Applied Intelligence (2024) 54:1497–1510
https://doi.org/10.1007/s10489-023-05254-4

# Enhancing missing facts inference in knowledge graph using triplet subgraph attention embeddings

Anish Khobragade[1] · Shashikant Ghumbre[2] · Vinod Pachghare[1]

Accepted: 24 December 2023 / Published online: 8 January 2024

**Abstract**
Due to the heterogeneous structure of the knowledge graph (KG), relationships between entities remain missing. However, optimal use of KG requires inference of missing fact triplet (entity-relation-entity). The fact inference predicts a missing relationship using an embedding approach in a supervised learning setup, representing entities and relationships in a low-dimensional vector space. Recent work uses attention-aware embeddings, but when applied directly to entire KG, attention mechanisms can be computationally expensive, especially for large graphs. The attention-based KG embedding model uses negative sampling, which can cause a gradient vanishing problem during learning. This paper proposes a novel triplet subgraph attention embedding (TSAE) model that combines a simplified graph attention mechanism with a neural network to learn embedding without negative sampling requirements. The attention layer processes the triplet-level subgraph entities to learn the central entity features by aggregating the neighbor's features. A neural network processes attention-aware triplet entity features through hidden layers to compute the likelihood of relationship types between triplet entities. TSAE generates more fine-grained entity embeddings using simplified attention mechanism, reduces computational complexity, and offers interpretable embeddings. Experimental results on the benchmark data sets exhibit TSAE superiority over the baselines. The case study shows the efficacy of the model for the KG completion task.

**Keywords** Embedding · Graph attention mechanism · Knowledge graph · Relation prediction · Representation learning

## 1 Introduction

KG is a graph-structured data model that integrates data and stores interlinked descriptions of entities. KG efficiently encodes knowledge and reasoning about complex topics. They help users grasp and model complex concepts and make better business decisions based on factual data [15]. In KG, the Resource Description Framework (RDF) is primarily used for describing linked data resources on the web. RDF uses a subject-predicate-object triplet format ($s-r-o$), where the subject ($s$) and object ($o$) are entities such as persons, places, locations, and events. A predicate ($r$) is a relation type between entity pairs, such as *is_a*, *live_in*, and *located_in*. These triplets are linked to create a graph of interconnected entities. Consider the triplet "ElonMusk-*wasBornIn*-Pretoria," in which *wasBornIn* is a relation type between Elon Musk and Pretoria. Large-scale open KG such as Yago, WikiData, DBpedia, FreeBase, and others are available publicly on the web [38]. The key to KGs success is its flexibility in merging and connecting heterogeneous data regardless of their disparity. However, entities and relations may have heterogeneous descriptions, leading to incomplete triplets. Besides, KGs can be incomplete due to user-generated organic entities that include meaningless representations, invalid or incomplete attributes, stale contents, or no member that mapped entity pairs. Manually enumerating the relation types among pairs of entities is complex and time consuming [4]. Since the KG is incomplete by design, it possibly leads to unreliable query results in applications such as question-answering [22], cyber defense [14], and recommendation systems [8].

✉ Anish Khobragade
anishraj.comp@coeptech.ac.in

Shashikant Ghumbre
shashi.ghumbre@gmail.com

Vinod Pachghare
vkp.comp@coeptech.ac.in

1 Computer Science and Engineering, COEP Technological University, Pune, Maharashtra 411005, India

2 Computer Engineering, Government College of Engineering and Research, Avasari Khurd, Pune, Maharashtra 412405, India

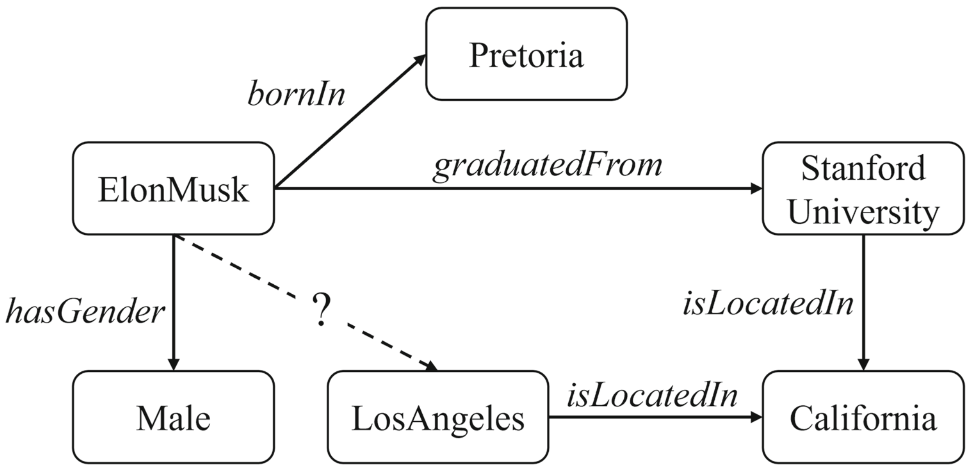


**Fig. 1** KG example from the YAGO3-10 dataset

In recent years, KG representation learning (KGRL) has focused on mapping a latent graph structure to a numerical feature vector space, also known as embedding. Those learned embeddings are used to estimate the missing links by computing the scores of all possible triplets. Progress in recent KGRL grouped into three types according to the embedding approach. The first category is the translation distance model [2], and its variants embed the triplet $(s - r - o)$ in real-value vector space by projecting them in different spaces with different constraints. The second category is the tensor factorization model [27], and its variants capture latent semantics of KG in partially observed tensor of size $|E| \times |E| \times |R|$, where $E$ is the entity set, and $R$ is the relation set. The third category is neural network models that use the multilayer perceptron and convolution neural network (CNN). ConvE [5] is a CNN model that encodes the entities and relations with the activation function to score the triplet. [15]. The existing KG embedding model learning process requires an artificially contaminated negative triplet for each true positive triplet to compute the learning loss. However, negative sampling techniques such as uniform [2], Bernoulli [34], and KBGAN [3] suffer from gradient vanishing problems affecting the model performance. Therefore, embedding models without negative sampling requirements is needed.

The learned embeddings are utilized to predict the missing entity for a given relation-entity pair $(? - r - o$ or $s - r - ?$ ). The entity prediction approach utilizes the score function to compute the likelihood of all possible triplets created by replacing thousands of entities from the entity set $E$, which is a computationally intensive process. However, KG involves fewer relationship types compared to lots of entity types. Consequently, relation prediction makes missing fact inference computationally more efficient than entity prediction. This research aims to efficiently predict the missing relation $(s - ? - o)$ between a given entity pair $(s, o)$. Consider the subset example from the YAGO3-10 dataset shown in Fig. 1. The knowledge graph has the triplets such as "ElonMusk-*wasBornIn*-Pretoria," "ElonMusk-*graduatedFrom*-StanfordUniversity," "LosAngeles-*isLocatedIn*-California" "ElonMusk-*hasGender*-*Male*," and "StanfordUniversity-*isLocatedIn*-California." The relation type between Elon Musk and Los Angeles is currently missing. Infer a missing relation *livesIn* for the given entity pair "ElonMusk-?-LosEngeles" as an additional true fact makes the perception of knowledge graph completion.

The fact that the KG structure has a higher variance than the mean of the entity degree means that the relationship types are unevenly distributed over the entity pairs. Empirically, a significant number of entity pairs are associated with most relationship types. However, some entities have more relations with other entities. Therefore, it appears that influential information is present around an entity. The graph attention network [32] uses neighborhood information to obtain the embeddings. Recent attention-based KG embedding models such as KGAE [25], GGAE [17], and DMACM [9] derive embeddings using node attention mechanisms that iteratively transmit different assigned attentions and integrate them through different aggregation functions. These models consider the several independent attentions computed by various transformation operations. Consequently, attention mechanisms can be computationally expensive when applied directly to the entire KG, especially for large graphs. Consequently, a KG embedding model with a simplified attention technique is needed to reduce complexity and enhance missing fact inference.

