Received: 28 May 2022 | Revised: 3 August 2023 | Accepted: 4 September 2023

DOI: 10.1111/exsy.13461

ORIGINAL ARTICLE

Expert Systems WILEY

CogNLG: Cognitive graph for KG-to-text generation

Peichao Lai[1] | Feiyang Ye[1] | Yanggeng Fu[1] | Zhiwei Chen[1] | Yingjie Wu[1] | Yilei Wang[1] | Victor Chang[2]

[1]College of Computer and Data Science, Fuzhou University, Fuzhou, China

[2]Department of Operations and Information Management, Aston Business School, Aston University, Birmingham, UK

Correspondence
Yilei Wang, College of Computer and Data Science, Fuzhou University, Fuzhou, China.
Email: yilei@fzu.edu.cn

Victor Chang, Department of Operations and Information Management, Aston Business School, Aston University, Birmingham, UK.
Email: victorchang.research@gmail.com and v.chang1@aston.ac.uk

Funding information
Natural Science Foundation of Fujian Province, PR China, Grant/Award Number: 2022J01120; Innovation Platform for Academician of Hainan Province, Grant/Award Number: YSPTZX202145; Fujian Province Industrial Guiding Project, Grant/Award Number: 2022H0012; Major Special Project for Industrial Science and Technology in Fujian Province, Grant/Award Number: 2022HZ022022; VC Research, Grant/Award Number: VCR 0000183

Abstract

Knowledge graph (KG) has been fully considered in natural language generation (NLG) tasks. A KG can help models generate controllable text and achieve better performance. However, most existing related approaches still lack explainability and scalability in large-scale knowledge reasoning. In this work, we propose a novel CogNLG framework for KG-to-text generation tasks. Our CogNLG is implemented based on the dual-process theory in cognitive science. It consists of two systems: one system acts as the analytic system for knowledge extraction, and another is the perceptual system for text generation by using existing knowledge. During text generation, CogNLG provides a visible and explainable reasoning path. Our framework shows excellent performance on all datasets and achieves a BLEU score of 36.7, which increases by 6.7 compared to the best competitor.

KEYWORDS
cognitive graph, KG-to-text, natural language generation

1 | INTRODUCTION

Language generation is an essential task in natural language processing (NLP), including tasks like machine translation, freeform question answering, dialog systems, (Choi et al., 2018; Dong et al., 2015; Peng et al., 2020) and so forth. Recently, pre-trained language models (PLMs) like GPTs (Brown et al., 2020; Radford et al., 2018; Radford et al., 2019), MASS (Song et al., 2019), BART (Lewis et al., 2020) and ProphetNet (Qi et al., 2020) have made significant progress in various natural language generation (NLG) tasks. However, most PLMs' superior performance is based on large model parameters and substantial training data. Although PLMs like GPT-2 (Radford et al., 2019) and GPT-3 (Brown et al., 2020) can generate correct syntactic content, they still make mistakes in specific areas, even commonsense reasoning. The reason for this phenomenon is that the models did not fully internalize all the knowledge during training. Moreover, it is unknowable what the model has learned because of the deep learning models' black-box nature.

Recent studies have demonstrated that knowledge graphs (KG) can effectively improve the performance of various NLP tasks. Knowledge graphs typically store a large amount of external knowledge through entity nodes and inter-entity relationships. It helps the model reduce the pressure of learning large-scale knowledge and focus on reasoning with existing knowledge. For example, in PLMs pre-training tasks like (Liu et al., 2020; Sun et al., 2019), KG effectively assists the model's learning ability and reduces large-scale pre-training costs. In named entity

recognition (NER) tasks, KGs like dictionaries help the model with weakly supervised learning without domain-specific annotated data (Lison et al., 2020). In NLG tasks, the explicit knowledge in KG effectively guides models to generate controlled, factual text (Koncel-Kedziorski et al., 2019). However, providing too much knowledge in KG-to-text tasks leads to the over-generation problem because that useless knowledge as noise will affect the model generation performance (Fu et al., 2020). Furthermore, it will be expensive to manually select accurate knowledge from a large-scale KG for the model training.

In general, the traditional KG-to-text models still face two main challenges: one is that most models lack explainability, making it difficult to find out where the issues come from when generating inappropriate content; and another challenge is that most existing approaches don't have scalability. When the provided knowledge contains errors or irrelevant information, the model's performance decreases badly. To address the above issues, we propose a cognitive graph framework called CogNLG for KG-to-text tasks insight by the dual-process theory (Evans, 1984; Evans, 2003; Evans, 2008; Sloman, 1996). The theory implies that human cognitive processing is divided into two systems: Systems 1 and 2. System 1 is a perceptual system that is intuitive, unconscious, and fast. The primary function of System 1 is to collect and retrieve intuitionistic information that humans perceive. System 2 is an analytic system for analyzing and reasoning the information provided by System 1. These two systems work together to form human cognition. Inspired by the dual-process theory, our CogNLG framework consists of Systems 1 and 2. Benefiting from the cognitive graph structure, the generating process of CogNLG is explainable, and our approach can filter out the accurate knowledge for the target text, which has scalability.

As shown in Figure 1, the cognitive graph for the KG-to-text task is constructed based on the input entities. Each input entity is initially defined as a source entity node. We add the new extension entity nodes by retrieving the association information of the existing entities through

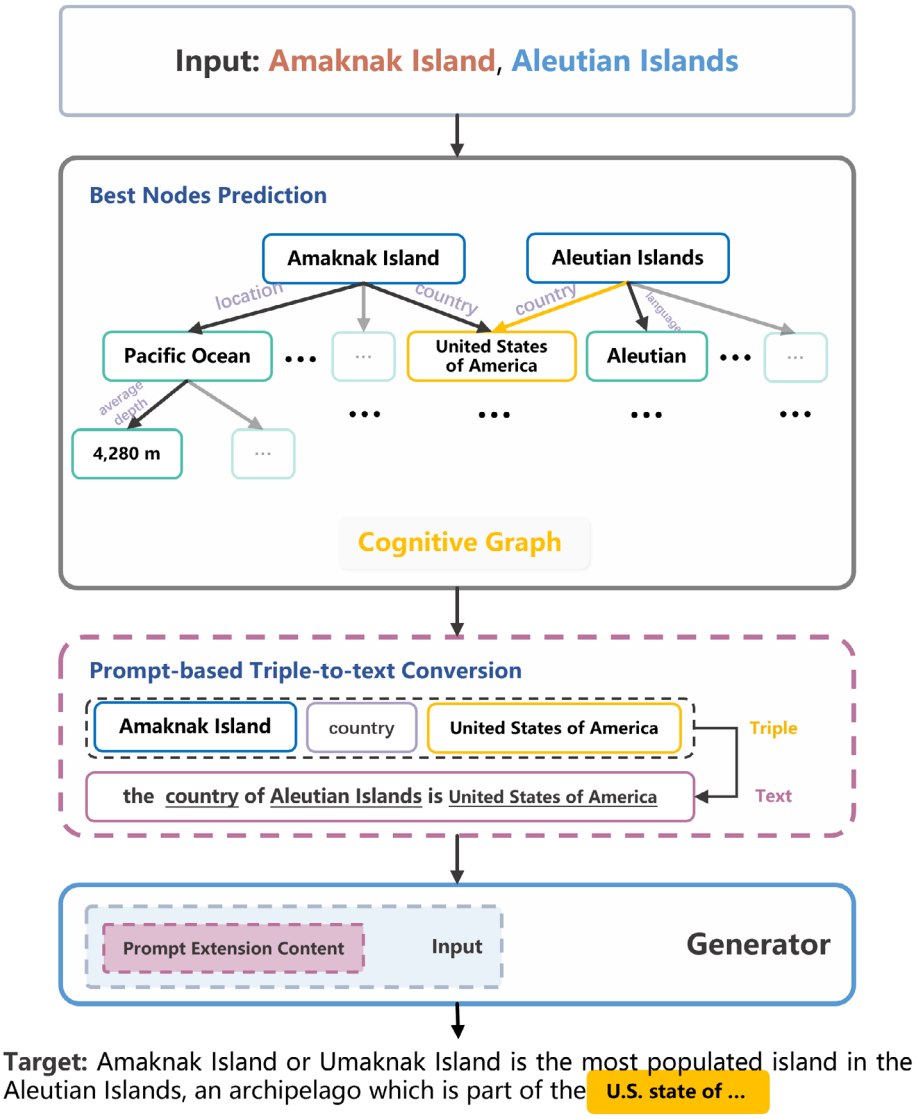


FIGURE 1 An example of the cognitive graph for KG-to-text generation, the circles in blue are the source entity, the circles in green are the extension entity, and the circle in gold is one of the predicted best nodes.

the wiki. Compared to the traditional approaches, which extract the entire KG, the CogNLG System 2 dynamically predicts the best nodes in each position and only extracts the valuable knowledge. According to the concept of text-to-text in the T5 model (Roberts et al., 2019), we use prompt-based templates to convert the best nodes triples to unstructured text. System 1 is a generator that predicts the next token based on the information provided by the input and the best nodes. In this work, we adopt the GPT-2 as System 1, and System 2 is an extractor implemented based on a graph convolutional neural network (GCN). To effect our implantation, our model evaluates two large-scale KG-to-text datasets called ENT-DESC (Cheng et al., 2020) and Person and Animal (Vrandečić & Krötzsch, 2014).

