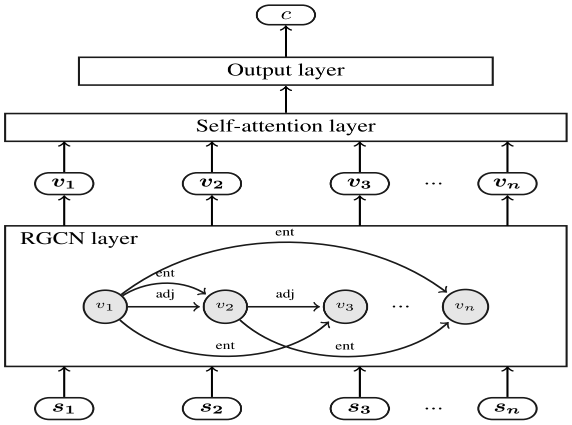


Figure 2: An overview of our coherence model.

herence evaluation tasks: (1) *Sentence Ordering (SO)* on the Wall Street Journal (WSJ) corpus, and (2) *Summary Coherence Rating (SCR)* on the Document Understanding Conference (DUC 2003) corpus. The results of our experiments confirm that our model consistently outperforms the neural grid-based coherence models (Tien Nguyen and Joty, 2017; Joty et al., 2018) by about 3.10% for SO and 1.2% for SCR. Our model performs on par with a recent coherence model (Moon et al., 2019), while our model uses 50% fewer parameters.

2 Method

2.1 Graph Representations

For a text as a sequence of sentences $T = (s_1, ..., s_n)$, we construct a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$ in which $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges, and $\mathcal{R}$ denotes the label set for edges (Figure 1). Each node $v_i \in \mathcal{V}$ is corresponded with a sentence s_i in the text T. We connect the nodes in a graph by two types of edges: (1) Edges with *"adj"* labels which connect nodes associated with any two adjacent sentences in the text to capture their linear order; and (2) Edges with *"ent"* labels which capture entity relations between sentences. We add an entity edge between nodes v_i and v_j if sentence s_i precedes sentence s_j and these sentences contain co-referring entity mentions. Edge directions capture the order of sentences. We use boldface notations for variables that refer to vectors or matrices.

2.2 Neural Graph-based Model

Our model consists of three layers (Figure 2): an RGCN, a self-attention, and an output layer.

RGCN As nodes in a graph represent sentences in a text, we first map sentences to vectors in an embedding space. Given sentence $s = (t_1, ..., t_{|s|})$ with $|s|$ tokens, we first map each token t to its corresponding embeddings $\boldsymbol{t}$. We then apply `BiLSTM` to embeddings of tokens to condition each token representation on the representations of its neighboring tokens in the sentence. :

$$\overrightarrow{\boldsymbol{H}}, \overleftarrow{\boldsymbol{H}} = \texttt{BiLSTM}\left([\boldsymbol{t_1}, \boldsymbol{t_2}, ..., \boldsymbol{t_{|s|}}]\right). \quad (1)$$

The reason that we use `BiLSTM` (instead of transformer-based encoders like BERT) is that we aim to keep our model's size in terms of the number of parameters efficient. We concatenate the output vectors associated with the last tokens in the left-to-right ($\overrightarrow{\boldsymbol{H}}$) and right-to-left ($\overleftarrow{\boldsymbol{H}}$) LSTM directions to obtain the sentence vector $\boldsymbol{s} = [\overrightarrow{\boldsymbol{H}}_{|\boldsymbol{s}|}; \overleftarrow{\boldsymbol{H}}_{|\boldsymbol{s}|}]$, where ";" is the concatenation function.

We adapt an RGCN layer to take these sentence vectors and enrich them with the graph structure of the text as well as edge types as follows:

$$\boldsymbol{v}_i = \sigma\Big(\sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_r(v_i)} \frac{1}{|\mathcal{N}_r(v_i)|} \boldsymbol{s}_j \boldsymbol{W}_r\Big), \quad (2)$$

where $\boldsymbol{W}_r \in \mathbb{R}^{d \times d}$ encodes the label $r \in \mathcal{R}$ between node v_j and v_i. The set $\mathcal{N}_r(v_i)$ contains the nodes connected to v_i by edges with label r.