We propose a novel Triplet Subgraph Attention Embedding (TSAE) model for relation prediction based on the preceding analysis to tackle the fact inference problem. TSAE uses an adjacency matrix to capture the subgraph structure from the input triplet entity pair and its neighborhood entities. TSAE maps the subgraph entities from lower to higher dimensions using a transformation matrix to expand the expressiveness of the feature. The attention mechanism computes the attention for each neighbor's entity to a triplet entity and subsequently updates the entity features using the aggregation function that involves itself and neighborhood attention-aware features. A dense feed-forward neural network processes attention-aware triplet entity features through hidden layers to compute the relative likelihood of all the relation types between the entity pairs. TSAE utilizes adjacent entity information (subgraph structure) to learn the entity features. The attention mechanism revises the features of a central entity (entity embedding) using features of all one-hop adjacent entities to take advantage of the substructure proximity. Based on the above discussion, TSAE has several advantages: (1) It limits the scope of attention to relevant subgraphs surrounding the triplets, reducing computational complexity while still capturing essential information. (2) It generates a more fine-grained embedding of the entity using simplified attention mechanisms that capture the better nuances of the neighborhood entities, leading to improved performance in inference tasks. (3) It offers more interpretable embedding because it explicitly considers the context of each triplet. Therefore, TSAE is helpful for applications where inexpensiveness, simplicity, and interpretability are essential. Key contributions are listed below:

1. We propose a triplet subgraph attention embedding model for the relation prediction based on the attention mechanism to update the central entity features by attending to all one-hop neighborhoods.
2. Design a feed-forward neural network to estimate the probability (score) of all the relation types between given entity pairs. Rank all triplets according to the respective relationship type score.
3. The experimental results on the three benchmark data sets show the superior performance of TSAE over state-of-the-art baselines. Extensive experiments to analyze model performance according to training parameters and a case study demonstrate effectiveness for the KG completion task.

## 2 Related work

In recent decades, translation distance models have made significant progress. Classical TransE [2] proposed the simplest embedding model, which tried to embed relational features $\boldsymbol{r}$ as a translation from the subject to object entity features while considering $\boldsymbol{s}+\boldsymbol{r} \approx \boldsymbol{o}$ as a triplet score. However, the translation distance creates an issue for complex relations categories [35]. TransH [34] gives a flexible solution by projecting the entity vectors in the relation-specific hyperplane. TransR [21] generalizes the approach of TransH by projecting entity vectors into different relation-specific spaces, improving model performance and increasing complexity. TransD [10] further separates the projection for the subject and object entities to obtain the projection with the same settings as TransH. The latest TorusE [6] extends the TransE principle precisely by projecting the entities and relations in a lie group called a torus. Hyun-Je Song enhances the translational model to learn new triplet entity embedding using existing learned embeddings as an incremental training process that minimizes the risk of parameter change [30]. DTransE [29] is the first-ever distributed training approach for TransE using a triplet partitioning algorithm to balance the resource workload by considering higher and lower-degree entities. TransX-pa [39] deals with entity circular and hierarchical structure using position-aware and self-attention block embedding to generalize the previous transnational model mechanism.

Another stream of embedding is the tensor factorization, first proposed in RESCAL [27] to capture the latent semantic in partially observed tensor $\boldsymbol{X}$ of size $|E| \times |E| \times |R|$. This tensor represents the presence and absence of a relationship type between the entity pair of the triplets. The underlying structure of KG is learned by rank-$d$ factorization, where each slice of $\boldsymbol{X}$ factorizes as $\boldsymbol{s}\mathbf{R}_k\boldsymbol{o}^\top$ to extract latent semantics. DistMult [36] simplifies the RECAL by acquiring the diagonal of $\mathbf{R}_k$ in a scoring function. Embedding captures the pairwise interaction among entities over the same dimension to reduce complexity. ComplEx [31] updates DistMult using a vector space for complex value features to improve the entity prediction performance. DURA [33] tackles the unrelated embedding for the entities with related semantics using duality-induced regularizer to ensure similar embedding for similar semantic entities.

Recent models adopted the convolution neural network. The first successful ConvE [5] model combines triplet subject entity and relation vectors in a 2D convolution layer that yields a feature map tensor, which then performs a dot product with the object vector to compute a score for a given triplet. ConvKB [26] combines triplet entities and relation vectors to a 2D convolution layer to increase learning efficiency. ConvR [12] captures relation-specific features from a given triplet entity vectors using relation-specific filters of 2D convolution. Graph2Seq [18] captures the subgraph structure using an information fusion approach to learn the entity embedding. A convolution layer of M-DCN [40] aggregates entities and relationship features, then extracts the interfaces between them based on a dynamically relevant filter.

In addition, some recent models have adopted an attention mechanism for KG embedding. KGAE [25] introduces a generalized node attention mechanism for link prediction by iteratively propagating different assigned attention. HARN [20] framework propagates attention based on relation-path neighbors of entities aggregated by different aggregator functions. GGAE [17] combines the one-hop neighbors and multi-hop neighbors entity features to compute the entity attention and relation attention to learn the entity and relation embedding. LSA-GAT [11] considers cycle and star-type spatial component structures using neighborhood aggregation combined with a CNN-based decoder to calculate triplet scores. The authors of [16] propose a graph attention-based model that treats a KG as an irregular graph and integrates graph-structured information through multiple independent channels. AR-KGAN [41] jointly models triplets and logical rules in a unified framework consisting of an encoder with a neighborhood aggregator. That effectively combines association rules-based and graph-based attention weights to capture entity relationships. JSSKGE [19] uses a graph attention mechanism to learn the subgraph structure and combines the soft logical rule to rectify the learned embeddings. T-GAE [7] utilizes attention mechanisms to capture the importance of different entities within the temporal knowledge graph. It introduces timespan-aware embedding to model the evolving nature of entities and relations using the LSTM network. The aforementioned methods deal with the problem of KG incompleteness. However, KGs can also contain corrupted triplets due to acquisition noise and deliberate attacks. Dealing with corrupted KGs involves techniques such as adopting anti-attacks, data validation, regular audits, and anomaly detection [23].

All the models reviewed above are for completing KG through the entity prediction task. But in general, fewer relationship types are used to connect many entities, as shown in Table 1, which illustrates the KG statistics. Moreover, the relationship types are not uniformly distributed across the entities, affecting the fact inference. The attention-based models consider the several independent attentions computed by various transformation operations. Accordingly, it can be computationally expensive when applied directly to the large size KG. Furthermore, the model uses artificially generated false (negative) samples to compute the training loss between true and false triplets. However, negative sampling is likely to cause the problem of vanishing gradients during training [13]. Therefore, we present a simplified graph attention technique to generate fine-grained embeddings without the negative sampling requirement for missing facts inference in the knowledge graph.

**Table 1** Statistics of data set

| Datasets | WN18RR | FB15k-237 | YAGO 3-10 |
|---|---|---|---|
| #Entities | 40943 | 14541 | 123182 |
| #Relations | 11 | 237 | 37 |
| #Triplets | 93012 | 310116 | 1089040 |
| #Train triplets | 86835 | 272115 | 1079040 |
| #Valid triplets | 3043 | 17535 | 5000 |
| #Test triplets | 3134 | 20466 | 5000 |
| Var [ Entity_degree] | 64.36 | 12336.01 | 542.64 |
| Avg [Entity_degree] | 4.17 | 37.39 | 8.76 |