The contributions of this work are as follows:

1. We propose a novel CogNLG framework for KG-to-text generation tasks based on cognitive science. Moreover, experiment results show that the cognitive graph helps to generate controlled, factual text.
2. We demonstrate that the two-system structure of the cognitive graph provides strong explainability in the process of text generation and scalability in large-scale knowledge reasoning.
3. The performance of our implementation in multiple metrics on the ENT-DESC and the Person and Animal dataset surpasses the state-of-art work.

The structure of this paper is described as follows. Section 2 introduces some related work about KG-to-text tasks. Then the model and method details are presented in Section 3. Section 4presents the experimental results. Finally, the conclusion and the future work are presented in Section 5.

2 | RELATED WORK

2.1 | KG-to-text

A knowledge graph (KG)-to-text generation task is essential in NLG. In traditional NLG tasks, the rule-based systems depend on many handcrafted templates (Belz & Reiter, 2006; Duma & Klein, 2013), which is time-consuming and unscalable. In the deep learning approaches, the Sequential-to-sequential (seq2seq) (Mei et al., 2016; Wiseman et al., 2017) and Variational Autoencoders (VAEs) (Liu et al., 2019; Serban et al., 2017) models' performance have been dramatically improved compared with the rule-based systems. However, deep learning models cannot internalize all knowledge, and this problem also arises in deep pre-trained language models. The KG's primary function is to provide adequate knowledge support for NLP tasks to enhance the logic and model performance. In KG-to-text generation tasks, (Li et al., 2020) implements a KG-to-text model through the multi-attention mechanism, which encodes the input knowledge triples through a bidirectional GRU unit. Recently graph neural networks (GNN) have developed rapidly, and there are some works attempts to combine GNN and KG for text generation. (Koncel-Kedziorski et al., 2019) presents a graph transformer to encode graph structure input, and (Beck et al., 2018) proposes a gate graph neural network for the graph-to-text generation. Moreover, some works store knowledge with memory networks to improve performance on tasks like multiple-turn dialog (Madotto et al., 2018; Yang et al., 2019). (Zhu et al., 2020) proposes a fact-aware summarization model to ensure that the content generated by the model conforms to factual logic. The MGCN models (Cheng et al., 2020) adopt multiple graph transformations to obtain the context feature in different scales, which achieve great performance on KG-to-text tasks. (Chen et al., 2020) and (Ji et al., 2020) use PLM models with knowledge injection to generate the content with commonsense. However, most of the existing work still lacks the explainability of generated text and scalable reasoning in large-scale knowledge.

2.2 | Cognitive graph

The idea of the cognitive graph is based on cognitive science, and it was first proposed by (Ding et al., 2019) for multi-hop reading comprehension in the field of NLP. And (Dong et al., 2015) proposes the Cognitive Knowledge Graph reasoning framework for one-shot knowledge graph relation reasoning at scale. Both approaches have two systems: Perceptual System and Analytic System, and this is a significant feature of the cognitive graph compared with typical KGs. The two-system structure is inspired by the dual-process theory, and they run at inconsistent speeds. The perceptual system is more intuitive and unconscious, which means faster computing; the analytic system refers to human logical thinking, and it is more complex, rational, and slower. For complex problems, the two systems cooperate effectively to improve the performance in natural language understanding (NLU), and this approach is also known as fast and slow thinking (Kahneman, 2011). And the study of (Rastogi et al., 2020) has shown that fast and slow thinking can effectively assist Artificial Intelligence (AI) decision-making. Another feature of the cognitive graph is that it supports reasoning in the large-scale dataset. A cognitive graph framework requires scalability as the human brain can quickly retrieve information from large amounts of knowledge.

3 | METHODS

In this section, we describe the implementation of the CogNLG framework in detail. Based on the dual-process theory (Evans, 1984; Evans, 2003; Evans, 2008; Sloman, 1996), the human content creation process can be divided into two systems. One system is used for subjective language expression, and it is based on the human brain's accumulated prior linguistic knowledge. The other system retrieves relevant clues in real-time to support language generation.

The NLU ability in NLG tasks is also critical to model performance. The NLG task can be treated as the reverse work of the reasoning tasks. The analytic system selects the most appropriate information from the existing knowledge to assist the perceptual system. (Ji et al., 2020) adopts GPT-2 and GCN (Vashishth et al., 2020) to build a two-system text generation model and achieved significant performance in commonsense text generation. In this work, we show that the cognitive graph is also effective for KG-to-text tasks. We gain extension knowledge triples associated with the input through Wikipedia API and sift the best information with the analytic system in real-time. Our implementation can provide inference paths in large-scale data and reduce the cost of manually sifting knowledge triple.

Inspired by the dual-process theory, the CogNLG framework mainly consists of two systems called **generator system**($\mathbb{S}_1$) and **extractor system** ($\mathbb{S}_2$). $\mathbb{S}_1$ plays the role of a perceptual system and requires a large amount of prior linguistic knowledge. Therefore, a pre-trained model is necessary because it is trained intensively on large-scale corpora. $\mathbb{S}_2$ is an analytic system. In $\mathbb{S}_2$, we construct a cognitive graph to collect the supporting knowledge for text generation. Then we use a GNN model to update the hidden state of each node dynamically and iteratively in the cognitive graph and predict the best nodes to support $\mathbb{S}_1$.

The overall model structure is illustrated in Figure 2. We adopt the GPT-2 as $\mathbb{S}_1$ and a GCN model as $\mathbb{S}_2$. The structure and functionality of each system are described in detail below.

3.1 | Generator system

The core task of $\mathbb{S}_1$ is text generation. We adopt a pre-trained decoder model as the based model of $\mathbb{S}_1$. GPT-2 is a decoder model for text generation, which is widely adopted as a pre-trained model in various NLG tasks (Chen et al., 2020; Ji et al., 2020). The state-of-art GPT-3 (Brown et al., 2020) as a large language model (LLM) contains 175 billion parameters that cannot be trained on typical workstations. Besides, the issues of lacking explainability and generating inappropriate content also occur in GPT-3. GPT-2 has a much smaller number of parameters than GPT-3. In this task, our experimental results demonstrate that our approach can successfully address the above issues with GPT-2.

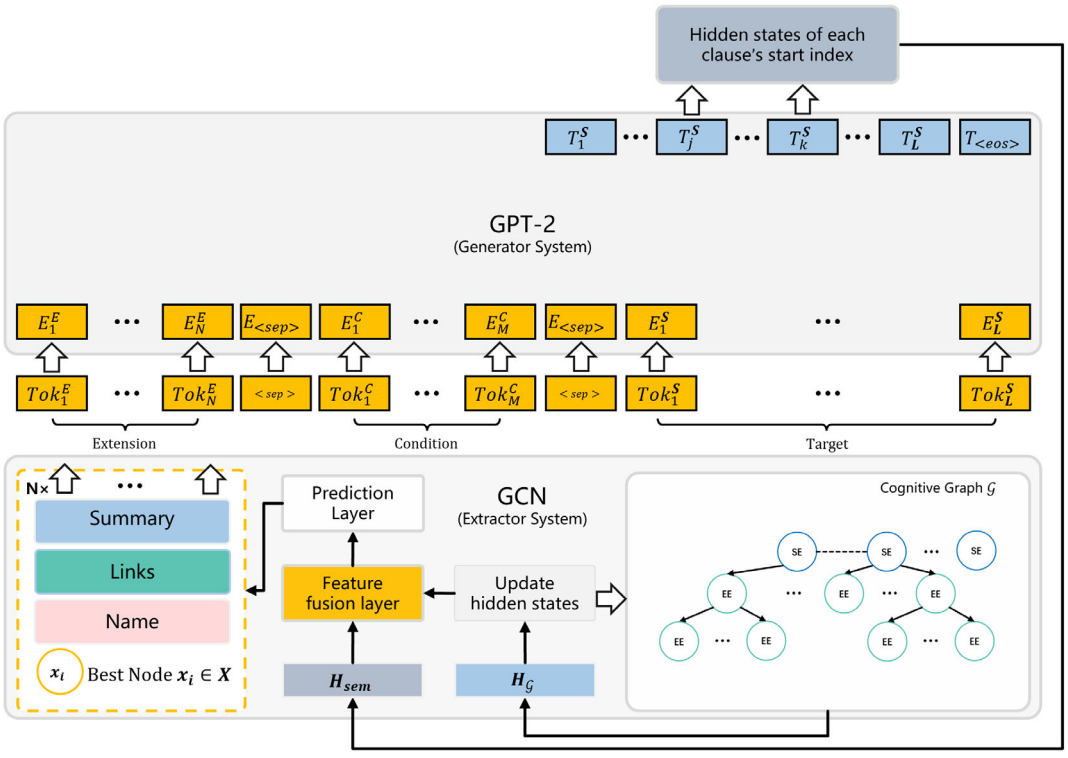


FIGURE 2 The overall model structure of the CogNLG framework.

