Online-Journals.org

iTDAF IETI Transactions on Data Analysis and Forecasting

iTDAF | eISSN: 2959-0442 | Vol. 2 No. 1 (2024) | OPEN ACCESS

https://doi.org/10.3991/itdaf.v2i1.40095

PAPER

Knowledge Inference Combining Convolutional Feature Extraction and Path Semantics Integration

Chen Xinyuan[1](✉), **Ubaldo Comite**[2]

[1]Fuzhou Technology and Business University, Fuzhou, China

[2]University Giustino Fortunato, Benevento, Italy

516040610@qq.com

ABSTRACT

Many knowledge representation models extract local patterns or semantic features using fact embeddings but often overlook path semantics. There is room for improvement in path-based approaches that rely solely on single paths. A customized convolutional neural network (CNN) architecture is proposed to encode multiple paths generated by random walks into vector sequences. For each path, the feature sequence is then merged into a single vector using bidirectional long short-term memory (LSTM) by concatenating both forward and backward hidden states. Semantic relevance between different paths and candidate relations is computed using the attention mechanism. The state vectors of the relations are calculated using weighted paths. These paths help determine the probabilities of the candidate relations, which are then used to assess the validity of the triples. Link prediction experiments on two benchmark datasets, NELL995 and FB15k-237, demonstrate the advantages of our solution. Our model shows a 7.19% improvement at Hits@3 on FB15k-237 compared to Att-Model + Type, another advanced model. The model is further applied to a large complex dataset, FC17, as well as a sparse dataset, NELL-One, for few-shot reasoning.

KEYWORDS

knowledge graph (KB), knowledge inference, embedding representation, path semantics, convolutional neural network (CNN), long short-term memory (LSTM), attention mechanism

NOMENCLATURE

BLSTM	Bidirectional LSTM
CEC	Constant Error Carousel
CNN	Convolutional Neural Network
CV	Computer Vision
GRU	Gated Recurrent Unit
KB	Knowledge Base
KG	Knowledge Graph
KGC	Knowledge Graph Completion
LSTM	Long Short-Term Memory
MAP	Mean Average Precision

Xinyuan, C., Comite, U. (2024). Knowledge Inference Combining Convolutional Feature Extraction and Path Semantics Integration. *IETI Transactions on Data Analysis and Forecasting (iTDAF)*, 2(1), pp. 48–62. https://doi.org/10.3991/itdaf.v2i1.40095

Article submitted 2023-04-02. Revision uploaded 2024-01-12. Final acceptance 2024-01-12.

MRR	Mean Reciprocal Rank
NLP	Natural Language Processing
PRA	Path Ranking Algorithm
RL	Reinforcement Learning
RNN	Recurrent Neural Network
2D	Two-Dimensional

1 INTRODUCTION

Knowledge base (KB) [1] organizes facts in the form of triples, which consist of entities and relations. Mainstream KBs such as NELL [2], YAGO [3], and Freebase [4] are extensively utilized in various fields such as information retrieval [5], question answering [6–7], personalized recommendation [8], and more.

Existing KBs are incomplete, i.e., there are entities or relations missing in lots of fact triples [9], which impairs the performance of downstream tasks. Knowledge graph completion (KGC) [10–11] aims to solve this problem by extracting local patterns or semantic features using knowledge embedding to generate new facts based on known information [12–13]. The core concepts, key issues, common techniques, and future directions are discussed, analyzed, and summarized in references [14–16]. Translation or rotation-based distance and tensor or matrix factorization-based semantic similarity gain prevalence among mainstream approaches [17], with TransE [18] and DistMult [19] as representatives, respectively. ConvE [20] utilizes convolutional neural networks (CNNs) to facilitate interactions between entities or relations and enhance feature extraction.

However, most approaches ignore the semantic information conveyed by relational paths between entity pairs, which could aid in determine the validity of triples [21–22]. Neelakantan et al. [23] and Das et al. [24] introduced recurrent neural network (RNN) for path embedding. Since ordinary RNNs may not learn semantic dependencies across long distances, Hochreiter et al. [25] proposed long short-term memory (LSTM), which computes information that should be forgotten or updated using a gated structure. The attention mechanism [26] is widely applied in computer vision (CV) and natural language processing (NLP) tasks [27–29]. On the basis of TransE, Xiong et al. [30] employed the reinforcement learning (RL) framework to encode agents into continuous spaces. This approach combines the advantages of distance models and path models, i.e., taking into account both local structure and semantic correlations.

This paper proposes an integrated framework named CLAP (knowledge inference with *C*NN, *L*STM, and *a*ttention mechanism) for improved local feature extraction and path semantics recognition. Paths are integrated with different weights using the attention mechanism.

The main work includes: (1) Designing a customized CNN framework to encode full paths into vector sequences; (2) Employing bidirectional LSTM (BLSTM) to merge each path into a single vector; (3) Introducing the attention mechanism to assign different weights based on the correlations between the candidate relations and paths, integrating which is used to calculate the probabilities of the candidates; (4) Conducting experiments to compare CLAP on benchmark datasets with baselines.

2 RELATED WORK

In KGC, embedding models aim to learn low-dimensional representations of entities and relations while preserving the original structural patterns and semantic constraints. SE [31] computes the dot products of relational matrices with head or tail entities, which is computationally expensive. Subsequent models strive to strike a balance between complexity and performance.

TransE maps relations as translational vectors and posits that if one triple holds, the head vector after translation should be close to the tail vector, i.e., $\boldsymbol{v}_h + \boldsymbol{v}_r \approx \boldsymbol{v}_t$, where $\boldsymbol{v}_h$, $\boldsymbol{v}_r$, $\boldsymbol{v}_t$ are the embeddings of entities and relations. Local features of triples are preserved in the same dimension of entity or relation vectors. TransH [32] proposes relation-specific hyperplanes, $\boldsymbol{w}_r$ while TransR [33] further replaces them $\boldsymbol{w}_r$ with mapping matrices $\boldsymbol{W}_r$. Both approaches aim to reflect the role differences of entities under various relations; however, they come with higher complexities.