## 3 Proposed methodology

### 3.1 Preliminary

The knowledge graph is a multi-relational structured network of real-world entities and relationships between a pair of entities connected by the directed edge. We define $KG = \{E, R, T\}$, where $E$ is a set of entities, $R$ is a set of relations, and $T$ denotes a set of observed fact triplets in the KG. Each triplet $(e_i, r_k, e_j)$ consists of a relation $r_k \in R$ between the entity pair $e_i, e_j \in E$. Entity feature vector $\boldsymbol{e}_i$, $\boldsymbol{e}_j \in \mathbb{R}^d$, where $d$ is the dimension size. Equation (1) creates the adjacency matrix $\boldsymbol{A}$ of size $|E| \times |E|$ representing the graph structure. The subgraph consists of a central entity $e_i$ and its surrounding neighbors. Let $\mathcal{N}_i$ signify the 1-hop incoming directed edge neighbors of $e_i$ including itself, as follows:

$$\boldsymbol{A}_{ij} = \begin{cases} 1 & \text{if } (e_i, r_k, e_j) \in T \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$\mathcal{N}_i = (e_i, \forall e_j \in E \wedge A_{ij} = 1 \mid e_i) \tag{2}$$

### 3.2 Problem formulation

In a supervised learning setup, the task of fact inference is formulated as an optimization problem to determine the missing relation between a given triplet entity pair. This is done by maximizing the likelihood of the true relation type using the updated attention-aware triplet entity features. Therefore, for the given pair of disjoint entities $(e_1, e_2)$, model the probability distribution $p(\forall r_k \in R | (e_1, e_2))$, such that the true relation type $r_{pred}$ gains the highest probability, as expressed in the following equation:

$$r_{pred} = max(p(\forall r_k \in R \mid (e_1, e_2))) \tag{3}$$

### 3.3 TSAE architecture

The TSAE framework consists of three parts. The first part takes the input triplet and generates the triplet subgraph set of triplet entities and their neighboring entities. The attention layer processes the subgraph set entity features to deliver attention-aware triplet entity features. The dense layer concatenates the attention-aware triplet feature vectors and processes them through hidden layers. The output layer

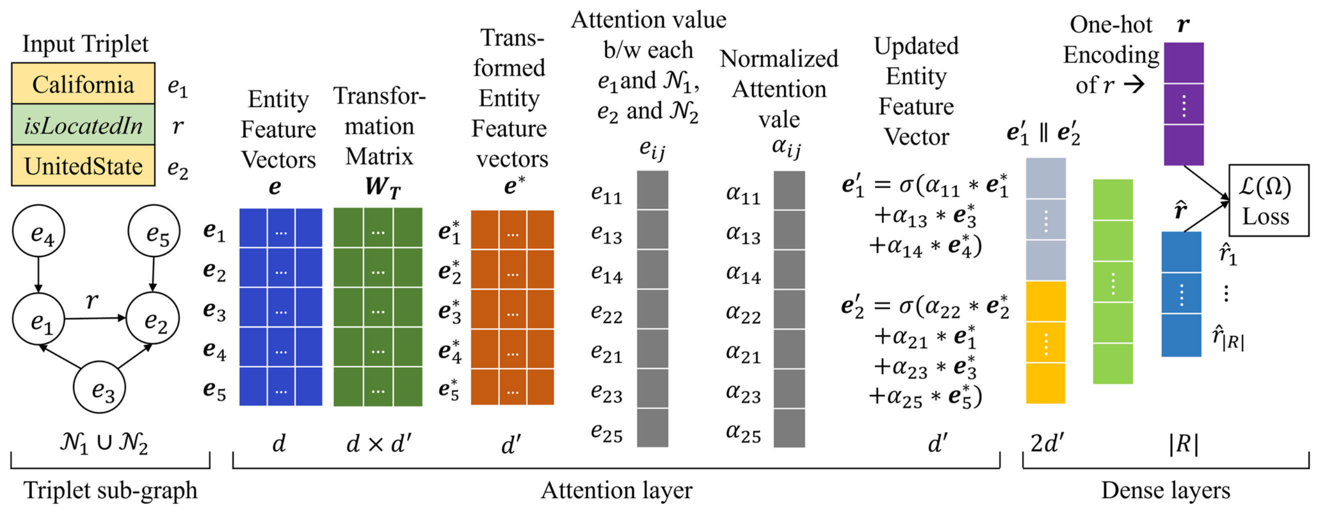

Input Triplet
California $e_1$
isLocatedIn $r$
UnitedState $e_2$
$\mathcal{N}_1 \cup \mathcal{N}_2$
Triplet sub-graph
Entity Feature Vectors $e$
$d$
Transfor-mation Matrix $W_T$
$d \times d'$
Trans-formed Entity Feature vectors $e^*$
$d'$
Attention value b/w each $e_1$and $\mathcal{N}_1$, $e_2$ and $\mathcal{N}_2$ $e_{ij}$
Normalized Attention vale $\alpha_{ij}$
Updated Entity Feature Vector
$e_1' = \sigma(\alpha_{11} * e_1^* + \alpha_{13} * e_3^* + \alpha_{14} * e_4^*)$
$e_2' = \sigma(\alpha_{22} * e_2^* + \alpha_{21} * e_1^* + \alpha_{23} * e_3^* + \alpha_{25} * e_5^*)$
Attention layer
$e_1' \parallel e_2'$
$2d'$
One-hot Encoding of $r$
$|R|$
$\mathcal{L}(\Omega)$ Loss
Dense layers

**Fig. 2** The design framework of the proposed TSAE model consists of a triplet subgraph to get neighborhood entities, an attention layer to generate embedding, and a Dense layer to compute the loss

consists of neurons equal to the number of relation types. Finally, compute the categorical cross-entropy loss between the output layer result vector and the one-hot encoding vector of the respective triplet true relation type. A working illustration of the TSAE model for the input triplet (California ($e_1$), *isLocatedIn* ($r$), UnitedState ($e_2$)) is shown in Fig. 2. The triplet subgraph creates a set $E_{sub} = \{e_1, e_2, e_3, e_4, e_5\}$ consisting of triplet entity pair neighborhoods. The attention layer map entity feature vector $\boldsymbol{e}_i \in \mathbb{R}^d$ to $\boldsymbol{e}_i^* \in \mathbb{R}^{d'}$ using $W_T \in \mathbb{R}^{d \times d'}$, computes the attention value $e_{ij}$ for $(e_1, e_2)$ and their neighborhood. Subsequently, update the triplet entity features using the aggregation function to obtain attention-aware features $\boldsymbol{e}_1'$ and $\boldsymbol{e}_2'$ using the normalized attention value $\alpha_{ij}$. Then, the dense layer takes the updated entity feature vector $\boldsymbol{e}_1' || \boldsymbol{e}_2'$ as input and feeds it to the hidden layers to obtain the relation probability at the output layer. Finally, we calculate the training loss between the output layer vector $\hat{\boldsymbol{r}}$ and the one-hot encoding vector $r$.

Based on the above discussion, Algorithm 1 presents the training process of the TSAE model. We first transfer the triplet set $T$, entity set $E$, relation set $R$, and the adjacency matrix $\boldsymbol{A}$ for training to an algorithm with embedding and transformation dimensions $d$ and $d'$, respectively. Xavier uniform approach is used to initialize the vector of dimension $d$ for each entity $e_i \in \mathbb{R}^d$ as follows:

$$\boldsymbol{e}_i = uniform(-\frac{6}{\sqrt{d}}, \frac{6}{\sqrt{d}}) \mid \forall e_i \in E \tag{4}$$

The uniform approach samples each scalar parameter $v_i \in \boldsymbol{e}_i = [v_0, v_1, \ldots, v_d]$ at a time independently from the interval of uniform distribution. Thus, each feature value $v_i$ has equal weightage to define the entity. We initialize each entity $e \in E$ feature vector using the Xavier uniform method, each relation vector $r \in R$ using one-hot encoding, and the transformation matrix $\boldsymbol{W}_T \in \mathbb{R}^{d \times d'}$ using the random uniform method, as shown in lines 1, 2, and 3 of an Algorithm 1, respectively.