Self-attention We use a multi-head self-attention (Vaswani et al., 2017) layer to estimate to what extent each sentence contributes to the coherence representation of a text. Each attention head computes a representation $\boldsymbol{z}_i$ of node vector $\boldsymbol{v_i}$ as follows:

$$\boldsymbol{z_i} = \sum_{j=1}^{n} \alpha_{ij} \left(\boldsymbol{v_j} \boldsymbol{W_a} \right), \tag{3}$$

where $\boldsymbol{W_a} \in \mathbb{R}^{d \times d}$ is learning parameters. We define attention weights α_{ij} as follows:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{n} \exp(e_{ik})} ,$$
$$e_{ij} = \frac{\left(\mathbf{W}_q \mathbf{v}_i\right)^{\top} \left(\mathbf{W}_k \mathbf{v}_j\right)}{d_s},$$

where e_{ij} is the attention function, and $\mathbf{W}_q, \mathbf{W}_k \in \mathbb{R}^{d \times d}$ are its parameters. d_s is the dimension of the input vectors. K independent attention heads are concatenated and linearly transformed to obtain final node representations, $\boldsymbol{v_i} = [\boldsymbol{z_i^{(1)}}; ...; \boldsymbol{z_i^{(K)}}]\boldsymbol{W_c}$.

Output layer We then apply a mean pooling to the output vectors of the attention layer to obtain a vector representing the coherence of the entire text. We map this vector to a score as follows:

$$c = (\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{v_i}) \boldsymbol{w_o} + b_o, \tag{4}$$

where $\boldsymbol{w_o} \in \mathbb{R}^d$ and $b_o \in \mathbb{R}$ are trainable parameters of the output layer. The output of the model c estimates the coherence degree of the entire text T.

2.3 Training and Evaluation

We train our model in a ranking scenario (Joty et al., 2018). Given T^+ as a text with a coherence degree higher than that of text T^-, we update parameters of our model with respect to the following loss function $\mathcal{L}(\Theta) = \max\{0, \tau - c^+ + c^-\}$, where c^+ and c^- are the coherence degrees our model estimates for text T^+ and text T^-, respectively. τ is the margin, Θ indicates all trainable parameters in our model. During training, our model shares all the layers to obtain c^+ and c^-. Once the model is trained for a task, we use it to score any text independently during evaluation for that task.

3 Experiments

We evaluate our model for two benchmark tasks for coherence modeling: *sentence ordering (SO)* and *summary coherence rating (SCR)*. In SO, a text is compared with random permutations of its sentences (Barzilay and Lapata, 2008). A coherence model should ideally rank a text higher than its permutations concerning coherence. In SCR, we deal with ranking summary texts, where each summary text comes with a coherence rating assigned by human judges (Barzilay and Lapata, 2008). Given a pair of summary texts with different coherence ratings, a coherence model is expected to rank them properly with respect to their coherence ratings.

	# Texts	**# Pairs**	**Avg. # Sent.**
Train	1240	23744	22.49
Dev	138	2678	18.85
Test	1053	20411	21.74

Table 1: Data splits used for sentence ordering.

Datasets For SO, we follow prior work (Moon et al., 2019; Joty et al., 2018; Tien Nguyen and Joty, 2017) and use the *Wall Street Journal (WSJ)* English news corpus. We use the same data splits and text permutations as used by Moon et al. (2019). Sections 00–13 of WSJ are used for training and sections 14–24 for testing (Table 1). We randomly select 10% of texts from the training set for development purposes. We compare any of these texts with 20 permutations.

For SCR, we use the dataset proposed by Barzilay and Lapata (2008) and used by prior work for coherence evaluation (Guinaudeau and Strube, 2013; Tien Nguyen and Joty, 2017). The dataset comprises texts from the DUC-2003 corpus, which contains English summaries produced by human experts and extractive summarization systems. Seven human annotators judged the summaries in a seven-point scale to rate how coherent the summaries were without having seen the source texts. For any summary in this dataset, the average of seven ratings, each assigned by a human judge, is taken as the coherence rating of the summary. Each data point in this dataset is a pair consisting of two summaries of the same text, where the rating of one of the summaries is higher than the rating of the other one. The training set contains 144 pairs, among which 14 pairs are used for development. The test set contains 80 pairs.