The model needs accurate external knowledge support in generating sequence to generate the text that conforms to the factual logic. Unlike other knowledge-based approaches, which introduce static external knowledge in each iteration, our $\mathbb{S}_1$ dynamically takes the filtered external knowledge from the Extractor System in each position. The input of $\mathbb{S}_1$ is divided into three parts: the *Extension* ($\mathcal{E}_E$), the *Condition* ($\mathcal{E}_C$), and the *Target* ($\mathcal{E}_T$). Furthermore, the input format can be described as the following equation:

$$Extension < \text{SEP} > Condition < \text{SEP} > Target \tag{1}$$

where $\mathcal{E}_E = \{e_1,...,e_N\}$ is the concatenation of the best nodes summary, and the best nodes are predicted by $\mathbb{S}_2$ dynamically based on the current output content semantic in $\mathbb{S}_1$ and each node's hidden representation in $\mathbb{S}_2$; $\mathcal{E}_C = \{c_1,...,c_M\}$ is the concatenation of input entities; $\mathcal{E}_T = \{s_1,...,s_L\}$ is the ground truth target sequence, and the $< \text{SEP} >$ represents the special separator token.

Like common sequence generation tasks, the tokenizer encodes the input tokens and maps them to a high-dimensional vector through an embedding layer. In this task, the GPT-2 only outputs the hidden states within the $\mathcal{E}_T$ index range. The output hidden states in position i can be denoted as $T_i^S \in \mathbb{R}^{1\times H}$, where H is the dimension size of the hidden features. The hidden states T_i^S also represent the $\mathbb{S}_2$ semantic feature input H_{sem} in position i and be used for the next token prediction in evaluation.

3.2 | Extractor system

External knowledge plays a critical role in the $\mathbb{S}_1$ generation performance. Therefore, it is essential to select the most relevant nodes based on the current semantic feature of $\mathbb{S}_1$ and the hidden representations of $\mathbb{S}_2$. In particular, the external support knowledge should be closely related to the current generated content topic. Otherwise, the external knowledge would affect model performance as noise.

We construct a cognitive graph $\mathcal{G} = < \mathcal{V}, E >$ based on the input entities. There are two types of nodes in $\mathcal{G}$: Source Entity (*SE*) node and Extension Entity (*EE*) node. *SE* nodes are derived from the input entities; *EE* nodes are derived from the links of each parent node. $\mathcal{G}$ initially consists of multiple *SE* nodes, then denote a layer depth variable d. The *EE* nodes for each layer are then generated based on the links of the parent node. Algorithm 1 describes the construction procedure in detail. Each node $v_i \in \mathcal{V}$ contains the node name, links, and summary. The links array contains the related child nodes, and it is obtained by retrieving the node name from the wiki. The summary represents the text computed from the triple of the node and its parent. Previous work has demonstrated that the prompt-based input effectively improves the model's understanding of semantic information. We collect all the relation types from the datasets and design several templates according to their part of speech. As shown in Figure 3, the triples can be converted into summaries by the templates.

The initial hidden state of $\mathcal{G}$ is denoted as $H_{\mathcal{G}} \in \mathbb{R}^{H\times H}$, where H is the max size of $\mathcal{G}$. In our approach, we set H equal to the dimension size of the hidden features in $\mathbb{S}_1$. For each node, the hidden feature is computed by $\mathbb{S}_1$:

$$h^j = Trm^j(T_{\text{Node}}) \tag{2}$$

$$H_{\mathcal{G}}^k = h_{\mathscr{L}}^{\mathbb{N}} \tag{3}$$

where the k is the index of k-th node, and the input T_{Node} is the concatenation embeddings of node name and summary. The *Trm* of $\mathbb{S}_1$ is a transformer decoder with $\mathbb{N}$ layers, the k-th node initial hidden feature $H_{\mathcal{G}}^k$ is the last position $\mathscr{L}$ output in the last layer of the decoder. To predict the best nodes based on the context in each position, we first combine the semantic feature of each node with the relationship between neighboring nodes by adopting a variant GCN model. The hidden feature $H_{\mathcal{G}}$ is updated iteratively, and the new hidden state $H'_{\mathcal{G}}$ of one backpropagation step is computed as:

$$\delta = \sigma\left(\left(AD^{-1}\right)^T \sigma(H_{\mathcal{G}} W_\alpha)\right) \tag{4}$$

$$H'_{\mathcal{G}} = \sigma\left(H_{\mathcal{G}} W_\beta + \delta\right) \tag{5}$$

where σ is the activation function, A is the adjacent matrix of $\mathcal{G}$, which represents the relationship between nodes, D is a diagonal matrix and $D_{jj} = \sum_i A_{ij}$. $W_\alpha, W_\beta \in \mathbb{R}^{H\times H}$ are learnable weight matrices. The transformed hidden state $\sigma(H_{\mathcal{G}} W_\alpha)$ is left-multiplied by $\left(AD^{-1}\right)^T$ to extract the adjacent features δ of the graph. Then the hidden state of the next iteration can be updated by the summation of δ and $H_{\mathcal{G}} W_\beta$. When human writing, people prepare all the relevant knowledge beforehand and dynamically select the most relevant knowledge for the current context in real-time, which is a more efficient and scalable process. Similar to this process, $\mathbb{S}_2$ predicts the best nodes X based on the hidden feature $H_{\mathcal{G}}$ and the semantic feature H_{sem} computed by $\mathbb{S}_1$:

ALGORITHM 1 **Cognitive graph construction**

Input:
Generator System $\mathbb{S}_1$, Source Entity List $\mathcal{S}$, Wiki Database $\mathcal{W}$, Node Extension Depth d
Initialize cognitive graph $\mathcal{G}$, define a buffer list $\mathcal{B}$ and entity node list $\mathcal{N}$
initialize current depth c with 0
for s in $\mathcal{S}$ **do**
  append $<s, c>$ to $\mathcal{B}$
end for
repeat
  pop a tuple $<x, y>$ from $\mathcal{B}$
  fetch the name, summary, and links of x in $\mathcal{W}$
  create entity node $n(x)$, append it to $\mathcal{N}$
  generate hidden feature $H_{\mathcal{G}}^{x}$ with $\mathbb{S}_1$ and add it to $\mathcal{G}$
  set c to $c + 1$
  if *links*(x) is not empty **then**
    for l in *links*(x) **do**
      append$<l, c>$ to $\mathcal{B}$
      add *edge*(x, l) to $\mathcal{G}$
    end for
  end if
until $c \geq d$ or *len*$(\mathcal{N})$ is larger than the size of $\mathcal{G}$;
return $\mathcal{G}$

Triple	Template	Summary
Andrea Guerra | gender | male	the @r of @s is @o	the gender of Andrea Guerra is male
JACK fm | owned | Celador	@s is @r by @o	JACK fm is owned by Celador

FIGURE 3 The examples of triple-to-text conversion, where the text in red (@s) represents the subject, the text in green (@o) represents the object, and the text in blue (@r) represents the relation.

$$H_{fusion} = \sigma([H_{\mathcal{G}}; H_{sem}]W_{fusion}) \tag{6}$$

$$X = max[\sigma(H_{fusion}W_{cls})] \tag{7}$$

where H_{fusion} is the fusion feature of $H_{\mathcal{G}}$ and H_{sem}, $W_{fusion} \in \mathbb{R}^{2H \times H}$ is a learnable weight matrix. Finally $W_{cls} \in \mathbb{R}^{H \times 2}$ maps H_{fusion} to H-size two dimensional vectors, then we compute the indices of max value and choose the nodes with index one as the best nodes.

3.3 | Training implementation

A significant challenge in the training of the CogNLG framework is how to determine the best nodes for the current sequence. Most common related datasets contain only source input entities and ground truth target text. Besides, manually labeling the best nodes of a dataset is costly. This section proposes a general unsupervised best node labeling approach based on BLEU score and a pre-trained text similarity model to implement best node labeling on KG-to-text datasets.

For each input of the training set, let $Y = [w_1, ..., w_n]$ be the ground truth target text of input, where w is the word of Y, n is the length of Y. We split Y into subsets $\{y_1, ..., y_m\}$ based on pauses (commas, periods, semicolons, and some conjunctions). In the meantime, we construct the input's cognitive graph $\mathcal{G}$. And the best nodes in y_i are computed as:

$$X_{\text{best}}^{y_i} = \text{top-k}_{\mathcal{V} \in \mathcal{G}}\{\text{score}(y_i, \mathcal{V})\} \tag{8}$$

$$\underset{v_j \in \mathcal{V}}{e_j}(y_i, v_j) = BLEU(y_i, v_j) + \text{Sim}(y_i, v_j) \tag{9}$$

$$\text{score}_{v_j \in \mathcal{V}}(y_i, v_j) = \frac{\exp(e_j/\tau)}{\sum_k \exp(e_k/\tau)} \tag{10}$$

where the Sim function is implemented based on a BERT (Devlin et al., 2019) pre-trained model with a binary-class fully connected layer. We use the summation score of BLEU and Sim function to compute the correlation e_j between v_j and y_i. Since the difference between the correlation score and the actual best node distribution, we use a variant softmax with temperature τ to smooth the correlation score and compute the final similarity score. Then we select the top-k nodes $X_{best}^{y_i}$ with similarity score as the set of best nodes in y_i. And the collection of best nodes in Y are $[X_{best}^{y_1},...,X_{best}^{y_m}]$. It is important to note that in the evaluation step, the model predicts each position's best nodes $X_{best}^{[w_1,...,w_j]}$. However, during the training step, the time cost for each best node computing is expensive. Therefore, we consider that the best nodes in $y \in Y$ are consistent across the range of y and only compute the best nodes in the start index of y.