**Algorithm 1** TSAE training

```
Require: Entity set $E$, Relation set $R$, Training set $T$, Adjacency matrix $\boldsymbol{A}$, dimension $d$, $d'$
Ensure: Attention-aware feature vectors $\boldsymbol{e}_i'$, Neural network weights matrix $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$
1: $\boldsymbol{e} \in \mathbb{R}^d \leftarrow Uniform(-\frac{6}{\sqrt{d}}, \frac{6}{\sqrt{d}}) \mid \forall e \in E$   ▷ (4)
2: $\boldsymbol{r} \in \{0,1\}^{|R|} \leftarrow OneHotEncoding(r) \mid \forall r \in R$
3: $\boldsymbol{W}_T \in R^{d \times d'}$
4: for $x_i \in T$ do
5:     $\{e_1, r, e_2\} = x_i$
6:     $T_{pair} = \{e_1, e_2\}$
7:     $\mathcal{N}_1 = \{e_1, \forall e_j \in E \wedge \boldsymbol{A}_{e_1 j} == 1\}$   ▷ (2)
8:     $\mathcal{N}_2 = \{e_2, \forall e_j \in E \wedge \boldsymbol{A}_{e_2 j} == 1\}$
9:     $E_{sub} = (\mathcal{N}_1 \cup \mathcal{N}_2)$
10:    for $e_i \in E_{sub}$ do
11:        $e_i^* = LeakyReLU(\boldsymbol{W}_T * e_i)$   ▷ (5)
12:    end for
13:    for $e_i \in T_{pair}$ do
14:        for $e_j \in \mathcal{N}_i$ do
15:            $e_{ij} = \boldsymbol{w}_a(e_i^* \parallel e_j^*)$   ▷ (6)
16:            $sum+ = e_{ij}$
17:        end for
18:        for $e_j \in \mathcal{N}_i$ do
19:            $\alpha_{ij} = \frac{e_{ij}}{sum}$   ▷ (7)
20:            $\boldsymbol{e}_i' + = \alpha_{ij} e_i^*$
21:        end for
22:        $\boldsymbol{e}_i' = \sigma(\boldsymbol{e}_i')$   ▷ (8)
23:    end for
24:    $\hat{\boldsymbol{r}} = \sigma(\boldsymbol{W}_3(LReLU(\boldsymbol{W}_2(LReLU(\boldsymbol{W}_1(\boldsymbol{e}_1' || \boldsymbol{e}_2'))))))$ ▷ LReLU: Leaky Rectified Linear Unit
25:    $\mathcal{L}(\Omega) = -\sum_{i=1}^{|R|} \boldsymbol{r}_i \cdot \log(\hat{\boldsymbol{r}}_i)$   ▷ (10)
26:    Update $\boldsymbol{W}_T, \boldsymbol{w}_a, \boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$ w.r.t gradient of $\mathcal{L}(\Omega)$
27: end for
```

## 3.4 Triplet subgraph

The first part of the model takes the input triplet $(e_1, r, e_2) \in T$ with an adjacency matrix $\boldsymbol{A}$ and generates a triplet subgraph set $E_{sub} = (\mathcal{N}_1 \cup \mathcal{N}_2)$ consisting of triplet entities and their neighborhood entities. We store the triplet entity pair $\{e_1, e_2\}$ into the $T_{pair}$ for later use, as shown in the lines from 5 to 9 of an Algorithm 1.

## 3.5 Entity attention mechanism

All neighborhood entities contribute equally to the information sharing process of the Graph Convolution Network (GCN) [37], which treats all neighboring entities with equal prominence. In response to this shortcoming, the Graph Attention Network (GAT) allocates different levels of prominence to each neighboring entity. As shown in Fig. 2, we incorporated GAT with an updated approach to derive entity features by exploiting the different prominent levels of each component's neighborhood using the attention mechanism. First, we map the entity $e_i \in E_{sub}$ features from lower dimension $d$ to higher dimension $d'$ feature space using a learnable transformation matrix $W_T \in \mathbb{R}^{d \times d'}$ to achieve adequate expressive power as follows:

$$e_i^* = LeakyReLU(W_T * e_i) \tag{5}$$

LeakyReLU restricts the extreme negative values of mapping. Lines 10 to 12 of an Algorithm 1 illustrates the mapping procedure of each entity from the triplet subgraph set according to (5). Attention value represents the comparative importance of each neighborhood entity features to the central target entity, including self-attention. Equation (6) expresses a single GAT layer to compute the attention value $e_{ij}$ between each triplet entity and their neighborhood set $\mathcal{N}_i$ and $\mathcal{N}_j$. Each $e_{ij}$ provides the importance of input edge from neighbor entities $e_j \in \mathcal{N}_i$ to the central entity $e_i$.

$$e_{ij} = \boldsymbol{w}_a(e_i^* \parallel e_j^*) \mid \forall e_j^* \in \mathcal{N}_i \tag{6}$$

where $\boldsymbol{w}_a \in \mathbb{R}^{2d'}$ is a learnable attention weight vector and $\parallel$ concatenation operation, line 15 of an Algorithm 1 illustrates the above step according to (6). Equation (7) normalizes the attention values to obtain the standard scaling across all neighborhoods. The central entity can have nonuniform neighborhood entities in its subgraph structure. Finally, aggregate all normalized attention values with the respective $e_j^*$ to update the features of the triplet entity as follows:

$$\alpha_{ij} = softmax(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{e_{ik} \in \mathcal{N}_i} \exp(e_{ik})} \tag{7}$$

$$e_i' = \sigma\Big(\sum_{j \in \mathcal{N}_i} \alpha_{ij} e_j^*\Big) \tag{8}$$

Line 16 of an Algorithm 1 shows the summation of attention value, and lines 18 to 21 illustrate the normalization and summation of normalized attention value according to (7). Accordingly, line 22 shows the aggregation function according to (8).

## 3.6 Dense neural network layer

The attention layer supplies the attention-aware triplet entity vectors $(e_1', e_2')$ to the next layer. As shown in the last part of Fig. 2 and illustrated in line 24 of the TSAE algorithm, the dense layer (feedforward neural network) combines the triplet entity feature vectors and processes them through the hidden layers to compute the predicted relation score distribution $\hat{\boldsymbol{r}}$. Intentionally, we fixed the number (size) of neurons in each hidden layer as follows to design a generalized dense layer:

$$size(H_{i+1}) = \lceil size(H_i) - ((2d' - |R|)/(|H| + 1)) \rceil \tag{9}$$

where $|H|$ represents the total number of hidden layers and $H_i$ signifies the specific hidden layer. However, the number of hidden layers depends on the model designer. The last output layer consists of neurons equal to the number of relation types in the KG, ie $size(H_{out}) = |R|$. The categorical cross-entropy loss between the target relation one-hot encoding vector $\boldsymbol{r}$ (the true class distribution) and the output vector $H_{out}$ ($\hat{\boldsymbol{r}}$ the predicted relation type score distribution) is calculated as follows:

$$\mathcal{L}(\Omega) = -\sum_{i=1}^{|R|} \boldsymbol{r}_i \cdot \log(\hat{\boldsymbol{r}}_i) \tag{10}$$

Line 15 of an Algorithm 1 illustrates the loss function according to (10), and line 26 shows the parameters updating step according to the gradient of loss.