Settings We compare our model (Section 2) with the following coherence models: *EntGraph* (Guinaudeau and Strube, 2013), *Neural EntGrid* (Tien Nguyen and Joty, 2017), *Lex. Neural EntGrid* (Joty et al., 2018), and Moon et al. (2019). We use the source code of the model proposed by Moon et al. (2019) to reproduce their

Model	SO	SCR
EntGraph	80.00	80.0
Neural EntGrid	85.93	86.3
Lex. Neural EntGrid	88.51	-
Moon et al. (2019)	90.69	75.0
Ours	**92.41**	**87.5**

Table 2: Results in accuracy (%) for sentence ordering (SO) and summary coherence rating (SCR).

results on our machines. For others, we report the results from their papers. We use word2vec (Mikolov et al., 2013) as word embeddings since we aim to compare with *Lex. Neural EntGrid* in identical settings. Additionally, it keeps the number of parameters in our model low. We leave the study about the impact of different embeddings on the performance of our model for future work. We construct our graphs using the grids identical with those used by *Neural EntGrid* where all nouns are taken as entity mentions, and the string match approach is used to detect coreferent mentions. The batch sizes for training and evaluation is 5, τ is set to 5, and we train our model up to 5 epochs. The sizes of the word vectors, the BiLSTM and the RGCN layer are 300, 256 and 512, respectively. We optimize the parameters by Adam with a learning rate 0.0001 and L2 regularization. We use only one RGCN layer and one head for our attention. At each epoch we evaluate the model on the validation set. We use the model with the best scores on the validation set for evaluations on the test set. We run all experiments on a V100 GPU where each run of our model takes on average about 5 hours. We use *accuracy* as the evaluation metric, which corresponds to the number of correct rankings divided by the number of comparisons.

4 Results and Discussion

Table 2 shows the accuracy of the examined models for the SO and SCR tasks. Overall, our neural graph-based coherence model outperforms the examined baseline coherence models for both tasks.

Our model performs substantially better than *EntGraph*. Similar to *EntGraph*, we use graphs to represent relations between sentences. However, *EntGraph* relies on merely entity-based relations to construct graphs and uses a heuristically-defined feature (i.e., the average outdegree of nodes in a graph) to estimate the text coherence. Our model performs better because our graphs contain edges for capturing linear order of sentences as well as entity-based relations. Moreover, our model adapts RGCN to extract features for estimating coherence.

Our model also outperforms the examined entity grid-based models. The *Neural EntGrid* and *Lex. Neural EntGrid* models represent entity relations in text by entity grids and then apply CNNs to these grids to extract features for modeling the text coherence. Differently, our model uses graphs to represent relations between sentences and applies RGCN to learn features from graphs.

Our model slightly outperforms the model proposed by Moon et al. (2019). We note that the best results for M&M are 92.93 for SO and 83.8 for SCR, achieved with ELMo as word embeddings. We compare with their Word2Vec setting to study the influence of our models, not word embeddings. Moon et al. (2019)'s model uses no explicit representations of text structure (neither graphs nor grids). It captures linear relations between adjacent sentences using a neural bilinear layer, and their relations with a global representation of a text using a CNN-based module. This model is trained by a language model loss together with a ranking loss specifically designed for SO. Our model achieves scores similar to those of (Moon et al., 2019)'s model, while our model is simpler and smaller. We compare the number of our model's parameters with that of the (Moon et al., 2019)'s model for SO. For a fair comparison, we use identical settings for encoding sentences in both models. The number of our model's parameters ($\approx$ 5.0 M) is almost half of that in the (Moon et al., 2019)'s model ($\approx$ 9.5 M), indicating that our model compete with this model while using 50% fewer number of parameters.

Note that the *Neural EntGrid*'s score for SCR is its best performing results, where the model is first pretrained for SO and then fine-tuned on the training set of the SCR's dataset. Our model outperforms the *Neural EntGrid* model while our model is trained for SCR from scratch, i.e., without pretraining. It is worth noting that the size of the test split used for SCR is small (80 text pairs). The improvements achieved by our model translates into the fact that our model makes 10 and 6 out of 80 correct rankings more than what *Neural EntGrid* and the (Moon et al., 2019)'s model make, respectively. However, such improvements on the SCR's dataset are important as texts in this dataset are associated with human-provided coherence ratings.

Model	SO	SCR
Ours	**92.41**	**87.5**
Ours w/o ent.	91.89	85.0
Ours w/o adj.	90.05	87.5

Table 3: The impact of different edge types.

Table 3 depicts the accuracy of our model when different edge sets are used to construct graphs. "Ours w/o ent." shows our model trained on graphs with only adjacent edges. "Ours w/o adj." shows our model trained on graphs with only entity edges. We observe that edges with "adj" labels are more predictive signals than entity-based edges for SO. This observation intuitively makes sense as perturbations may change the order of only adjacent sentences. For SCR, entity-based relations are more predictive. Summary texts are supposed to express information about entities from source documents in a few sentences. Interestingly, by removing edges with "adj" labels, the performance of our model does not decrease for SCR. In sum, our model performs its best for both tasks when both edge types are used to construct graphs.