3.4 | Loss function

In $\mathbb{S}_1$, the final task is to generate the ground truth target text $Y = [w_1,...,w_n]$, suppose $w_t \in y_i$, where y_i is a subset of Y. The loss function of $\mathbb{S}_1$ is defined as follows.

$$\mathcal{L}_{gpt} = -\sum_{t=1}^{n+1} \log P(w_t | w_{<t}, X_{best}^{y_i}) \tag{11}$$

Where n is the length of Y, the model stops after the $<eos>$ token is generated. For best node prediction in $\mathbb{S}_2$, we compute the loss between probabilities and the fusion feature:

$$\mathcal{L}_{gcn} = -\sum_{t=1}^{n+1} \log P(\text{score}(y_i, \mathcal{V}) | w_{<t}, \mathcal{G}) \tag{12}$$

and the final loss can be optimized as:

$$\mathcal{L} = \mathcal{L}_{gpt} + \alpha \mathcal{L}_{gcn} \tag{13}$$

where α is a hyper-parameter, in this task, the value of α decreases during the training procedure, and the final loss is back-propagated to optimize both systems in CogNLG.

4 | EXPERIMENTS

4.1 | Dataset

ENT-DESC The ENT-DESC dataset (Cheng et al., 2020) is extracted from Wikipedia with more than 9.9 million pages. The dataset contains domains like humans, events, locations, etc. It consists of 110k instances and is significantly larger than related data-to-text datasets such as WebNLG (Gardent et al., 2017), AGENDA (Koncel-Kedziorski et al., 2019), and E2E (Novikova et al., 2017).

Person and animal The Person and Animal dataset (Vrandečić & Krötzsch, 2014) is extracted from structured KG WikiData and Wikipedia. There are two main types of entities in this dataset named Person and Animal. Compared with the ENT-DESC dataset, it contains only one entity with multiple references in each input. During our experiments, we only generate the first reference for better comparison.

We follow the same experiment setup of (Cheng et al., 2020), the dataset is randomly split into three subsets for the training set (80%), development set (10%), and test set (10%). Table 1 presents the statistics of the datasets. Each dataset item contains a list of source input entities, a ground truth target text, and the associated topic-related entities with the source input. In training and evaluating our CogNLG framework, we

TABLE 1 Statistics of datasets.

Dataset	Train	Dev	Test	Entities
ENT-DESC	88,651	11,081	11,081	657,554
Person and animal	352,782	44,101	44,101	440,984

only take the source input entities and the target text. Then we use the Wikipedia API to search for a 2-hop (the node extension depth d is set as two as it is large enough in this task) path associated with the input entities and save all path triples. The node name, summary and the related child nodes are extracted from the response node info of Wikipedia. We collect 620 million triples and store them in the database. Considering the time-consuming training, we construct a cognitive graph for each item in advance according to the method mentioned in Section 3.2. We use the default triples provided in the datasets for other related approaches.

4.2 | Baselines and metrics

To evaluate the performance of CogNLG comprehensively, we compare it with the state-of-the-art KG-to-text MGCN (Cheng et al., 2020) models and other GNN-based aggregation methods: GraphTransformer (Koncel-Kedziorski et al., 2019), GRN (Beck et al., 2018), GCN (Marcheggiani & Perez-Beltrachini, 2018) and DeepGCN (Guo et al., 2019). We also compare our model with transformer-based models: DSG (Fu et al., 2020) and KGPT (Chen et al., 2020), where KGPT is a decoder-based pre-trained language model with knowledge injection.

We evaluate CogNLG and other approaches on multiple evaluation metrics, including BLEU (Papineni et al., 2002), METEOR (Denkowski & Lavie, 2011), TER (Snover et al., 2006), ROUGE (Lin, 2004), and PARENT (Dhingra et al., 2019). Both the evaluation measures BLEU and ROUGE are based on n-gram analysis. BLEU measures text similarity to the reference based on n-gram overlap, while ROUGE further measures the longest common subsequence (LCS) between the generated and the reference text. METEOR considers word order and synonymy to evaluate the generated content quality. TER quantifies the edit distance between generated and reference text. PARENT combines TER with paragraph reuse for summary evaluation. By utilizing these metrics, we obtain objective measures of model performance, enabling comparisons with other approaches in the field of KG-to-text generation.

4.3 | Experimental details

We implement the CogNLG framework based on the Huggingface Transformers (Wolf et al., 2020). The pre-trained model of the GPT-2 in the $\mathbb{S}_1$ is "gpt2", released by (Radford et al., 2019). In this task, the number of transformer layers $\mathbb{N}$ is 12, and the hidden size H is 768; all the activation functions are *gelu* (Hendrycks & Gimpel, 2016); the value of k in Equation (8) is set as 6, and the temperature τ in Equation (10) is set to 0.2; the hyper-parameter α is initialized with 0.5 and linearly decreasing to 0.2 in 5000 steps. We optimize CogNLG with Adam (Kingma & Ba, 2017), the learning rate of $\mathbb{S}_1$ is 5×10^{-5} and the learning rate of $\mathbb{S}_2$ is 1×10^{-4}. To accelerate the convergence of the CogNLG, we further pre-trained vanilla GPT-2 with input entities and target text. The batch size of the pre-trained step is eight, and the batch size of CogNLG is set as one because its input length changes in real-time. During decoding, we use Nucleus Sampling (Holtzman et al., 2019) with top-8 tokens, which is more efficient than beam search.

4.4 | Results

The overall experimental results on the ENT-DESC and the Person and Animal datasets are shown in Tables 2 and 3. On the ENT-DESC dataset, our CogNLG framework is superior to all related approaches. Our model and KGPT (Chen et al., 2020) outperform the vanilla MGCN in all evaluation metrics. The BLEU score of CogNLG increases by 11.0 compared with the MGCN; this implies a significant improvement in performance for adopting the pre-trained model and two-system architecture. Compared to the MGCN ensemble models (MGCN + CNN + delex and MGCN + SUM + delex), our CogNLG outperforms them in BLEU and METEOR scores, but fell behind in ROUGE_L. It indicates that our model can generate content with more diversity and ensure fluency and accuracy.

TABLE 2 Comparison of different models on the ENT-DESC test set.

Models	BLEU	METEOR	TER↓	$ROUGE_1$	$ROUGE_2$	$ROUGE_L$	PARENT
GraphTransformer (Koncel-Kedziorski et al., 2019)	19.1	16.1	94.5	53.7	37.6	54.3	21.4
GRN (Beck et al., 2018)	24.4	18.9	70.8	54.1	38.3	55.5	21.3
GCN (Marcheggiani & Perez-Beltrachini, 2018)	24.8	19.3	70.4	54.9	39.1	56.2	21.8
DeepGCN (Guo et al., 2019)	24.9	19.3	70.2	55.0	39.3	56.2	21.8
DSG (Fu et al., 2020)	20.7	18.8	77.0	54.1	34.4	46.9	14.2
KGPT (Chen et al., 2020)	30.7	26.2	57.5	64.3	47.3	57.0	32.8
MGCN (Cheng et al., 2020)	25.7	19.8	69.3	55.8	40.0	57.0	23.5
MGCN+CNN+delex (Cheng et al., 2020)	29.6	23.7	63.2	63.0	46.7	**63.2**	31.9
MGCN+SUM+delex (Cheng et al., 2020)	30.0	23.7	67.4	62.6	46.3	62.7	31.5
CogNLG	**36.7**	**28.6**	**49.4**	**65.2**	**48.0**	58.9	**37.9**

Note: ↓ represents lower is better.

TABLE 3 Comparison of different models on the person and animal test set.

Models	BLEU	METEOR	TER↓	$ROUGE_1$	$ROUGE_2$	$ROUGE_L$	PARENT
GraphTransformer (Koncel-Kedziorski et al., 2019)	44.0	33.8	48.5	71.1	56.3	63.6	22.7
GRN (Beck et al., 2018)	18.5	17.0	69.5	48.5	25.4	42.4	17.3
GCN (Marcheggiani & Perez-Beltrachini, 2018)	18.1	16.7	69.9	48.0	24.9	42.0	16.2
DeepGCN (Guo et al., 2019)	21.2	18.9	67.1	51.4	28.8	45.3	21.5
DSG (Fu et al., 2020)	41.4	33.9	**46.3**	**72.4**	**59.4**	64.5	25.9
KGPT (Chen et al., 2020)	39.3	31.8	53.1	66.7	52.1	58.4	**46.8**
MGCN (Cheng et al., 2020)	18.2	16.8	70.2	51.8	29.6	45.7	22.7
MGCN+CNN+delex (Cheng et al., 2020)	21.8	19.3	66.7	51.8	29.6	45.7	22.7
MGCN+SUM+delex (Cheng et al., 2020)	20.3	18.4	67.8	50.6	27.8	44.5	20.4
CogNLG	**48.2**	**34.4**	49.1	69.9	57.2	**64.8**	24.6

Note: ↓ represents lower is better.