## 3.7 Fact inference

Figure 3 illustrates the functional structure of the proposed fact inference model as a relation prediction between the given disjoint entity pair "ElonMusk-?-California." The entity Elon Musk and California are given as a subject $(e_1)$ and object $(e_2)$ entity of a missing triplet. The triplet subgraph creates a set $E_{sub} = (\mathcal{N}_1 \cup \mathcal{N}_2)$ consisting of triplet entity pair neighborhoods from the sample adjacency matrix $\boldsymbol{A}$ of entity $e_1$ and $e_2$. The attention layer delivers the updated triplet entity pair feature vectors $e_1'$ and $e_2'$ using (8). The dense layer computes the probability distribution of all the

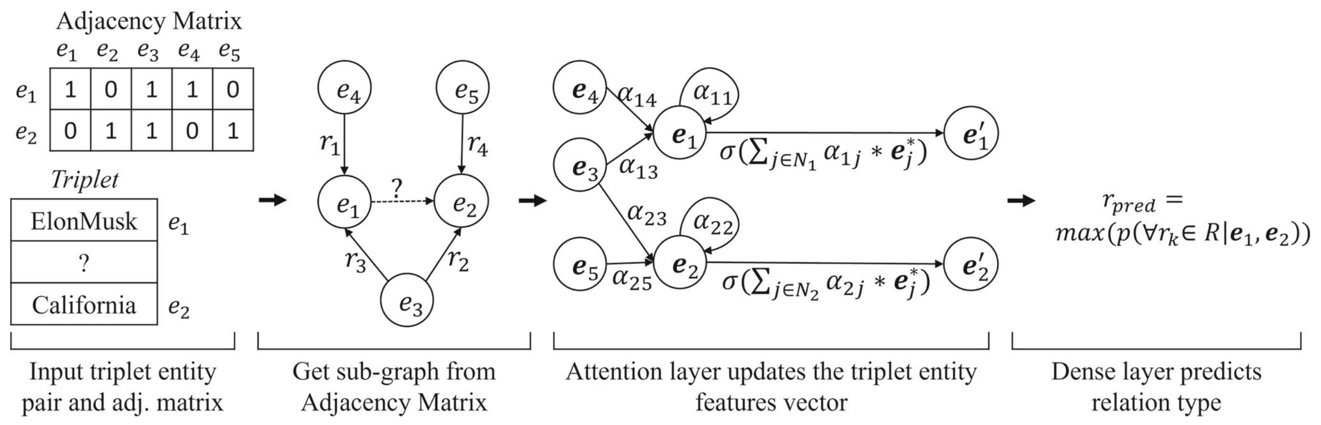

Adjacency Matrix
$e_1$ $e_2$ $e_3$ $e_4$ $e_5$
$e_1$ 1 0 1 1 0
$e_2$ 0 1 1 0 1
Triplet
ElonMusk $e_1$
?
California $e_2$
Input triplet entity pair and adj. matrix
Get sub-graph from Adjacency Matrix
Attention layer updates the triplet entity features vector
$\sigma(\sum_{j\in N_1} \alpha_{1j} * e_j^*)$
$\sigma(\sum_{j\in N_2} \alpha_{2j} * e_j^*)$
$r_{pred} = max(p(\forall r_k \in R|e_1, e_2))$
Dense layer predicts relation type

Fig. 3 Functional structure of fact inference using TSAE

relation types. TSAE infers the true relation type $r_{pred}$ as the highest scored (probability) relation for the given disjoined entity pair $(e_1, e_2)$ using Algorithm 2. We provide the test set $\mathcal{T}$, attention-ware entity set $\boldsymbol{e}'$, and dense layer weight matrices ($\boldsymbol{W}_1$, $\boldsymbol{W}_2$, $\boldsymbol{W}_3$ ) as input to the algorithm. The algorithm predicts the relationship and its ranks for each test triplet to measure the performance of the model. Initially, the predicted relation set, true relation set, and their rank set are initiated to the empty set, as shown in line 1 of an Algorithm 2. Algorithm line 3 obtains the triplet entity pair ($e_1 - e_2$ considered as a disjointed entity pair), then line 4 stores its true relation ($r_k$) in the $TR$ set. The learned entity pair feature vectors $(\boldsymbol{e}'_1, \boldsymbol{e}'_2)$ obtained in lines 5 and 6. The dense layer processes the concatenated entity feature vectors $(\boldsymbol{e}'_1||\boldsymbol{e}'_2)$ using the respective weight matrix in algorithm line 7, which computes the probability (score) of all the relation types in vector $\hat{\boldsymbol{r}} \in \mathbb{R}^{|R|}$. Algorithm lines 8 and 9 are dedicated to storing and sorting the score of each relation type in the intermediate relation score list $RL$ (the relation-wise score of each $r \in R$). Subsequently, line 10 collects the highest score relation in the predicted relation set $PR$, and line 11 collects their rank (index of a relation from $RL$). Steps from lines 3 to 11 are iterated for each triplet belonging to the test set.

## 4 Experimental setup

### 4.1 Data set

This study considers three benchmark data sets well-known for KG completion tasks. Table 1 describes the statistical information about the dataset and data split. WN18-RR is a subset of WN18 [2] derived from WordNet, the standard lexical knowledge graph for the English language. Likewise, FB15k-237 is a subset of FB15k [2] derived from Freebase, an online collection of multi-relational structured data about real-world facts. The WN18 and FB15k datasets have problems with test leakage due to the inverse relationship between the entity pairs, which were later released with the new benchmark WN18RR and FB15k-237 datasets by removing reversible relationships [5]. The YAGO3-10 dataset derived from the YAGO3 [24] contains millions of triplets belonging to people, organizations, and cities, with 37 different types of relationships.

### 4.2 Evaluation metrics

We evaluate the proposed models on the relation prediction task by ranking the score (likelihood) of all the true relation type $r \in R$ for the given triplet entity pair $(e_1, e_2 \in E)$ from the test set $\mathcal{T}$. We use the standard ranking metrics Mean Reciprocal Rank (MRR) and Hit@N

$$MRR = \frac{1}{|\mathcal{T}|} \sum_{(e_1, r_k, e_2)\in\mathcal{T}} \frac{1}{RR_{r_k}} \tag{11}$$

**Algorithm 2** Relation prediction.

```
Require: Test set $\mathcal{T}$, attention-aware entity set $\boldsymbol{e}'$, learned weight matrix
    $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$
Ensure: Predicted relation set $PR$, True relation set $TR$ and their ranks
    $RR$
1: $PR, TR, RR = \{\phi\}$
2: for $x_i \in \mathcal{T}$ do
3:     $e_1, r_k, e_2 = x_i$
4:     $TR = TR \cup \{r_k\}$                          ▷ Collects true triplet relation
5:     $\boldsymbol{e}'_1 \leftarrow e_1$
6:     $\boldsymbol{e}'_2 \leftarrow e_2$
7:     $\hat{\boldsymbol{r}} = \sigma(\boldsymbol{W}_3(LReLU(\boldsymbol{W}_2(LReLU(\boldsymbol{W}_1(\boldsymbol{e}'_1||\boldsymbol{e}'_2))))))$
8:     $RL = \{(i, \hat{r}_i) \mid \forall \hat{r}_i \in \hat{\boldsymbol{r}}\}$          ▷ Store relation and its score
9:     $RL = sort(RL)$                           ▷ Sort the list in descending order
10:    $PR = PR \cup \{RL(0)\}$   ▷ Collect highest score relation present
    at the first position of RL
11:    $RR = RR \cup \{(index(r_k, RL) + 1)\}$ ▷ Collect the rank of each
    triplet true relation type
12: end for
```

$$Hit@N = \frac{1}{|\mathcal{T}|} \sum_{(e_1, r_k, e_2) \in \mathcal{T}} (RR_{r_k} \leq N) \tag{12}$$

where $|\mathcal{T}|$ Indicates triplets count in the test set, and $RR_{r_k}$ is the true relation rank for the given triplet entity pair $(e_1, e_2)$. MRR is the average reciprocal rank, and Hit@N is the average rank within the $N$ cut-off for each true relation type $r_k$. However, we use 1 and 5 cut-offs in the validation and test set. MRR and Hit@$N$ are computed in the range of 0 to 1. The higher MRR and Hit@$N$ indicate superior performance.

### 4.3 Baselines

TransE [2] learns the entity features using the translation distance between the subject and relation features equivalent to the object features. TransH [34] simplifies the polysemy problem of TransE using the projection of an entity in the relationship-specific hyperplane. DistMult [36] reduces the complexity by taking the diagonal of the semantic tensor factor in a scoring function that captures the pairwise interaction among the entities. ComplEx [31] extends the DistMult using complex-valued embedding for entities and relations. ConvE [5], provides the framework to learn entity features using 2D convolution over subject and relation features, then add object entity features to score the triplet. ConvKB [26] learns entity features by concatenating entity pair and relation features for 2D convolution, which has relevance with TransE. DMACM [9] incorporated the directional self-attention network combined with a convolutional layer as an encoder to derive the entity features. GGAE [17] combines the one-hop neighbors and multi-hop neighbors entity features to compute the entity attention.