Among the similarity models, RESCAL calculates triple scores by factorizing of third-order adjacency tensors. DistMult represents relations as diagonal matrices to simplify computations. ComplEx [34] further introduces the complex space for knowledge embedding.

In recent years, CNN has been introduced into NLP [35] with fewer parameters and lower computational overhead than fully connected networks. In ConvE, entities and relations $\boldsymbol{v}_h$, $\boldsymbol{v}_r$ are concatenated, reshaped, and then input into a convolutional layer. Feature tensors are vectorized by filters and computed $\boldsymbol{v}_t$ for triple scores. The two-dimensional (2D) convolution could enhance interactions between entities and relations [36].

Most models above only consider direct correlations and ignore the semantics passed down relational paths [37–38]. Lao et al. [21–22] used the depth first random walk algorithm to generate paths. Das et al. [39] propose Minerva, which considers historical paths in knowledge graph (KG) traversal. Luo et al. [40] combined relational paths with TransE. However, such studies consider paths as atomic features, resulting in large feature matrices and high computational costs [41–42].

Recurrent neural networks (RNNs) were was originally designed to process sequential data and have achieved success in fields such as speech and video recognition. Neelakantan et al. [23] proposed path-RNN, which decomposes paths into relational sequences and inputs them into an RNN. Paths with the highest scores are selected to complete missing triples. Parameter sharing within the same layer reduces computations. However, there may be multiple paths associated with candidate relations simultaneously, and a single path may not provide sufficient information. Das et al. [24] integrated multi-path semantics with Mean or LogSumExp operations, which ignore the differences in semantic correlations.

Ordinary RNNs are plagued by the gradient vanishing problem and have difficulty learning long-distance semantic dependencies. LSTM introduces a gated structure to control information flow, and there are numerous variants [43–44].

Zhang et al. [45] argue that integrating path information is essential for knowledge representation and reasoning, particularly in complex scenarios. Xiong et al. [46] argue that the continuous growth and sparsity of knowledge bases (KBs) necessitate few-shot, one-shot, and even zero-shot reasoning capabilities, where auxiliary information such as path semantics is beneficial. Related studies include references [47] and [48].

Recently, the attention mechanism has been widely applied to NLP tasks [49]. Bahdanau et al. [50] and Vaswani et al. [51] designed machine translation decoders

with such mechanisms. Jiang et al. [28] proposed an attentive knowledge reasoning solution that assigns different weights to paths based on their semantic relevance. Nathani et al. [52] employed the mechanism to extract neighbor information in graphs for relational clustering. The attention mechanism is not adept at processing long sequences either. Zhou et al. [27] proposed an integrated model, Att-BLSTM, in which BLSTM [53] is used to generate sentence embeddings with word embeddings. Scores are then computed using the attention mechanism for relational classification.

Since CNN-based approaches and path-based methods have different strengths, this paper proposes CLAP as an integrated solution for improved embedding feature extraction and semantic utilization.

3 FRAMEWORK OF CLAP

The framework of CLAP is shown in Figure 1. The code is publicly available at https://github.com/ch9t/CLAP. For a given entity pair and candidate relations, a customized CNN is used to encode multiple paths obtained by random walks between the entities, considering their relational composition. Paths with variable lengths are mapped to vector sequences with the same lengths while retaining local structures. For each path, the feature sequence is then merged into a single vector using BLSTM, concatenating both forward and backward hidden states. The path vectors are equivalent to the sentence embeddings [27]. Semantic relevance between different paths and candidate relations is computed using the attention mechanism. The state vectors of the relations are calculated using weighted paths. These paths help determine the probabilities of the relations, which are then used to assess the validity of the triples.

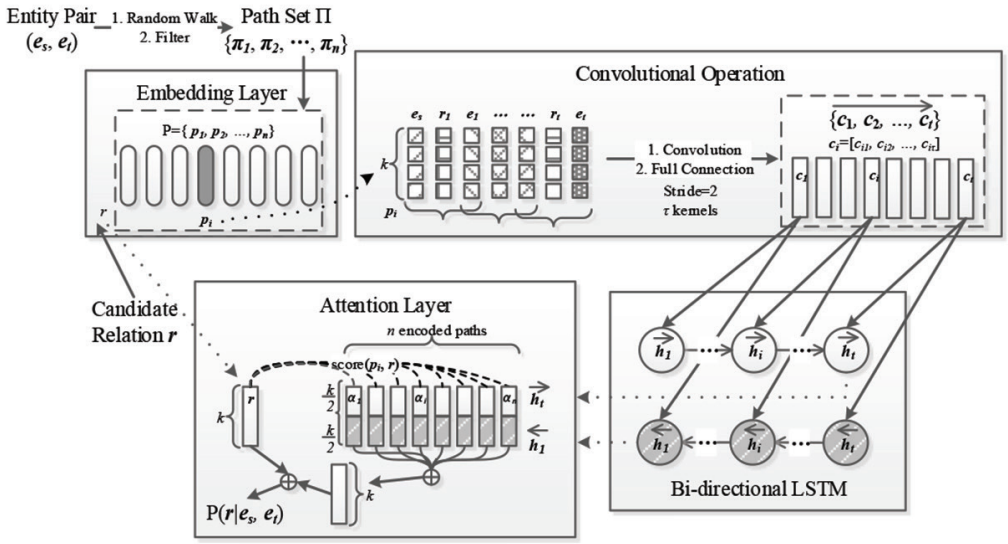

Fig. 1. Framework of CLAP

3.1 Vector embedding of relational paths with CNN

Each KG contains an entity set E and a relation set R. In one triple, subject (h, r, t), $h \in$ E represents the head entity, object $t \in$ E represents the tail entity, and predicate