5 Conclusions

We introduced a neural graph-based model for local coherence assessment. We construct a graph of relations among sentences in a text using entity-based and linear relations between sentences. We apply relational graph convolutional networks to such graphs to extract features encoding coherence. Our model outperforms its counterparts for sentence ordering and summary coherence rating. The high performance of current coherence models on tasks with synthetic data possibly being not representative of real-life performance (Mohiuddin et al.). So, we aim to further study the performance of our model for tasks with natural data.

Acknowledgements

This work has been supported by the DFG-funded research training group "Adaptive Preparation of Information form Heterogeneous Sources" (AIPHES, GRK 1994/1) and the German Research Foundation through the German-Israeli Project Cooperation (DIP, grant DA 1600/1-1 and grant GU 798/17-1). We thank anonymous reviewers for their constructive suggestions.

References

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics,* Ann Arbor, Mich., 25–30 June 2005, pages 141–148.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Leo Born, Mohsen Mesgar, and Michael Strube. 2017. Using a graph-based coherence model in document-level machine translation. In *Proceedings of the Third Workshop on Discourse in Machine Translation,* Copenhagen, Denmark, 8 September, 2017, pages 26–35.

Jill Burstein, Joel Tetreault, and Slava Andreyev. 2010. Using entity-based features to model coherence in student essays. In *Proceedings of Human Language Technologies 2010: The Conference of the North American Chapter of the Association for Computational Linguistics,* Los Angeles, Cal., 2–4 June 2010, pages 681–684.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers),* Minneapolis, Minnesota, 2–7 June 2019, pages 4171–4186.

Camille Guinaudeau and Michael Strube. 2013. Graph-based local coherence modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Sofia, Bulgaria, 4–9 August 2013, pages 93–103.

Aravind K. Joshi and Scott Weinstein. 1981. Control of inference: Role of some aspects of discourse structure – centering. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence,* Vancouver, B.C., Canada, 24–28 August 1981, pages 385–387.

Shafiq Joty, Muhammad Tasnim Mohiuddin, and Dat Tien Nguyen. 2018. Coherence modeling of asynchronous conversations: A neural entity grid approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Melbourne, Australia, 15–20 July 2018, pages 558–568.

Shaohui Kuang, Deyi Xiong, Weihua Luo, and Guodong Zhou. 2018. Modeling coherence for neural machine translation with dynamic and topic caches. In *Proceedings of the 27th International Conference on Computational Linguistics,* Santa Fe, New Mexico, USA, 20–26 August 2018, pages 596–606.

Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. 2021. ∞-former: Infinite memory transformer. arXiv:2109.00301.

Mohsen Mesgar, Sebastian Bücker, and Iryna Gurevych. 2020. Dialogue coherence assessment without explicit dialogue act labels. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Online, 5-10 July, 2020, pages 1439–1450.

Mohsen Mesgar, Edwin Simpson, and Iryna Gurevych. 2021. Improving factual consistency between a response and persona facts. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume,* Online, 19–23 April 2021, pages 549–562.

Mohsen Mesgar and Michael Strube. 2014. Normalized entity graph for computing local coherence. In *Proceedings of TextGraphs-9: Graph-based Methods for Natural Language Processing, Workshop at EMNLP 2014,* Doha, Qatar, 29 October 2014, pages 1–5.

Mohsen Mesgar and Michael Strube. 2016. Lexical coherence graph modeling using word embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* San Diego, Cal., 12–17 June 2016, pages 1414–1423.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the ICLR 2013 Workshop Track.*

Tasnim Mohiuddin, Prathyusha Jwalapuram, Xiang Lin, and Shafiq Joty. Rethinking coherence modeling: Synthetic vs. downstream tasks. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume,* Online, 19–23 April 2021, pages 3528–3539.

Han Cheol Moon, Tasnim Mohiuddin, Shafiq Joty, and Chi Xu. 2019. A unified neural coherence model. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing,* Hong Kong, China, 3–7 November 2019, pages 2262–2272.

Daraksha Parveen, Mohsen Mesgar, and Michael Strube. 2016. Generating coherent summaries of scientific articles using coherence patterns. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing,* Austin, Tex., 1–5 November 2016, pages 772–783.

Michael S. Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of the Semantic Web - 15th International Conference, ESWC 2018,* Crete, Greece, 3–7 June, 2018, pages 593–607.

Dat Tien Nguyen and Shafiq Joty. 2017. A neural local coherence model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* Vancouver, Canada, 30 July–4 August 2017, pages 1320–1330.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *31st Conference on Neural Information Processing Systems (NIPS 2017),* Long Beach, CA., USA, 4–9 December 2017, pages 6000–6010.