### 4.4 Implementation details

For reasonable evaluation, we initialized a feature vector of size 250 using Xavier's method, set the training epoch to 1000 with a batch size of 512, and optimized the training process using stochastic gradient descent with a learning rate of 0.0001 for all baselines and TSAE. In the TSAE transformation matrix, $\boldsymbol{W}_T$ size is set to 250 $\times$ 500, LeackyReLU with a negative slope of 0.2, dropout rate 0.5, kernel regularizer, and attention kernel regularizer set to $L_1$ norm. The feedforward neural network consists of fixed input layer $size(H_{inp}) = 1000$ and output layer $size(H_{out}) = |R|$ with two hidden layers of $size(H_i) \in [673, 346]$ set according to (9) for the WN18RR. However, the network consists of 3 hidden layers of $size(H_i) \in [810, 619, 428]$ and $size(H_i) \in [760, 5190, 278]$ for the FB15k-237 and YAGO3-10 respectively. The LeakyReLU activates each hidden layer neuron with a negative slope set to 0.2 and the dropout set to 0.3, and a sigmoid is applied to the output layer. Pairwise margin loss is used to compute the loss during training all the baselines with $L_2$ regularization, a margin set to 5. Moreover, the negative samples are generated by using [13]. ConvKB trains with 100 filters of size $1 \times 3$ initialized by the truncated normal distribution. The baseline models are mainly designed for entity prediction $(? - r - o$ or $s - r - ?)$ for given entity relation pair $(s, r$ or $r, o)$. We redesign the triplet scoring function of baselines to accomplish the task of relation prediction $(s - ? - o)$. We use a PyTorch [28] to design the TSAE and PyKEEN [1] to obtain the results of baseline models with their basic settings.

## 5 Results and discussions

### 5.1 Model evaluation

This section presents the experimental result of TSAE and baselines in the MRR and Hit@N. The underline indicates the second highest, and the bold case indicates the outperforming results. Table 2 illustrates the experimental results of all the models on three data sets. Translational models operate on

**Table 2** Relation prediction results in MRR and Hit@N on WN18RR, FB15k-237, and YAGO3-10 data set

| Models | WN18RR | | | FB15k-237 | | | YAGO3-10 | | |
|---|---|---|---|---|---|---|---|---|---|
| | MRR | Hit@1 | Hit@5 | MRR | Hit@1 | Hit@5 | MRR | Hit@1 | Hit@5 |
| TransE | 0.7436 | 0.6614 | 0.8056 | 0.7992 | 0.6991 | 0.8458 | 0.8309 | 0.7185 | 0.8393 |
| TransH | 0.7749 | 0.6876 | 0.8244 | 0.8134 | 0.7265 | 0.8567 | 0.8563 | 0.7663 | 0.8524 |
| DistMult | 0.8535 | 0.7954 | 0.8957 | 0.8962 | 0.7803 | 0.9038 | 0.8981 | 0.8011 | 0.8931 |
| ComplEx | 0.8743 | 0.8119 | 0.9242 | 0.9132 | 0.8052 | 0.9111 | 0.9126 | 0.8296 | 0.9236 |
| ConvE | 0.8680 | 0.8069 | 0.9285 | 0.9179 | 0.8351 | 0.9380 | 0.9075 | 0.8205 | 0.9297 |
| ConvKB | 0.8730 | 0.8097 | 0.9327 | 0.9058 | 0.8161 | 0.9239 | 0.8701 | 0.7935 | 0.9184 |
| DMACM | 0.8852 | 0.8157 | 0.9661 | 0.9259 | 0.8522 | 0.9394 | 0.9172 | 0.8494 | 0.9448 |
| GGAE | 0.8938 | 0.8228 | **0.9803** | 0.9321 | 0.8676 | 0.9427 | 0.9166 | 0.8481 | 0.9497 |
| TSAE | **0.9172** | **0.8489** | 0.9766 | **0.9530** | **0.8886** | **0.9680** | **0.9381** | **0.8673** | **0.9554** |

**Table 3** Complex relation categorization of WN18RR data set (M represents Many, values in bracket represent relation frequency in the test set)

| Category | Method $e_s - r - e_o$ | Triplet Count | Relations |
|---|---|---|---|
| 1-to-1 | $e_s \leq 1$ and $e_o \leq 1$ | 42 | $\{r_1\_similar\_to$ (3), $r_2\_verb\_group$ (39)$\}$ |
| 1-to-M | $e_s \leq 1$ and $e_o \leq 1$ | 475 | $\{r_3\_member\_meronym$ (253), $r_4\_has\_part$ (172), $r_5\_member\_of\_domain\_region$ (26), $r_6\_member\_of\_domain\_usage$ (24)$\}$ |
| M-to-1 | $e_s \geq 1$ and $eo \leq 1$ | 1487 | $\{r_7\_hypernym$ (1251), $r_8\_instance\_hypernym$ (122), $r_9\_synset\_domain_topic\_of$ (114)$\}$ |
| M-to-M | $e_s \geq 1$ and $e_o \geq 1$ | 1130 | $\{r_{10}\_also\_see$ (56), $r_{11}\_derivationally\_related\_from$ (1074)$\}$ |

triplet-level proximity structure and do not consider graph structure, which results in unsatisfactory performance on all three data sets. Tensor factorization models generate the relation-specific tensor in which each 2D slice captures the relation-specific graph structure. Thus, both tensor factorization models outperform the translational model and are nearly close to the convolutional model. Among them, ComplEx performs better on the WN18RR and YAGO3-10.

The ConvE and ConvKB concatenate relation and entity pair features using the convolution layers and then try to mimic the translation model at the dense layer through activation functions. ConvE performs well on FB15k-237 but performs relatively poor on lower node degree data sets WN18RR and YAGO3-10. Attention-based models DMCM and GGAE perform better than translational, factorization based, and convolutional models. Specifically, GGAE scored the most Hit@5 on WIN18RR. However, TSAE captures the triplet subgraph level proximity, showing superior performance across all data sets. Specifically, it achieves more than 2% relative improvement in MRR and Hit@1 on the WN18RR. Likewise, TSAE achieves 2% relative improvement in MRR and Hit@1 on the FB15k-237 dataset. Furthermore, TSAE achieves 2.1% relative improvement in Hit@1 on the YAGO3-10 dataset.