$r \in R$ represents the relation. The embedding of the triple is denoted as ($\boldsymbol{e}_s$, $\boldsymbol{r}$, $\boldsymbol{e}_t$), reflecting the orderly link between entities. There may be multiple paths between one entity pair, so treating the paths as atomic features leads to a rapid growth of the feature matrix as the data increases, which is infeasible. ConvE uses CNN to extract factual features while significantly reducing the number of parameters. In this paper, a customized CNN architecture is proposed to embed the paths into low-dimensional representations. Firstly, the path ranking algorithm (PRA) is applied to obtain highly probable paths with the head and tail entities $\boldsymbol{e}_s$, $\boldsymbol{e}_t$ as the start and the end, respectively. With random Walk, PRA starts from the head entity $\boldsymbol{e}_s$, searches paths within lengths in a specified scope throughout the entire graph, and records the relational sequence along with intermediate entities in each path. Record the probabilities of different paths reaching the tail entity $\boldsymbol{e}_t$, and filter them according to a preset threshold. A complete path $\boldsymbol{\pi}$ can be denoted as $\{\boldsymbol{e}_s, \boldsymbol{r}_1, \boldsymbol{e}_1, \boldsymbol{r}_2, \boldsymbol{e}_2, ..., \boldsymbol{e}_{i-1}, \boldsymbol{r}_i, \boldsymbol{e}_i, ..., \boldsymbol{r}_t, \boldsymbol{e}_t\} \in \Pi$, in which the relational sequence is $\{\boldsymbol{r}_1, \boldsymbol{r}_2, ..., \boldsymbol{r}_t\}$, $(\boldsymbol{e}_{i-1}, \boldsymbol{r}_i, \boldsymbol{e}_i)$ represents the i-th triple in the path, and Π represents the filtered collection of paths. The numbers of relations in different paths vary. Take the longest path; the number of relations is expressed as t. Set all paths to the same length t, and fill in blanks with zeros.

Vector representations of entity types [24] are utilized to reduce the number of parameters and address the issue of certain entities in the Test Set not being present in the Train Set. The entity pairs and relations are transformed into k-dimensional vectors using the embedding matrix, i.e., $\boldsymbol{e}_s, \boldsymbol{e}_t, \boldsymbol{r} \in \mathbb{R}^k$ and then input into the path convolution layer. The size and stride of the filter ω have a significant impact on feature extraction and calculation cost. To avoid extracting meaningless features, we use a unified size $\omega \in \mathbb{R}^{k\times 3}$ and stride of 2. Multiple kernels are employed to traverse paths, Ω and τ representing the collection and the number of kernels respectively, i.e., $\tau = |\Omega|$. Take all triples on each path as units or windows and extract their local patterns one by one. Concatenate all the features; the th feature vector of one path could be denoted as $\boldsymbol{c}_i = [c_{i1}, c_{i2}, ..., c_{i\tau}]$, $\boldsymbol{c}_i \in \mathbb{R}^\tau$, $c_{i\tau} = f(\omega_\tau[\boldsymbol{e}_{i-1}, \boldsymbol{r}_i, \boldsymbol{e}_i] + b)$ where f represents the nonlinear activation function ReLU [23–24], and b is the bias. After convolution, the vector sequence for each path is represented as $\{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_t\}$ an input into BLSTM.

3.2 Path feature merging with bidirectional LSTM

It is difficult for ordinary RNNs to learn long-distance semantic dependencies. Zhou et al. [27] employed BLSTM networks, which check the current states of nodes or cells using peephole connections to enhance bidirectional correlations between the constant error carousel (CEC) and each gate. The bidirectional gated recurrent unit (GRU) adopted by Lu R et al. [54] simplifies the cell structure and reduces the parameter number with a similar coupling gated structure [55]. We use BLSTM to merge the vector sequences into single vectors.

Each τ-dimensional output vector $\boldsymbol{c}_i$ from the convolution layer is considered a time step in BLSTM. BLSTM reads data from two opposite directions, forward and backward, with the outputs denoted as $\overrightarrow{\boldsymbol{h}_j}$ and $\overleftarrow{\boldsymbol{h}_j}$ respectively. Two sets of hidden states are obtained, i.e., for a vector sequence $\{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_t\}$, the state sequence $\{\overrightarrow{\boldsymbol{h}_1}, \overrightarrow{\boldsymbol{h}_2}, ..., \overrightarrow{\boldsymbol{h}_j}, ..., \overrightarrow{\boldsymbol{h}_t}\}$ is obtained with the forward LSTM network, and $\{\overleftarrow{\boldsymbol{h}_1}, \overleftarrow{\boldsymbol{h}_2}, ..., \overleftarrow{\boldsymbol{h}_j}, ..., \overleftarrow{\boldsymbol{h}_t}\}$ is obtained with the backward network. In order to reduce the parameter count, the last hidden state of the forward sequence is concatenated with the first hidden state of the backward sequence to generate a vector representation $\boldsymbol{p}$ for the complete path $\boldsymbol{\pi}$, $\boldsymbol{p} = \left[\overrightarrow{\boldsymbol{h}_t}, \overleftarrow{\boldsymbol{h}_1}\right]$, $\boldsymbol{p} \in \mathbb{R}^k$, preserving sequential information.

The dimension of hidden states of cells is set to $\frac{k}{2}$ facilitate concatenation and matching with candidate relations. All n paths are processed simultaneously the same encoders in the Time Distributed layer of Keras. The set of vector representations $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_n\}$, $P \in \mathbb{R}^{k \times n}$ is input into the attention layer.

3.3 Path integration with attention mechanism

Max and Mean operations ignore the differences in evidential reasoning among different paths. The additive attention mechanism proposed by Bahdanau et al. [50] is employed for path integration, as it offers greater adaptability for various value intervals compared to dot product-based semantic similarity [24], [51]. The correlation score($\boldsymbol{p}_i$, $\boldsymbol{r}$) is calculated between each path of the head and tail entity pair and the vector representation of the candidate relation $\boldsymbol{r}$ separately, as shown in (1).