On the Person and Animal dataset, our model outperforms other approaches in BLEU, METEOR, and $ROUGE_L$. The result shows that MGCN performs better in complex graph structure scenarios with multiple input entities. Since the average number of entities in the Person and Animal dataset is much smaller than in the ENT-DESC dataset, MGCN performs worse than other related approaches. Our CogNLG still shows excellent performance on this dataset, and we can conclude that CogNLG has strong robustness on different entity structure datasets.

CogNLG has excellent scalability, which can filter knowledge noise, and it is a highlight compared with other traditional approaches. To analyze the scalability of CogNLG, we introduce triple noise for the ENT-DESC dataset. For each item in the ENT-DESC dataset, we randomly add the same number of triple noises as the origin triples. Triple noise refers to triples unrelated to the target text, leading to a severe decline in model performance when the model cannot effectively filter noise. Table 4 illustrates the performance comparison on the ENT-DESC test set with noise. The results show that all approaches performance decrease when introducing the triple noise. CogNLG shows great stability and still achieves the best performance compared with other approaches.

We further analyze the $\mathbb{S}_2$ knowledge extraction performance of CogNLG to evaluate its scalability in detail. The knowledge extraction performance can be treated as the best nodes prediction performance in Section 3.3. We introduce the F_1 score as the evaluation metric and evaluate the test set of the ENT-DESC and the Person and Animal. We set the max size of the cognitive graph equal to the hidden size H. Table 5 presents the results of the prediction performance in different hidden sizes. It is shown that $\mathbb{S}_2$ performance remains stable on both datasets when the hidden size changes. The results also explain why CogNLG remains stable on the ENT-DESC dataset with triple noise and demonstrate that CogNLG has outstanding scalability in large-scale dataset inference.

Over-generation is one potential reason for the model performance decrease caused by the triple noise. When the model can't effectively filter irrelevant entities, the generated text will contain irrelevant entity information. We count the proportion of irrelevant entities generated by different models under the original ENT-DESC dataset and the dataset with triple noise. As shown in Figure 4, the irrelevant entities proportion

TABLE 4 Comparison of different models on the ENT-DESC test set with random tripple noise.

Models	BLEU	METEOR	TER↓	$ROUGE_1$	$ROUGE_2$	$ROUGE_L$	PARENT
GraphTransformer (Koncel-Kedziorski et al., 2019)	14.3	16.0	128.4	51.7	31.9	39.1	8.1
GRN (Beck et al., 2018)	20.4	15.1	87.0	46.6	30.5	39.6	19.0
GCN (Marcheggiani & Perez-Beltrachini, 2018)	16.9	14.4	102.4	45.8	29.8	38.1	18.2
DeepGCN (Guo et al., 2019)	19.3	14.5	83.7	44.4	28.5	37.9	17.7
DSG (Fu et al., 2020)	12.4	11.9	77.0	41.8	25.7	38.5	13.5
KGPT (Chen et al., 2020)	11.1	8.3	90.1	30.8	19.3	27.2	24.2
MGCN (Cheng et al., 2020)	20.6	15.5	87.0	47.0	31.0	40.1	19.1
CogNLG	**28.1**	**24.2**	**65.0**	**58.6**	**39.7**	**52.4**	**28.4**

Note: ↓ represents lower is better.

TABLE 5 The performance of $\mathbb{S}_2$ on the ENT-DESC and the person and animal test set.

Datasets	Hidden size	F_1	Precision	Recall
ENT-DESC	$H = 768$	87.7	87.0	88.4
	$H = 1024$	86.4	85.3	87.6
Person and animal	$H = 768$	91.1	94.8	87.7
	$H = 1024$	88.6	95.2	82.8

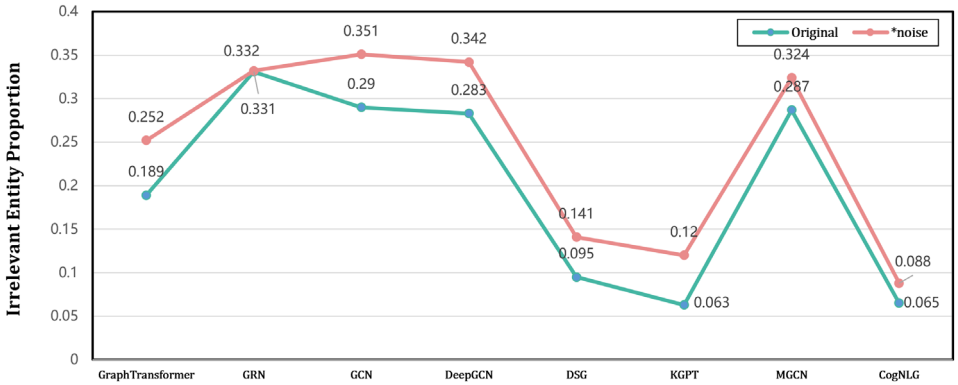


FIGURE 4 The irrelevant entities proportion of the generated text on the ENT-DESC dataset. "*noise" represents the value with triple noise.

increases in all models when introducing the triple noise. Due to the excellent performance of the $\mathbb{S}_2$, our CogNLG generated text contains the lowest irrelevant entity proportion, which implies it can effectively suppress the over-generation issue.

4.5 | Analysis and discussion

4.5.1 | Ablation study

We designed several experiments to analyze the performance impact of each component in CogNLG. To compare the results without external knowledge and only to use $\mathbb{S}_1$, we use the vanilla GPT-2 for training and evaluating with input entities and the ground truth target text. As shown in Table 6, the BLEU score of GPT-2 decreased by 12.5 compared with CogNLG, which is worse than most graph-based approaches in Table 2. The results prove that external knowledge plays a vital role in the performance of KG-to-text tasks.

The CogNLG-R is designed with the same structure as the CogNLG framework but disables the $\mathbb{S}_2$ predictor and replaces it with a random selector for best nodes selection. The CogNLG-R performs poorly and even worse than the vanilla GPT-2. It implies that inaccurate knowledge would reduce the model performance as noise, and CogNLG $\mathbb{S}_2$ contributes to filtering accurate knowledge for $\mathbb{S}_1$.

TABLE 6 Results of the ablation study on the ENT-DESC test set.

Models	BLEU	METEOR	TER↓	$ROUGE_1$	$ROUGE_2$	$ROUGE_L$	PARENT
GPT-2	24.2	20.5	83.1	53.2	35.2	46.6	23.3
CogNLG-R	23	20.9	69.7	51.7	33.0	45.7	21.4
CogNLG-T	35.5	28.3	**49.1**	64.8	47.1	58.1	37.1
CogNLG-O	**37.3**	28.4	57.7	63.6	47.7	58.3	32.3
CogNLG	36.7	**28.6**	49.4	**65.2**	**48.0**	**58.9**	**37.9**

Note: ↓ represents lower is better.

To analyze the impact of the triple-to-text policy, the CogNLG-T takes the original triples as $\mathcal{E}_E$ input in $\mathbb{S}_1$. The performance of CogNLG-T decreased slightly compared to CogNLG, indicating that the transformation from triple to text can effectively improve the model's internalization of the triple relational.

The CogNLG-O represents the model performance training with the original triples from the ENT-DESC dataset instead of using the triples from the Wiki Database. As shown in Table 6, the performance of CogNLG-O is similar to CogNLG.

4.5.2 | Explainability analysis

To verify the performance of CogNLG explainability, we present some cases of how the CogNLG $\mathbb{S}_2$ is reasoning the best nodes. As shown in Figure 5, each case consists of a simplified cognitive graph at the top and the generated text at the bottom. In the beginning, the $\mathcal{E}_E$ and $\mathcal{E}_T$ in $\mathbb{S}_1$ are missing, and $\mathbb{S}_2$ predicts the best node based on the semantics of $\mathcal{E}_E$. The best nodes prediction at the beginning of the generated sentence are shown in cases (a) and (b). We observed that the relation of best nodes in both cases are related to the sentence's subject. In case (a), the subject is a person, and the best nodes include "gender", "birthday", "country", "occupation", and so forth. The subject of case (b) is an airplane base, and the best nodes include the "instance of the subject", "established date", "country", and so forth. In case (c), when the model encounters the preposition "in", $\mathbb{S}_2$ successfully predicts the best node "Villorsonnens" required by the next token. We observe that the best nodes usually remain constant within a clause. However, when it comes to "is" or prepositions like "in", "at", the best nodes change significantly. Case (d) illustrated that $\mathbb{S}_2$ successfully deduces the relationship between "Villorsonnens" and "Fribourg", which reflects reasoning ability to the depth information of the cognitive graph.