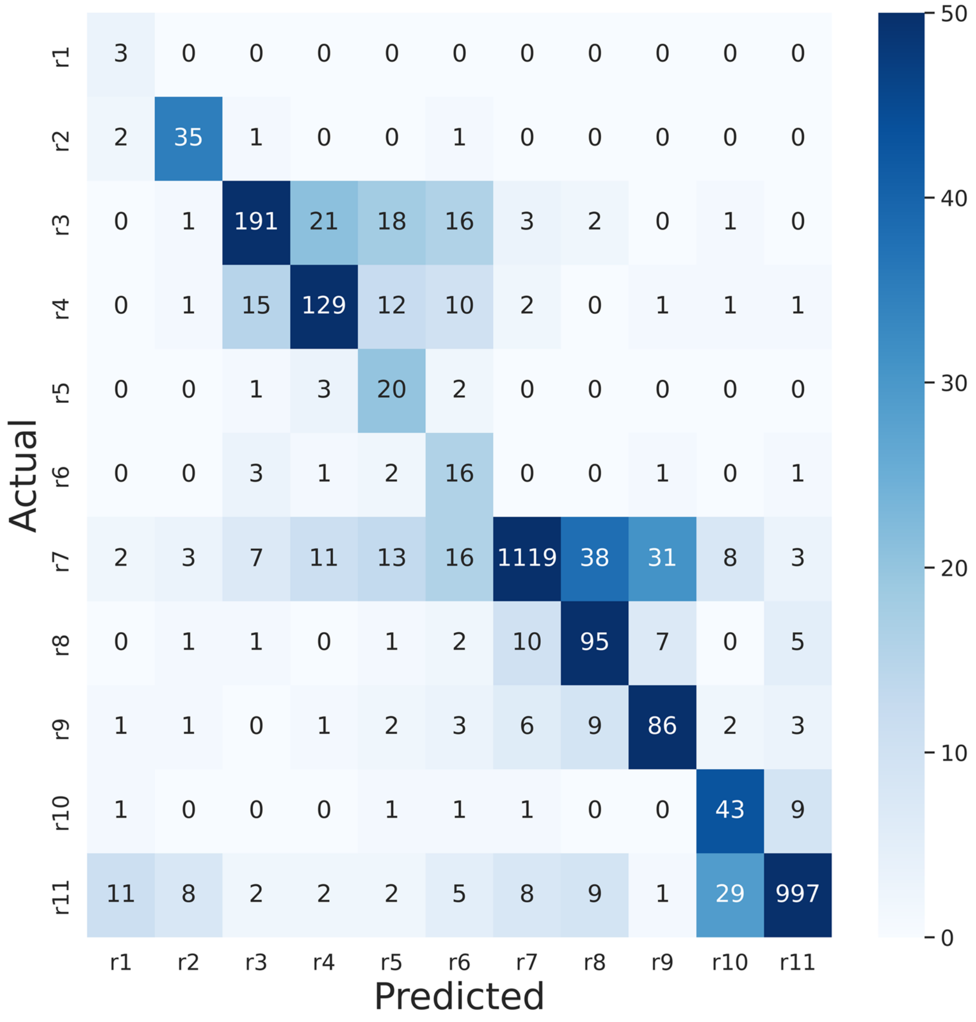
We tested the performance of TSAE on the triplets belonging to the complex relation categories present in the KG, as suggested by [2]. Complex relation categories are generated by finding the average number of entities present in each relation subject ($s$) side and/or object ($o$) side. Table 3 shows complex relation categorization of WN18RR consists of 11 types of relationships. Table 4 presents the category-wise average relation prediction result in Hit@1, in which TSAE achieves superior performance, including 2%, 2.1%, and 2.8% relative improvements in 1-to-Many, Many-to-1 and Many-to-Many relation category over the baseline GGAE respectively. In contrast, GGAE outperformed TSAE in the 1-to-Many relation category.

The WN18RR test set consists of 3134 triplets and relation frequencies stated in Table 3. Relation prediction Algorithm 2 returns the true triplets set ($TR$) and predicted relation set ($PR$) utilized to generate the confusion matrix. Figure 4 illustrates the relation prediction heatmap on the test set triplets. Diagonal elements of the heatmap indicate the predicted true relation, where TSAE achieves superior correct relation type prediction from lowest 75% (129 out of 172) for "$r_4\_has\_part$" to highest 100% (3 out of 3) for "$r_1\_similar\_to$". However, false relationship-type predictions belong mainly to the same complex relation categories, as shown in Fig. 4.

We recorded the MRR and the loss during the training process after every five epochs for all the models on the validation set. We have stored the best model during the training process on the MRR criteria, which is then utilized to evaluate the model on the test set. Figure 5a shows experimental results of model loss versus epochs. TSAE achieves the best outcome in 600 epochs. Other models take more epochs to update entity features for the best results. TSAE achieves an MRR of up to 0.85 in the first 355 epochs, as shown in Fig. 5b, then saturates with minor performance changes. Attention mechanisms improve relational prediction performance by providing additional support for entity feature learning.

**Table 4** Complex relation category-wise results of relation prediction in Hit@1 on WN18RR data set

| Models | Hit@1 | | | |
|---|---|---|---|---|
| | 1-TO-1 | 1-TO-M | M-TO-1 | M-TO-M |
| TransE | 0.7143 | 0.5579 | 0.6859 | 0.6876 |
| TransH | 0.7381 | 0.5684 | 0.7196 | 0.7035 |
| DistMult | 0.7857 | 0.6737 | 0.7935 | 0.8761 |
| ComplEx | 0.8333 | 0.6905 | 0.8339 | 0.8850 |
| ConvE | 0.8333 | 0.7368 | 0.8070 | 0.8496 |
| ConvKB | 0.8095 | 0.6947 | 0.8272 | 0.8584 |
| DMACM | 0.8598 | 0.7408 | 0.8386 | 0.8812 |
| GGAE | 0.8643 | **0.7516** | 0.8559 | 0.8953 |
| TSAE | **0.8810** | 0.7495 | **0.8742** | **0.9204** |



| | r1 | r2 | r3 | r4 | r5 | r6 | r7 | r8 | r9 | r10 | r11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| r1 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| r2 | 2 | 35 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| r3 | 0 | 1 | 191 | 21 | 18 | 16 | 3 | 2 | 0 | 1 | 0 |
| r4 | 0 | 1 | 15 | 129 | 12 | 10 | 2 | 0 | 1 | 1 | 1 |
| r5 | 0 | 0 | 1 | 3 | 20 | 2 | 0 | 0 | 0 | 0 | 0 |
| r6 | 0 | 0 | 3 | 1 | 2 | 16 | 0 | 0 | 1 | 0 | 1 |
| r7 | 2 | 3 | 7 | 11 | 13 | 16 | 1119 | 38 | 31 | 8 | 3 |
| r8 | 0 | 1 | 1 | 0 | 1 | 2 | 10 | 95 | 7 | 0 | 5 |
| r9 | 1 | 1 | 0 | 1 | 2 | 3 | 6 | 9 | 86 | 2 | 3 |
| r10 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 43 | 9 |
| r11 | 11 | 8 | 2 | 2 | 2 | 5 | 8 | 9 | 1 | 29 | 997 |

**Fig. 4** Heatmap of actual versus predicted relations for each triplet from a test set of WN18RR

## 5.2 Model analysis

This part presents the impact of model design parameters on TSAE performance in Hit@1 as a primary metric for relation prediction. TSAE utilizes the summation operation in the aggregation function to update the features of the entity, given in (8). Furthermore, we conducted experiments with average and max aggregation functions in TSAE to investigate the significance of other aggregation functions, given in (13) and (14), respectively. The results of these aggregation functions on three data sets are shown in Table 5. Consistent performance of TSAE (*avg*) and TSAE (*sum*) can be seen on the WN18RR dataset. Likewise, a similar performance can be seen on FB15kk-237. Also, TSAE (*avg*) and TSAE

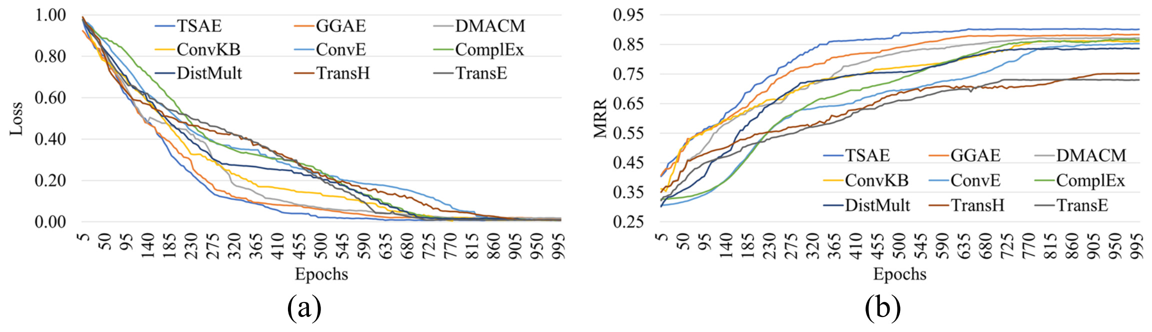


**Fig. 5** Convergence of TSAE and baselines during training on the WN18RR dataset recorded for (a) Loss and (b) MRR

**Table 5** Effect of aggregation function on TSAE in Hit@1

| Aggregation Function | Hit@1 | | |
|---|---|---|---|
| | WN18RR | FB15k-237 | YAGO3-10 |
| TSAE (*avg*) | 0.8402 | 0.8814 | 0.8532 |
| TSAE (*max*) | 0.8374 | 0.8778 | 0.8558 |
| TSAE (*sum*) | 0.8498 | 0.8886 | 0.8673 |