$$\text{score}(\boldsymbol{p}_i, \boldsymbol{r}) = \tanh(\boldsymbol{p}_i \boldsymbol{W}_s)\boldsymbol{r} \tag{1}$$

where, $\boldsymbol{W}_s \in \mathbb{R}^{k \times k}$ is the weight parameter. The weight α_i for each path is then assigned with the scores, $\alpha_i = \frac{\exp(\text{score}(\boldsymbol{p}_i, \boldsymbol{r}))}{\sum_1^n \exp(\text{score}(\boldsymbol{p}_i, \boldsymbol{r}))}$. The state vector $\boldsymbol{c}$ of the candidate relation is obtained by weighted calculation, $\boldsymbol{c} = \sum_1^n \alpha_i \boldsymbol{p}_i$ and the probability score $P(\boldsymbol{r} | \boldsymbol{e}_s, \boldsymbol{e}_t)$ for the relation is calculated to determine whether the triple is valid, shown in (2).

$$P(\boldsymbol{r} | \boldsymbol{e}_s, \boldsymbol{e}_t) = f(\boldsymbol{W}_p(\boldsymbol{c} + \boldsymbol{r})) \tag{2}$$

The weight parameter $\boldsymbol{W}_p \in \mathbb{R}^k$ is missing, and it f represents a nonlinear activation function like the Sigmoid function. With weight allocation, paths with varying degrees of semantic correlations to the candidate relation are distinguished.

The Adam optimizer [56] is used to train CLAP. The loss function is defined as equation (3).

$$L(\Theta) = -\frac{1}{N}\left(\sum_{(\boldsymbol{e}_s, \boldsymbol{r}, \boldsymbol{e}_t) \in T^+} \log P(\boldsymbol{r} | \boldsymbol{e}_s, \boldsymbol{e}_t) + \sum_{(\widehat{\boldsymbol{e}_s}, \hat{\boldsymbol{r}}, \widehat{\boldsymbol{e}_t}) \in T^-} \log\left(1 - P\left(\hat{\boldsymbol{r}} | \widehat{\boldsymbol{e}_s}, \widehat{\boldsymbol{e}_t}\right)\right)\right) + \lambda \|\Theta\|_2^2 \tag{3}$$

Where, N is the total number of training samples is missing. T^+,T^- represent the set of valid triples and invalid triples, respectively. Θ represents all parameters that need to be learned (initialized randomly). L2 regularization is adopted to prevent overfitting.

4 EXPERIMENT AND ANALYSIS

Physical environment: Experiments are conducted on a Lenovo SR590 server with the following hardware configuration: 20-core Xeon*2 processor, 16GB*8 memory, 1.2TB*3 SAS disks (in RAID5 mode), and a cluster of GTX3080Ti GPUs.

Task description: Link prediction involves inferring new facts for KGC. It computes in which the probability of the connection between a given pair of entities and a specific relation to determine the validity of the triple. The ranking of the correct answer among all candidates is used for evaluation. Take the tail entity prediction for

example, for the query (Joe Biden, *is President of*?), we expect "the U.S." or "America" to be ranked first.

Datasets: Two conventional benchmark datasets, FB15k-237 and NELL995, a large dataset FC17 (simulating real-world scenarios) [28], [45], and a sparse dataset NELL-One [46] are utilized, all of which are publicly available. Dataset statistics are shown in Table 1, except for NELL-One (a small dataset). In NELL995, the triples with the top 200 highest frequency relations are kept. Toutanova et al. [38] removed reverse triples from FB15k and created FB15k-237 to eliminate high score loopholes. The distribution of relation patterns in FB15k-237 is more complex than in NELL995. Most from the data of FC17 is sourced from Freebase [4] and aligned with ClueWeb [7]. In our experiment, we selected 46 relations with the highest frequencies. NELL-One is a subset of NELL that contains the number of triple instances ∈ [50,500] for each relation.

Table 1. Dataset statistics

Datasets	#Entities	#Relations	#Train Set	#Val. Set	#Test Set	#Tasks
NELL995	75492	200	154213	5000	5000	12
FB15k-237	14541	237	272115	17535	20466	20
FC17	1.8e7	25994	3.05e5	1.2e4	1.2e4	46

Metrics: Several commonly accepted metrics are adopted, including mean average precision (MAP), mean reciprocal rank (MRR), Hits@N (proportion of the valid triple ranked in the first N candidates, N = 1, 3, 5), precision, recall, and F1 scores. The definitions of these metrics are listed from (4) to (8).

$$\mathrm{MAP} = \frac{1}{|Q_r|}\sum_{q\in Q_r}\mathrm{AP}(q) \tag{4}$$

where Q_r is the set of queries, AP refers to the average precision and $q \in Q_r$.

$$\mathrm{MRR} = \frac{1}{|Q_r|}\sum_{q\in Q_r}\frac{1}{|rank_q|} \tag{5}$$

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{6}$$

where TP refers to true positive and FP refers to false positive.

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{7}$$

where FN refers to false negative.

$$\mathrm{F}_1 = 2\cdot\frac{\mathrm{Precision}\cdot\mathrm{Recall}}{\mathrm{Precision}+\mathrm{Recall}} \tag{8}$$

Baseline models include:

1. TransE [18] (2013) is a classic translation-based distance model. It was only used on NELL-One due to performance concerns.
2. DistMult [19] (2015) utilizes diagonal matrices to represent relations.

3. DeepPath [30] (2017) is a reinforcement learning framework that is based on TransE.
4. Single-Model [24] (2017) employed RNN to process relational sequences. LogSumExp, as recommended in the original study, is adopted.
5. Att-Model + Type [28] (2017), which could be considered the Single-Model with the attention mechanism, was re-implemented in our study.
6. ConvE [20] (2018) utilizes 2D convolution for concatenating entities and relation.
7. G-GAT [52] (2019) utilized the attention mechanism to extract neighbor features and focused on relation prediction in complex datasets. The results on FB15k-237 from the original study are cited.
8. M-walk [29] (2018) combines RL and RNN, but it is only used on conventional datasets.
9. GMH [45] (2020) is a multi-hop reasoning framework that emphasizes both local features and the overall graph structure. It is designed for complex scenarios and was only used on FC17.
10. G-matching [46] (2018) is a support framework for few-shot reasoning that primarily exploits local patterns. The performance of CLAP, TransE, and DistMult (with and without G-matching) is compared on NELL-One.

GMH takes pre-trained embeddings of ConvE as input and achieves the best performance when the upper limit of distance is set to 6. For other models, recommended hyperparameters from the original studies are adopted.