We also present the relation tag of the top three best nodes prediction results for each token in Table 7. The predicted relation tags of best nodes have a high correlation with the current token, making it easy to forecast what token the model will generate in the next position. The visual analysis of best nodes prediction demonstrates that CogNLG shows explainability ability in NLG tasks. It can be concluded that $\mathbb{S}_2$ provides all nodes that it considers nodes to have a high probability of being generated in the following text to prevent information omission. $\mathbb{S}_1$ then decides what to generate based on the context. In particular, $\mathbb{S}_1$'s selection of knowledge is influenced by the semantic distribution of the training set.

4.5.3 | Case study

We study example outputs by GCN, MGCN+SUM, and CogNLG to compare the efficiency of knowledge extraction. The results are shown in Figure 6, where GCN and MGCN+SUM use the KG provided by the ENT-DESC dataset. The highlighted text in red represents the main input entity, and the highlighted text in blue represents the topic-related entities. The first row in Figure 6 is the gold reference. GCN fails to generate the main entity, which means it cannot extract knowledge accurately. The CogNLG $\mathbb{S}_2$ is implemented based on a variant GCN with aggregate semantic information from $\mathbb{S}_1$. It successfully filters out accurate knowledge by binary classification based on the current context and node relational information. Compared to the output text generated by MGCN+SUM, CogNLG describes related entities more accurately. It further demonstrates that the CogNLG excellent performance benefits from the two-system structure, making $\mathbb{S}_1$ focus on organizing language expression and $\mathbb{S}_2$ focus on knowledge extraction.

4.5.4 | Error study

We visualize the CogNLG evaluation BLEU − 4 and $ROUGE_L$ scores distribution on the ENT-DESC test set. The results are illustrated in Figure 7. Most cases show a Gaussian distribution near the average scores in Table 2. However, it can be observed a bipolar distribution in many cases (b) and (c) display the average target reference length and best node labels in each $ROUGE_L$ score range, and they both have a Gaussian

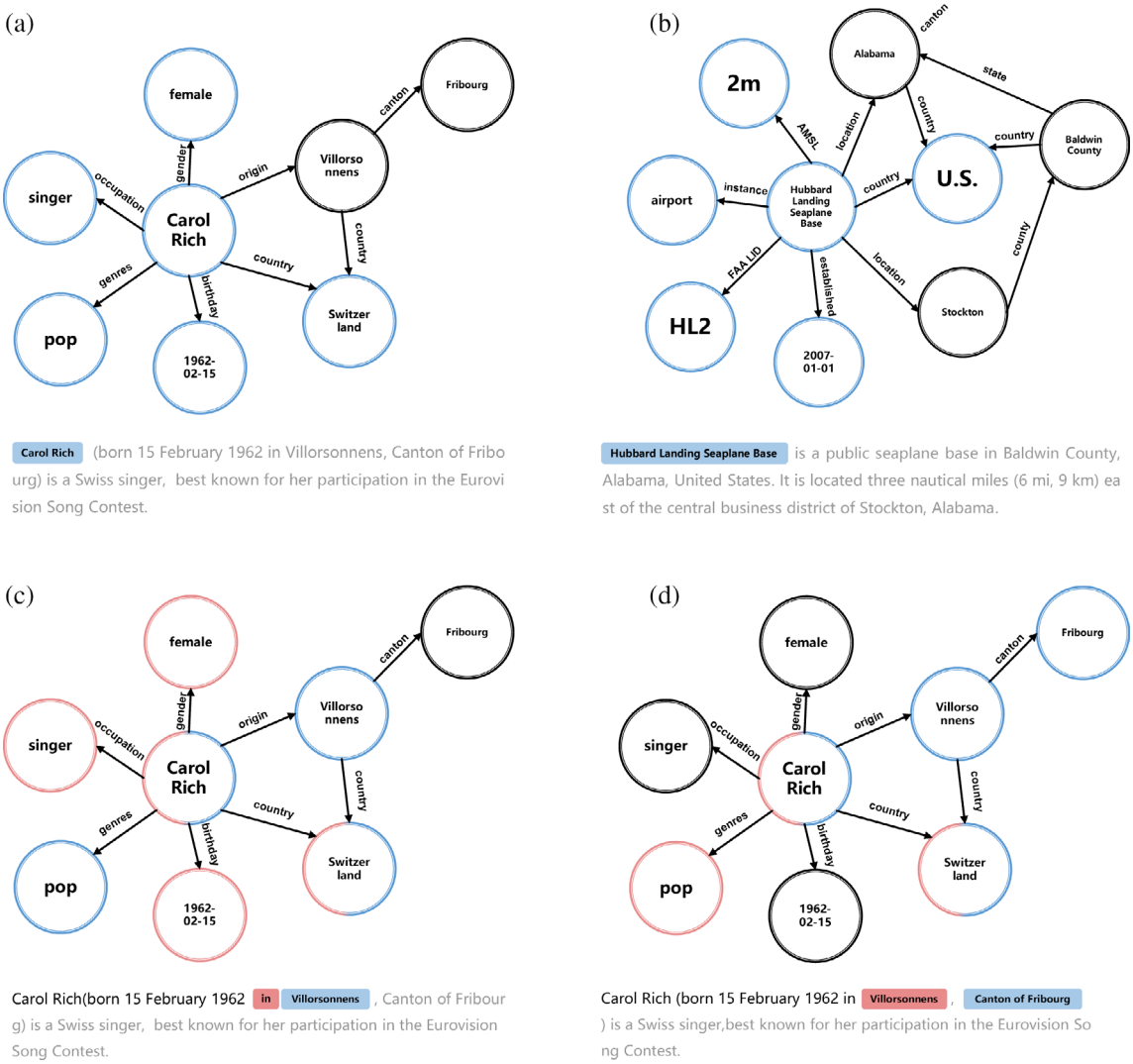


FIGURE 5 Cases of cognitive graph reasoning path. The text in red is the current last position token, and the blue is the next prediction token. The red circles are the best node for the current token, and the blue circles are the best node for the next token (red and blue overlapping circles represent the best nodes for both).

TABLE 7 Case results on predicting the top three best nodes for each token.

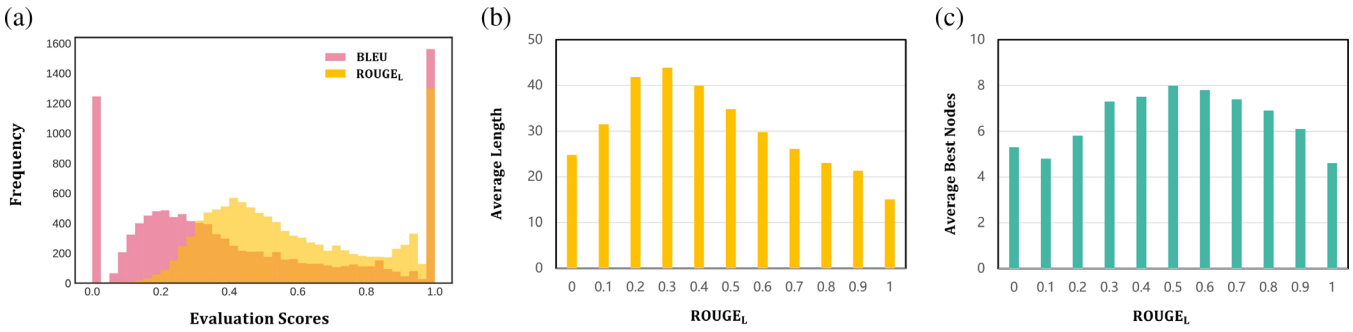
Dataset	
ENT-DESC	Silva Dimmo (*Person*)→[*Birthday*,*Gender*,*Occupation*]
	Milton (*Location*)→[*Instance*,*Location*,*Country*]
	the University of Science and Technology (*Organization*)→[*Instance*,*Location*,*Country*]
Person and animal	Frederic Alderson (*Person*)→[*Birthday*,*Occupation*,*Institution*]
	the African penguin (*Animal*)→[*Taxon*,*Instance*,*Location*]
	Alaska (*Location*)→[*Instance*,*Location*,*Country*]

distribution. The reasons for the bipolar distribution cases in (a) are shorter text length and fewer best node labels. We find that the generated outputs get high scores in short texts if the reference contains enough topic-related entity information. On the contrary, the generated outputs get low scores if the reference is mainly composed of verbs, adjectives, and so forth, and lacks topic-related nouns.