Implementation Details: The reliability and reasoning efficiency of long paths gradually decline, so the path length limit is set to 4. Correspondingly, the maximum number of elements is 9, including intermediate entities or fillings. Set the probability threshold of the random alk to 0.1. The Bernoulli distribution [32] is used to generate invalid triples by randomly replacing head or tail entities.

Performance is verified on Val. Set. If the improvement in accuracy in the last 10 epochs $< 10^{-2}$, the training is stopped, and the parameters are finalized. The upper limit of epochs is set to 1000, and in most cases, the training stops with fewer than 500 epochs. Grid Search is employed to find optimal hyperparameters. The hyperparameter pool is as follows: minibatch size = 64, learning rate, $\gamma \in [1e^{-5}, 1e^{-4}, 5e^{-4}]$ dimension, $k \in [50, 100, 200]$, number of hidden nodes in LSTM, and $\in [64, 128]$, $\tau \in [50, 100]$, L2 regularization coefficient, $\in [0, 0.001, 0.01, 0.1, 0.5]$. For other parameters in Adam, the default setting is adopted.

Results on conventional datasets are shown in Table 2. The best performance is in bold, and the sub-optimal is in italic with underline. The slash indicates that the original results are unavailable. CLAP performs best overall and shows improvement over two similar methods, Single-Model and Att-Model + Type, particularly on the complex dataset FB15k-237. It maintains the same level of time complexity for both training and prediction. On NELL995, due to the limited number of paths for certain entity pairs, path-based models show a slight decrease in performance at Hits@1, 3, whereas CLAP remains relatively stable. DistMult is effective in extracting entity similarity features, achieving high mean reciprocal rank (MRR) scores on both datasets without taking path semantics into account. On the dense dataset NELL995, DeepPath compensates for the limited expressivity caused by translation operations with RL-based path extension and maintains stability across various indicators. ConvE performs well on NELL995, while scores plunge on FB15k-237. This implies that concatenation and reshaping may help extract relational features, but neglecting translation attributes may lead to local pattern losses. G-GAT, designed

for complex datasets, outperforms DeepPath but is outperformed by Att-Model + Type. This suggests that multi-hop paths could offer richer semantics compared to single-hop neighbors. Nathani et al. also discuss their intention to incorporate additional semantic information, such as text descriptions. M-walk achieves the highest score at Hits@1 on NELL995 but faces interference from invalid paths on FB15k-237.

Table 2. Performance comparison on NELL995 and FB15k-237

Dataset	NELL995				FB15k-237			
Model	MAP	MRR	Hits@1	Hits@3	MAP	MRR	Hits@1	Hits@3
DistMult	/	*0.860*	0.752	0.865	/	*0.558*	0.446	0.573
DeepPath	0.811	0.852	0.808	0.884	0.553	0.495	0.449	0.524
Single-Model	0.827	0.833	0.765	0.903	0.525	0.512	0.496	0.557
Att-Model + Type	*0.838*	0.847	0.783	0.905	*0.558*	0.556	*0.513*	*0.626*
ConvE	/	**0.862**	0.826	*0.919*	/	0.509	0.430	0.527
G-GAT	/	/	/	/	/	0.518	0.460	0.540
M-walk	0.829	0.848	**0.834**	0.910	0.532	0.488	0.475	0.543
CLAP	**0.846**	0.859	*0.829*	**0.941**	**0.564**	**0.589**	**0.528**	**0.671**

Select outstanding baselines and compare their MAP scores on different relations in NELL995. The results are shown in Figure 2. DeepPath only considers local features, while Single-Model does not differentiate the weights for paths with varying degrees of semantic correlations. CLAP addresses these shortcomings across all 10 relations. Compared with Att-Model + Type, CLAP outperforms in 7 relations, particularly in complex relations such as *atheletePlaysForTeam* and *bornLocation*. This suggests that the fusion of convolutional feature extraction and BLSTM path merging is beneficial for knowledge representation.

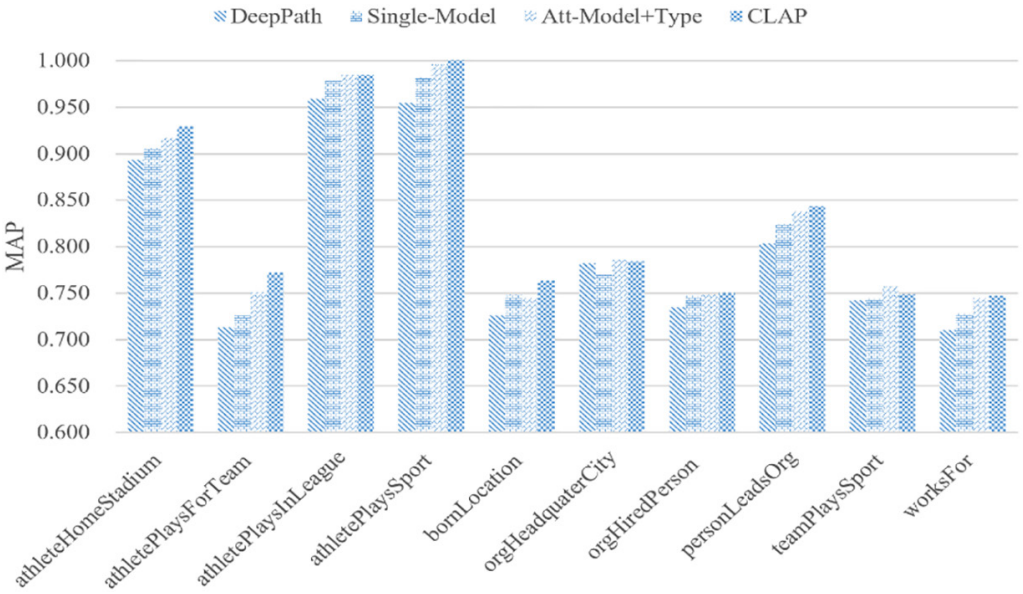

Fig. 2. Comparison of MAP scores on various relations of NELL995

Results on FC17 are shown in Table 3. CLAP achieves the highest scores in all three indicators. Compared with Att-Model + Type, GMH considers the overall graph structure, enhancing accuracy in recognizing valid relations and mitigating the effects of incorrect relations in long-distance reasoning. However, the computational cost is high, and the model converges slowly. For CLAP, the path length is set manually, while GMH can adaptively adjust the threshold, which is saved for future work.

Table 3. Performance comparison on FC17

Model	MRR	Hits@1	Hits@3
Att-Model + Type	0.243	0.114	0.154
GMH	*0.254*	*0.139*	*0.183*
CLAP	**0.282**	**0.146**	**0.188**

Results on Nell-One are displayed in Table 4. Without G-matching, CLAP is stronger than TransE and DistMult, demonstrating the significance of integrating path semantics. After G-matching is applied, the scores of all three models increase, indicating that single-hop local structure is helpful in discovering similar facts. The proportions of improvement are 94.0% (TransE), 65.7% (DistMult), and 6.2% (CLAP) respectively, suggesting that the neighboring information could be effectively replaced with path semantics to a great extent.

Table 4. Performance comparison on NELL-One

Model	MRR	Hits@1	Hits@5
TransE	0.083	0.039	0.147
DistMult	0.105	0.066	0.136
CLAP	*0.178*	0.108	0.197
G-matching (TransE)	0.161	*0.129*	*0.210*
G-matching (DistMult)	0.174	0.114	0.202
G-matching (CLAP)	**0.189**	**0.143**	**0.226**

An extended study was conducted to evaluate the effects of different entity type coverage, various path lengths, and different LSTMs on NELL995. The results are presented in Table 5. Most entities in NELL995 carry type information, while embedded representations are employed for others. The differences among various coverage options are minimal. When the coverage is low, performance slightly decreases if the Test Set contains entities that do not appear in the Train Set. When the path length is set to 4, performance increases slightly, implying: (1) there are not enough paths between some entity pairs when the threshold is small; and (2) short paths provide most of the semantic information. Different LSTMs have trivial impacts.

Single-Model and Att-Model + Type were selected for comparison based on indicators such as Precision, Recall, and F1 scores on NELL995. The results are shown in Figures 3 and 4. CLAP demonstrates a more balanced performance and achieves higher F1 scores compared to the baselines. When the recall score increases, the decline curve of precision is relatively smooth, implying the superiority of the framework, especially the advantages of integrating convolutions.

Table 5. Comparison between different coverage, path lengths, and LSTM models on NELL995

	MAP	MRR	Hits@1	Hits@3
Coverage = 30%	0.842	0.855	0.824	0.936
Coverage = 70%	0.845	**0.861**	0.825	0.939
Coverage = 100%	**0.846**	0.859	**0.829**	**0.941**
Path Length = 3	0.833	0.842	0.817	0.926
Path Length = 4	**0.846**	0.859	**0.829**	**0.941**
BLSTM [27]	0.841	0.857	0.827	0.937
Bi-GRU [55]	0.845	0.852	0.828	**0.941**
Our BLSTM	**0.846**	0.859	**0.829**	**0.941**

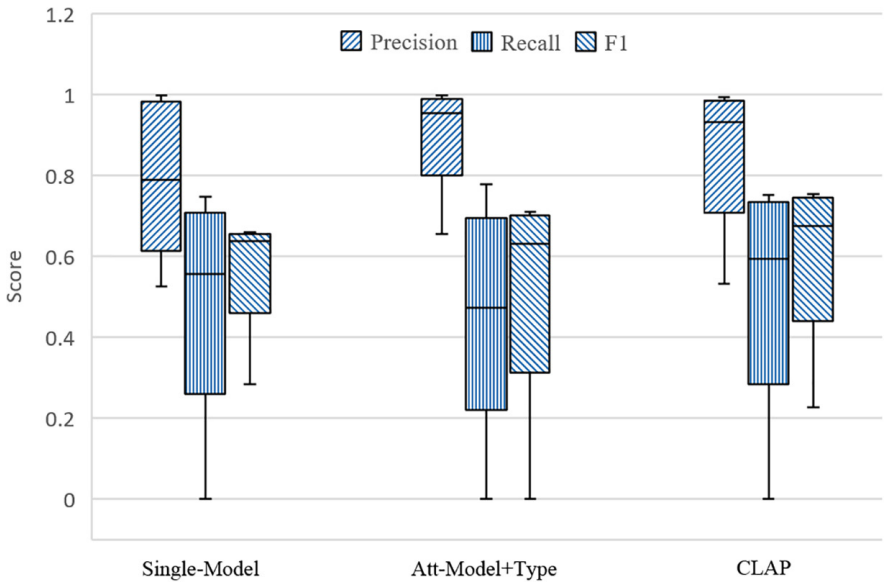

Fig. 3. Comparison of precision, recall and F1 scores on NELL995

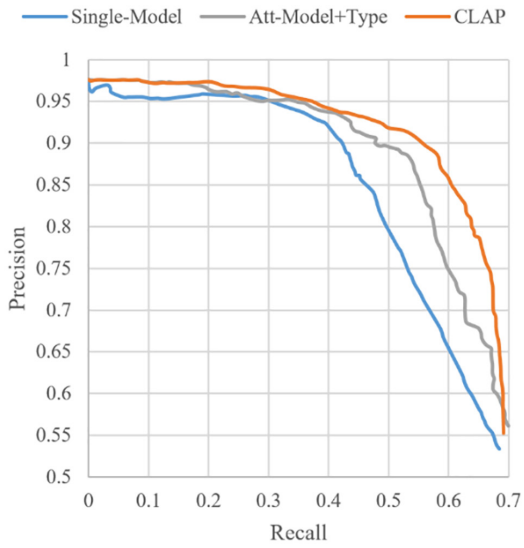

Fig. 4. Comparison of precision-recall curve on NELL995

5 CONCLUSION

For better feature extraction and path semantics recognition, a customized CNN framework combines the bidirectional LSTM and the attention mechanism. This integration aims to enhance entity and relation interactions, merge relational sequences into single vectors, and integrate path semantics with weights to compute triple probability scores. Experimental results show that CLAP has a strong learning ability for complex relations and can conduct knowledge reasoning on conventional, large, and sparse datasets, achieving overall higher precision, recall, and F1 scores. Still, there is room for improvement in datasets that do not offer enough paths for entity pairs. Therefore, future work includes utilizing the RL framework, introducing fact confidence [57], integrating multi-modal information, and/or expanding embedding spaces for higher expressivity.