We also observed that some cases get low scores due to entity missing or mismatching. The external knowledge is obtained from the Wikipedia API, and we use fuzzy matching to retrieve entity association information. Therefore, it inevitably leads to some missing or mismatched issues. To study the consequences of missing entities, we selected an instance and manually removed the birthday and country from the graph. As shown in Table 8, when CogNLG misses the birthday and country, it still generates a fake birthday and country. Moreover, we observed that

Gold	The New Jersey Symphony Orchestra is an American symphony orchestra based in the state of New Jersey. The NJSO is the state orchestra of New Jersey, performing concert series in six venues across the state, and is the resident orchestra of the New Jersey Performing Arts Center in Newark, New Jersey.
GCN	The Newark Philharmonic Orchestra is an American orchestra based in Newark, New Jersey, United States.
MGCN+SUM	The New Jersey Symphony Orchestra is an American chamber orchestra based in Newark, New Jersey. The orchestra performs at the Newark Symphony Center at the Newark Symphony Center in Newark, New Jersey.
CogNLG	The New Jersey Symphony Orchestra is an American professional orchestra based in New Jersey. The orchestra's primary concert venue is the New Jersey Performing Arts Center in Newark, New Jersey.

FIGURE 6 Comparison of different models on an example.



Evaluation scores distribution.

Average target reference length in each $ROUGE_L$ score range.

Average best nodes in each $ROUGE_L$ score range.

FIGURE 7 Data distribution visualization on the ENT-DESC test set.

TABLE 8 An example of CogNLG generation with knowledge missing.

Gold	**Lee Aaron (born Karen Lynn Greening, July 21, 1962) is a Canadian rock singer.**
Missing birthday and country	Lee Aaron (born June 21, 1969) is an American rock musician.
+birthday	Lee Aaron (born July 21, 1962) is an American rock musician.
+birthday	
+country	Lee Aaron (born July 21, 1962) is a Canadian rock singer, songwriter, and musician.

the birthday was randomly different each time the model generates, but the country was the same. The result is similar to using vanilla GPT-2 without any external knowledge, that is, the generated sentences are grammatically smooth but may not follow factual logic.

A deep end-to-end model cannot control whether it generates related entities randomly or fixedly because it is unknown what knowledge is internalized. The two-system-based CogNLG helps us locate the wrong entities by observing the cognitive graph. After adding the missing entities, the model generates the relevant text accurately.

4.5.5 | Diversity analysis

To evaluate the diversity of the model, we randomly sort the input entities and make three predictions on each test set. We adopt the Self-BLEU (Zhu et al., 2018) score to compute the diversity of the generated text. As shown in Table 9, CogNLG achieves the lowest Self-BLEU score on

TABLE 9 Self-BLEU scores of different models on the test set.

Model/Dataset	ENT-DESC	Person and animal
GraphTransformer	52.02	69.09
GRN	75.25	67.57
GCN	79.20	68.40
DeepGCN	72.49	67.12
DSG	39.25	53.32
KGPT	67.42	51.60
MGCN	73.72	65.71
CogNLG	**33.26**	**49.97**

both datasets, indicating that the generated text of CogNLG is more diverse. We believe it is due to the design of the two system structures, which makes the model $\mathbb{S}_1$ to adapt to the information provided by the $\mathbb{S}_2$ with increasing generation diversity during training.

5 | CONCLUSION

This paper proposes a CogNLG framework for KG-to-text tasks and obtains the state-of-art results on the ENT-DESC and the Person and Animal dataset. The implementation is based on cognitive science. Our CogNLG can reason at the large-scale dataset and has great explainability and scalability with the two-system structure. We demonstrate that the cognitive graph can be applied to NLG tasks with outstanding performance. In the future, we will work to improve the training efficiency of CogNLG and extend it to domain-specific multimodal generation tasks.

ACKNOWLEDGMENTS

Prof Wang's work is supported by the Natural Science Foundation of Fujian Province, PR China (2022J01120); the Innovation Platform for Academician of Hainan Province (YSPTZX202145); Fujian Province Industrial Guiding Project (2022H0012); Major Special Project for Industrial Science and Technology in Fujian Province (2022HZ022022). Prof Chang's work is partly supported by VC Research (VCR 0000183).

DATA AVAILABILITY STATEMENT

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

ORCID

Yanggeng Fu https://orcid.org/0000-0002-8507-9189
Victor Chang https://orcid.org/0000-0002-8012-5852

REFERENCES

Beck, D., Haffari, G., & Cohn, T. (2018). Graph-to-sequence learning using gated graph neural networks. Proceedings of the 56th annual meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 273–283.

Belz, A., & Reiter, E. (2006). Comparing automatic and human evaluation of nlg systems. Paper presented at: 11th conference of the European chapter of the association for computational linguistics.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. arXiv preprint arXiv:2005.14165.

Chen, W., Su, Y., Yan, X., & Wang, W. Y. (2020). Kgpt: Knowledge-grounded pre-training for data-to-text generation. arXiv preprint arXiv:2010.02307.

Cheng, L., Wu, D., Bing, L., Zhang, Y., Jie, Z., Lu, W., & Si, L. (2020). Ent-desc: Entity description generation by exploring knowledge graph. arXiv preprint arXiv:2004.14813.

Choi, E., He, H., Iyyer, M., Yatskar, M., Yih, W.-t., Choi, Y., Liang, P., & Zettlemoyer, L. (2018). Quac: Question answering in context. Proceedings of the 2018 conference on empirical methods in natural language processing. pp. 2174–2184.

Denkowski, M., & Lavie, A. (2011). Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. Proceedings of the sixth workshop on statistical machine translation. pp. 85–91.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

Dhingra, B., Faruqui, M., Parikh, A., Chang, M.-W., Das, D., & Cohen, W. (2019). Handling divergent reference texts when evaluating table-to-text generation. Proceedings of the 57th annual meeting of the Association for Computational Linguistics. pp. 4884–4895.

Ding, M., Zhou, C., Chen, Q., Yang, H., & Tang, J. (2019). Cognitive graph for multi-hop reading comprehension at scale. Proceedings of the 57th annual meeting of the Association for Computational Linguistics. pp. 2694–2703.

Dong, D., Wu, H., He, W., Yu, D., & Wang, H. (2015). Multi-task learning for multiple language translation. Proceedings of the 53rd annual meeting of the Association for Computational Linguistics and the 7th international joint conference on natural language processing (Volume 1: Long Papers). pp. 1723–1732.

Duma, D., & Klein, E. (2013). Generating natural language from linked data: Unsupervised template extraction. Proceedings of the 10th international conference on computational semantics (IWCS 2013)–long papers. pp. 83–94.

Evans, J. S. B. (1984). Heuristic and analytic processes in reasoning. *British Journal of Psychology*, *75*(4), 451–468.

Evans, J. S. B. (2003). In two minds: Dual-process accounts of reasoning. *Trends in Cognitive Sciences*, *7*(10), 454–459.

Evans, J. S. B. (2008). Dual-processing accounts of reasoning, judgment, and social cognition. *Annual Review of Psychology*, *59*, 255–278.

Fu, Z., Shi, B., Lam, W., Bing, L., & Liu, Z. (2020). Partially-aligned data-to-text generation with distant supervision. Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP). pp. 9183–9193.

Gardent, C., Shimorina, A., Narayan, S., & Perez-Beltrachini, L. (2017). The webnlg challenge: Generating text from rdf data. Proceedings of the 10th international conference on natural language generation. pp. 124–133.

Guo, Z., Zhang, Y., Teng, Z., & Lu, W. (2019). Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, *7*, 297–312.

Hendrycks, D., & Gimpel, K. (2016). Bridging nonlinearities and stochastic regularizers with Gaussian error linear units.

Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2019). The curious case of neural text degeneration. International conference on learning representations.

Ji, H., Ke, P., Huang, S., Wei, F., Zhu, X., & Huang, M. (2020). Language generation with multi-hop reasoning on commonsense knowledge graph. Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP). pp. 725–736.

Kahneman, D. (2011). *Thinking, fast and slow*. Macmillan.

Kingma, D. P., & Ba, J. (2017). Adam: A method for stochastic optimization.

Koncel-Kedziorski, R., Bekal, D., Luan, Y., Lapata, M., & Hajishirzi, H. (2019). Text generation from knowledge graphs with graph transformers. 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). pp. 2284–2293.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2020). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. Proceedings of the 58th annual meeting of the Association for Computational Linguistics. pp. 7871–7880.

Li, W., Peng, R., Wang, Y., & Yan, Z. (2020). Knowledge graph based natural language generation with adapted pointer-generator networks. *Neurocomputing*, *382*, 174–187.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. Text summarization branches out. pp. 74–81.

Lison, P., Barnes, J., Hubin, A., & Touileb, S. (2020). Named entity recognition without labelled data: A weak supervision approach. Proceedings of the 58th annual meeting of the Association for Computational Linguistics.

Liu, W., Zhou, P., Zhao, Z., Wang, Z., Ju, Q., Deng, H., & Wang, P. (2020). K-bert: Enabling language representation with knowledge graph. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*, 2901–2908.

Liu, Z., Fu, Z., Cao, J., de Melo, G., Tam, Y.-C., Niu, C., & Zhou, J. (2019). Rhetorically controlled encoder-decoder for modern Chinese poetry generation. Proceedings of the 57th annual meeting of the Association for Computational Linguistics. pp. 1992–2001.

Madotto, A., Wu, C.-S., & Fung, P. (2018). Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. Proceedings of the 56th annual meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1468–1478.

Marcheggiani, D., & Perez-Beltrachini, L. (2018). Deep graph convolutional encoders for structured data to text generation. Proceedings of the 11th international conference on natural language generation. pp. 1–9.