6 ACKNOWLEDGMENT

On behalf of all the authors, I express our gratitude to J. Wang, Q. Chen, and S. Chen for their selfless assistance and admirable patience.

7 REFERENCES

[1] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes *et al.*, "DBpedia-a large-scale, multilingual knowledge base extracted from Wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015. https://doi.org/10.3233/SW-140134

[2] T. Mitchell, W. Cohen, F. Hruschka, P. Talukdar, B. Yang, J. Betteridge *et al.*, "Never-ending learning," *Communications of the ACM*, vol. 61, no. 5, pp. 103–115, 2018. https://doi.org/10.1145/3191513

[3] T. Rebele, F. Suchanek, J. Hoffart, J. Biega, E. Kuzey, and G. Weikum, "YAGO: A multilingual knowledge base from wikipedia, wordnet, and geonames," in *Proceedings of International Semantic Web Conference*, Kobe, 2016, pp. 177–185. https://doi.org/10.1007/978-3-319-46547-0_19

[4] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, Vancouver, 2008, pp. 1247–1250. https://doi.org/10.1145/1376616.1376746

[5] C. Xiong, R. Power, and J. Callan, "Explicit semantic ranking for academic search via knowledge graph embedding," in *Proceedings of the 26th International Conference on World Wide Web*, Geneva, 2017, pp. 1271–1279. https://doi.org/10.1145/3038912.3052558

[6] Y. Hao, Y. Zhang, K. Liu, S. He, Z. Liu, H. Wu, and J. Zhao, "An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (vol. 1: Long Papers)*, Vancouver, 2017, pp. 221–231. https://doi.org/10.18653/v1/P17-1021

[7] A. Bordes, S. Chopra, and J. Weston, "Question answering with subgraph embeddings," *arXiv Preprint*, no. 1406.3676, 2014. https://doi.org/10.3115/v1/D14-1067

[8] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T. S. Chua, "Explainable reasoning over knowledge graphs for recommendation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Hawaii, 2019, pp. 5329–5336. https://doi.org/10.1609/aaai.v33i01.33015329

[9] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta *et al.*, "Knowledge base completion via search-based question answering," in *Proceedings of the 23rd International Conference on World Wide Web*, Seoul Korea, 2014, pp. 515–526. https://doi.org/10.1145/2566486.2568032

[10] Q. Liu, Y. Li, H. Duan, Y. Liu, and Z. G. Qin, "Overview of knowledge graph construction techniques," *Computer Research and Development*, vol. 53, no. 3, pp. 582–600, 2016.

[11] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic Web*, vol. 8, no. 3, pp. 489–508, 2017. https://doi.org/10.3233/SW-160218

[12] X. Chen, S. Jia, and Y. Xiang, "A review: Knowledge reasoning over knowledge graph," *Expert Systems with Applications*, vol. 141, no. 3, pp. 429–449, 2020. https://doi.org/10.1016/j.eswa.2019.112948

[13] A. Rossi, D. Barbosa, D. Firmani, A. Matinata, and P. Merialdo, "Knowledge graph embedding for link prediction: A comparative analysis," *arXiv Preprint*, no. 2002.00819, 2020.

[14] Z. Sun, S. Vashishth, S. Sanyal, P. Talukdar, and Y. Yang, "A re-evaluation of knowledge graph completion methods," *arXiv Preprint*, no. 1911.03903, 2019. https://doi.org/10.18653/v1/2020.acl-main.489

[15] H. Cai, V. W. Zheng, and K. C. C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018. https://doi.org/10.1109/TKDE.2018.2807452

[16] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 1, pp. 163–170, 2020.

[17] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017. https://doi.org/10.1109/TKDE.2017.2754499

[18] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embedding for modeling multi-relational data," *Advances in Neural Information Processing Systems*, vol. 26, pp. 2787–2795, 2013.

[19] B. Yang, W. T. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," *arXiv Preprint*, no. 1412.6575, 2014.

[20] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2D knowledge graph embeddings," in *Proceedings of National Conference on Artificial Intelligence*, New Orleans, 2018, vol. 32, no. 1, pp. 1811–1818. https://doi.org/10.1609/aaai.v32i1.11573

[21] N. Lao and W. W. Cohen, "Relational retrieval using a combination of path-constrained random walks," *Machine Learning*, vol. 81, no. 1, pp. 53–67, 2010. https://doi.org/10.1007/s10994-010-5205-8

[22] N. Lao, T. Mitchell, and W. W. Cohen, "Random walk inference and learning in a large scale knowledge base," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Edinburgh, 2011, pp. 529–539.

[23] A. Neelakantan, B. Roth, and A. McCallum, "Compositional vector space models for knowledge base completion," *arXiv Preprint*, no. 1504.06662, 2015. https://doi.org/10.3115/v1/P15-1016

[24] R. Das, A. Neelakantan, D. Belanger *et al.*, "Chains of reasoning over entities, relations, and text using recurrent neural networks," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Valencia, 2017, pp. 132–141. https://doi.org/10.18653/v1/E17-1013

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. https://doi.org/10.1162/neco.1997.9.8.1735

[26] K. Xu, J. Ba, R. Kiros *et al.*, "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of International Conference on Machine Learning*, Lille, 2015, pp. 2048–2057.

[27] P. Zhou, W. Shi, J. Tian *et al.*, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Berlin, 2016, pp. 207–212. https://doi.org/10.18653/v1/P16-2034

[28] X. Jiang, Q. Wang, B. Qi *et al.*, "Attentive path combination for knowledge graph completion," in *Proceedings of Asian Conference on Machine Learning*, Seoul, 2017, pp. 590–605.

[29] Y. Shen, J. Chen, P. S. Huang *et al.*, "M-walk: Learning to walk over graphs using monte carlo tree search," *arXiv Preprint*, no. 1802.04394, 2018.

[30] W. Xiong, T. Hoang, and W. Y. Wang, "DeepPath: A reinforcement learning method for knowledge graph reasoning," *arXiv Preprint*, no. 1707.06690, 2017. https://doi.org/10.18653/v1/D17-1060

[31] A. Bordes, X. Glorot, J. Weston *et al.*, "A semantic matching energy function for learning with multi-relational data," *Machine Learning*, vol. 94, no. 2, pp. 233–259, 2014. https://doi.org/10.1007/s10994-013-5363-6

[32] Z. Wang, J. Zhang, J. Feng *et al.*, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of National Conference on Artificial Intelligence*, Québec, 2014, vol. 28, no. 1, pp. 1112–1119. https://doi.org/10.1609/aaai.v28i1.8870

[33] Y. Lin, Z. Liu, M. Sun *et al.*, "Learning entity and relation embeddings for knowledge graph completion," in *Proceedings of National Conference on Artificial Intelligence*, Texas, 2015, vol. 29, no. 1, pp. 2181–2187. https://doi.org/10.1609/aaai.v29i1.9491

[34] T. Trouillon, J. Welbl, S. Riedel *et al.*, "Complex embeddings for simple link prediction," in *Proceedings of International Conference on Machine Learning*, New York, 2016, pp. 2071–2080.

[35] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv Preprint*, no. 1408.5882, 2014. https://doi.org/10.3115/v1/D14-1181

[36] I. Balažević, C. Allen, and T. M. Hospedales, "Hypernetwork knowledge graph embeddings," in *Proceedings of International Conference on Artificial Neural Networks*, Munich, 2019, pp. 553–565. https://doi.org/10.1007/978-3-030-30493-5_52

[37] Q. Wang, J. Liu, Y. Luo *et al.*, "Knowledge base completion via coupled path ranking," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, 2016, pp. 1308–1318. https://doi.org/10.18653/v1/P16-1124

[38] K. Toutanova, X. V. Lin, W. Yih *et al.*, "Compositional learning of embeddings for relation paths in knowledge base and text," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, 2016, pp. 1434–1444. https://doi.org/10.18653/v1/P16-1136

[39] R. Das, S. Dhuliawala, M. Zaheer *et al.*, "Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning," *arXiv Preprint*, no. 1711.05851.

[40] Y. Luo, Q. Wang, B. Wang *et al.*, "Context-dependent knowledge graph embedding," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Libson, 2015, pp. 1656–1661. https://doi.org/10.18653/v1/D15-1191

[41] C. Shang, Y. Tang, J. Huang *et al.*, "End-to-end structure-aware convolutional networks for knowledge base completion," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Hawaii, 2019, pp. 3060–3067. https://doi.org/10.1609/aaai.v33i01.33013060

[42] Y. L. Tuan, Y. N. Chen, and H. Lee, "DyKgChat: Benchmarking dialogue generation grounding on dynamic knowledge graphs," *arXiv Preprint*, no. 1910.00610, 2019. https://doi.org/10.18653/v1/D19-1194

[43] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proceedings of International Conference on Machine Learning*, Lille, 2015, pp. 2342–2350.

[44] K. Greff, R. K. Srivastava, J. Koutník *et al.*, "LSTM: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2016. https://doi.org/10.1109/TNNLS.2016.2582924

[45] Y. Zhang, X. Zhang, J. Wang *et al.*, "GMH: A general multi-hop reasoning model for KG completion," *arXiv Preprint,* arXiv:2010.07620, 2020. https://doi.org/10.18653/v1/2021.emnlp-main.276

[46] W. Xiong, M. Yu, S. Chang *et al.*, "One-shot relational learning for knowledge graphs," *arXiv Preprint*, no. 1808.09040, 2018. https://doi.org/10.18653/v1/D18-1223

[47] R. Takahashi, R. Tian, and K. Inui, "Interpretable and compositional relation learning by joint training with an autoencoder," *arXiv Preprint*, no. 1805.09547, 2018. https://doi.org/10.18653/v1/P18-1200

[48] X. V. Lin, R. Socher, and C. Xiong, "Multi-hop knowledge graph reasoning with reward shaping," *arXiv Preprint*, no. 1808.10568, 2018. https://doi.org/10.18653/v1/D18-1362

[49] Q. Xie, X. Ma, Z. Dai *et al.*, "An interpretable knowledge transfer model for knowledge base completion," *arXiv Preprint*, no. 1704.05908, 2017. https://doi.org/10.18653/v1/P17-1088

[50] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv Preprint*, no. 1409.0473, 2014.

[51] A. Vaswani, N. Shazeer, N. Parmar *et al.*, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.

[52] D. Nathani, J. Chauhan, C. Sharma *et al.*, "Learning attention-based embeddings for relation prediction in knowledge graphs," *arXiv Preprint*, no. 1906.01195, 2019. https://doi.org/10.18653/v1/P19-1466

[53] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proceedings of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, 2013, pp. 6645–6649. https://doi.org/10.1109/ICASSP.2013.6638947

[54] R. Lu and Z. Duan, "Bidirectional GRU for sound event detection," *Detection and Classification of Acoustic Scenes and Events*, vol. 11, no. 16, pp. 17–20, 2017.

[55] J. Chung, C. Gulcehre, K. H. Cho *et al.*, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv Preprint*, no. 1412.3555, 2014.

[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv Preprint*, no. 1412.6980, 2014.

[57] X. Chen, M. Chen, W. Shi *et al.*, "Embedding uncertain knowledge graphs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, New York, 2019, pp. 3363–3370. https://doi.org/10.1609/aaai.v33i01.33013363

8 AUTHORS

Chen Xinyuan, Fuzhou Technology and Business University, Fuzhou, China (E-mail: 516040610@qq.com).

Ubaldo Comite, University Giustino Fortunato, Benevento, Italy.