Mei, H., UChicago, T., Bansal, M., & Walter, M. R. (2016). What to talk about and how? Selective generation using lstms with coarse-to-fine alignment. Proceedings of NAACL-HLT. pp. 720–730.

Novikova, J., Dušek, O., & Rieser, V. (2017). The e2e dataset: New challenges for end-to-end generation. Proceedings of the 18th annual SIGdial meeting on discourse and dialogue. pp. 201–206.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: A method for automatic evaluation of machine translation. Proceedings of the 40th annual meeting of the Association for Computational Linguistics. pp. 311–318.

Peng, B., Zhu, C., Li, C., Li, X., Li, J., Zeng, M., & Gao, J. (2020). Few-shot natural language generation for task-oriented dialog. Proceedings of the 2020 conference on empirical methods in natural language processing: Findings. pp. 172–182.

Qi, W., Yan, Y., Gong, Y., Liu, D., Duan, N., Chen, J., Zhang, R., & Zhou, M. (2020). Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. Proceedings of the 2020 conference on empirical methods in natural language processing: Findings. pp. 2401–2410.

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, *1*(8), 9.

Rastogi, C., Zhang, Y., Wei, D., Varshney, K. R., Dhurandhar, A., & Tomsett, R. (2020). Deciding fast and slow: The role of cognitive biases in ai-assisted decision-making.

Roberts, A., Raffel, C., Lee, K., Matena, M., Shazeer, N., Liu, P. J., Narang, S., Li, W., & Zhou, Y. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer.

Serban, I., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A., & Bengio, Y. (2017). A hierarchical latent variable encoder-decoder model for generating dialogues. Proceedings of the AAAI conference on artificial intelligence. Vol. 31.

Sloman, S. A. (1996). The empirical case for two systems of reasoning. *Psychological Bulletin*, *119*(1), 3.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., & Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. Proceedings of the 7th conference of the Association for Machine Translation in the Americas: Technical papers. pp. 223–231.

Song, K., Tan, X., Qin, T., Lu, J., & Liu, T.-Y. (2019). Mass: Masked sequence to sequence pre-training for language generation. Icml.

Sun, Y., Wang, S., Li, Y., Feng, S., Chen, X., Zhang, H., Tian, X., Zhu, D., Tian, H., & Wu, H. (2019). Ernie: Enhanced representation through knowledge integration. arXiv preprint arXiv:1904.09223.

Vashishth, S., Sanyal, S., Nitin, V., & Talukdar, P. (2020). Composition-based multi-relational graph convolutional networks.

Vrandečić, D., & Krötzsch, M. (2014). Wikidata: A free collaborative knowledgebase. *Communications of the ACM*, *57*(10), 78–85. https://doi.org/10.1145/2629489

Wiseman, S., Shieber, S. M., & Rush, A. M. (2017). Challenges in data-to-document generation. Emnlp.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., … Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations. Association for Computational Linguistics, pp. 38–45. https://www.aclweb.org/anthology/2020.emnlp-demos.6

Yang, P., Li, L., Luo, F., Liu, T., & Sun, X. (2019). Enhancing topic-to-essay generation with external commonsense knowledge. Proceedings of the 57th annual meeting of the Association for Computational Linguistics. pp. 2002–2012.

Zhu, C., Hinthorn, W., Xu, R., Zeng, Q., Zeng, M., Huang, X., & Jiang, M. (2020). Boosting factual correctness of abstractive summarization with knowledge graph. arXiv e-prints, arXiv–2003.

Zhu, Y., Lu, S., Zheng, L., Guo, J., Zhang, W., Wang, J., & Yu, Y. (2018). Texygen: A benchmarking platform for text generation models. The 41st international ACM SIGIR conference on research & development in information retrieval. pp. 1097–1100.

AUTHOR BIOGRAPHIES

Peichao Lai is a Ph.D. candidate in the College of Computer and Data Science at Fuzhou University, Fuzhou, China. He received the B.S. degree in Software Engineering from Fuzhou University in 2019. His primary areas of focus encompass the exciting domains of named entity recognition, natural language generation, and natural language inference. In these domains, he is dedicated to exploring and contributing to the latest advancements, aiming to make a meaningful impact on the field of computer science. One of his distinct areas of dedication lies in the study of few-shot, semi-supervised, and transfer learning methodologies. In an era where data-driven approaches are paramount, he is keenly interested in developing and refining techniques that can effectively harness limited data resources.

Feiyang Ye is a Master's candidate at Fuzhou University's College of Computer and Data Science. He holds a Bachelor of Engineering (B.E.) degree in Computer Science and Technology from Fuzhou University. His primary research interests include Named Entity Recognition (NER) and Natural Language Generation (NLG). In the field of NER, he focuses on developing advanced algorithms and models to improve the accuracy of entity recognition in text data. In the domain of NLG, his research centers on computer-generated natural language text, exploring techniques to enhance the quality and fluency of machine-generated content. He is dedicated to advancing these areas within the field of Natural Language Processing.

Yanggeng Fu received the Ph.D. degree from Fuzhou University, Fuzhou, China, in 2013. He is a full professor and a doctoral supervisor with the College of Computer and Data Science, Fuzhou University. He has published more than 50 international refereed journal and conference papers in his areas of interest. His teaching specialization and research interests include data structures and algorithms, data mining, machine learning, and intelligent decision support systems. Additionally, Dr. Fu serves as the Head of the Institute of Computer with Application at Fuzhou University, where he maintains strong connections with industry. He has conducted more than 10 projects in the past few years, all relating to his areas of expertise.

Zhiwei Chen is a Master's graduate in the College of Computer and Data Science at Fuzhou University. He received the B.S. degree in Computer Science and Technology from Fuzhou University. His primary research interests encompass natural language inference and natural language generation. He is dedicated to leveraging external knowledge bases to enhance model performance across various tasks, with a particular focus on Chinese. Considering the specific traits of the Chinese language, he utilizes Graph Neural Networks to extract features that boost the model's performance.

Yingjie Wu is a professor in the College of Computer and Data Science at Fuzhou University. He received Master's and Ph.D. degrees from Fuzhou University and Southeast University. He presided over the research work of one national natural science foundation project, two Fujian natural science foundation projects and one class A science and technology project of the Fujian Provincial Department of Education. He has published over 30 academic papers. His main research interests include differential privacy protection, object detection, and information extraction.

Yilei Wang is an associate professor in the College of Computer and Data Science at Fuzhou University. Her academic journey is marked by a distinguished educational background, as she earned both her Master's and Ph.D. degrees from Southeast University and Fuzhou University, respectively. Her primary research areas encompass recommendation algorithms, data mining, and natural language inference, where she has made valuable contributions and conducted pioneering work. In addition to her research endeavors, she has actively participated in various research projects and has received funding from both provincial and national sources. She has served as the principal investigator for two projects funded by the Fujian Provincial Natural Science Foundation and has been a key contributor to two projects supported by the National Natural Science Foundation of China.

Victor Chang is a Professor of Business Analytics at Operations and Information Management, Aston Business School, Aston University UK, since mid-May 2022. He was previously a Professor of Data Science and Information Systems at the School of Computing, Engineering and Digital Technologies, Teesside University, UK, between September 2019 and mid-May 2022. He has deep knowledge and extensive experience in AI-oriented Data Science and has significant contributions in multiple disciplines. Within 4 years, Prof Chang completed PhD. (CS, Southampton) and PG Cert (Higher Education, Fellow, Greenwich) while working for several projects simultaneously. Before becoming an academic, he has achieved 97% on average in 27 IT certifications. He won 2001 full Scholarship, a European Award on Cloud Migration in 2011, IEEE Outstanding Service Award in 2015, best papers in 2012, 2015 and 2018, the 2016 European award: Best Project in Research, 2016-2018 SEID Excellent Scholar, Suzhou, China, Outstanding Young Scientist award in 2017, 2017 special award on Data Science, 2017-2022 INSTICC Service Awards, Talent Award Suzhou 2019, Top 2% Scientist 2017/2018, 2019/2020 & 2020/2021, the most productive AI- based Data Analytics Scientist between 2010 and 2019, Highly Cited Researcher 2021 and numerous awards mainly since 2011. Prof Chang was involved in different projects worth more than £14 million in Europe and Asia. He has published 3 books as sole authors and the editor of 2 books on Cloud Computing and related technologies. He published 1 book on web development, 1 book on mobile app and 1 book on Neo4j. He gave 40 keynotes at international conferences. He is widely regarded as one of the most active and influential young scientist and expert in IoT/Data Science/Cloud/security/AI/IS, as he has the experience to develop 10 different services for multiple disciplines. He is the founding conference chair for IoTBDS, COMPLEXIS and FEMIB to build up and foster active research communities globally with positive impacts.

How to cite this article: Lai, P., Ye, F., Fu, Y., Chen, Z., Wu, Y., Wang, Y., & Chang, V. (2023). CogNLG: Cognitive graph for KG-to-text generation. *Expert Systems*, e13461. https://doi.org/10.1111/exsy.13461