**Table 6** Hidden layer size effects on the TSAE performance in Hit@1 on WN18RR data set

| $\|H\|$ | $size(H_i)$ | Hit@1 |
|---|---|---|
| 1 | [509] | 0.8273 |
| 2 | [673, 346] | 0.8498 |
| 3 | [755, 510, 265] | 0.8508 |
| 4 | [804, 608, 412, 216] | 0.8567 |

(*max*) performance are similar on YAGO3-10, but TSAE (*sum*) achieves a 1% relative improvement.

$$e'_i = \sigma\Big(\frac{1}{|\mathcal{N}_i|}\sum_{j\in\mathcal{N}_i}\alpha_{ij}e^*_j\Big) \tag{13}$$

$$e'_i = \sigma\Big(max(\alpha_{ij}e^*_j \mid j \in \mathcal{N}_i)\Big) \tag{14}$$

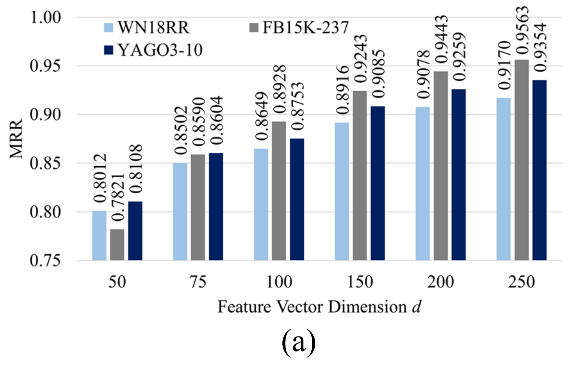
The features vector dimension plays a vital role in KG embedding. To analyze the effect of the vector dimensions, we performed several experiments with dimensions $d \in [50, 75, 100, 150, 200, 250]$ on the three data sets. Figure 6 shows the MRR and Hit@1 bar charts to demonstrate the effect of vector size on TSAE. It can be seen that the performance of TSAE in MRR and Hit@1 improves with increasing dimension size. However, TSAE performs satisfactorily at $d = 250$ on all data sets. Due to resource limitations, we have experimented with dimensions up to 250, but performance can be further improved for larger sizes.

The transformation matrix $W_T \in R^{d\times d'}$ provides sufficient expressive power to TSAE. Consequently, to investigate the importance of the transformation matrix on TSAE performance, we conducted some experiments with different matrix dimensions. Taking into account previous investigations, we fixed $d = 250$ and chose $d' \in [300, 400, 500, 600, 700, 800]$ for experiments. The general results are plotted in Fig. 7, which shows the effect of $d'$ on Hit@1 over the three data sets. It can be found that TSAE achieves superior performance over the baseline up to 500 dimensions. Later, performance saturates at higher values from 600 to 700, slightly improving at 800.

The dense layer computes the relation likelihood for the given triplet entity pair. However, the number of hidden layers gradually affect the learning model prediction performance. Subsequently, we perform the experiments with $H_i \in [1, 2, 3, 4]$ for the fixed input layer $size(H_{inp}) = 1000$ (concatenates $\boldsymbol{e}'_1, \boldsymbol{e}'_2 \in \mathbb{R}^{500}$) and output layer $size(H_{out}) = |R|$ (number of existing relations). Table 6 shows the TSAE sensitivity over the number of hidden layers and their size is calculated using (9). TSAE with two hidden layers on the WN18RR gives superior results, afterwards Hit@1 is saturated with barely small improvement up to four layers. TSAE performs decently with two hidden layers but achieves 2% and 3% relative improvement with three hidden layers on the FB15k-237 and YAGO3-10 data sets.

## 5.3 Case study

To demonstrate the effectiveness of TSAE for completing knowledge graphs, we present a case study on the YAGO3-10 test set. Table 7 shows the relation prediction results in Hit@5 for six entity pairs from the test set. The TSAE results for each pair of entities are arranged in descending order of the predicted score. The bold case indicates the ground-truth relationship type between given entity pairs. The TSAE predicts the true relationship type *isConnectedTo* at a first rank (higher score) for the first entity pair. Similarly, identical results can be observed for the next three entity pairs.



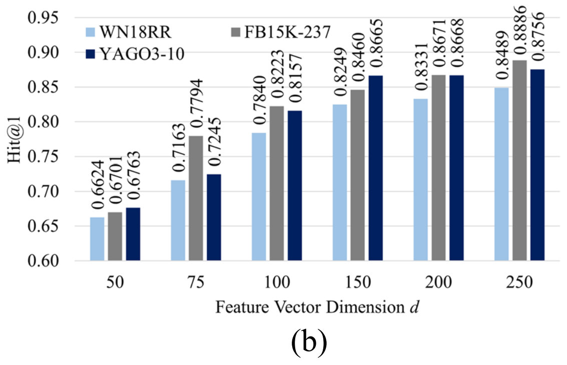
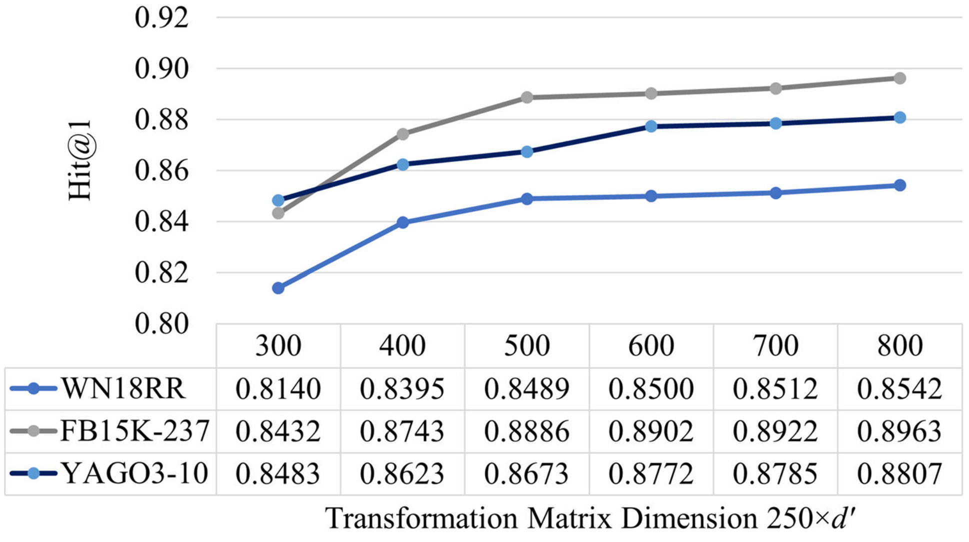


**Fig. 6** Features vector dimension *d* effects on the TSAE performance in (a) MRR and (b) Hit@1



| | 300 | 400 | 500 | 600 | 700 | 800 |
|---|---|---|---|---|---|---|
| WN18RR | 0.8140 | 0.8395 | 0.8489 | 0.8500 | 0.8512 | 0.8542 |
| FB15K-237 | 0.8432 | 0.8743 | 0.8886 | 0.8902 | 0.8922 | 0.8963 |
| YAGO3-10 | 0.8483 | 0.8623 | 0.8673 | 0.8772 | 0.8785 | 0.8807 |

**Fig. 7** Transformation matrix dimension $250 \times d'$ effects on the TSAE performance in Hit@1

However, TSAE predicts lower scores for true relationships between given fifth and sixth triplet entity pairs due to less frequent entities in the training set, such as Moscow and Vinay_Pathak.

## 6 Conclusion

In this paper, we propose TSAE, a novel embedding model for enhancing missing facts inference in knowledge graphs. TSAE combines the simplest form of an attention mechanism with a neural network. The attention layer computes the attention value between the entity pair and their neighborhood. Subsequently, it updates the triplet entity feature by aggregating neighborhood attention features. The neural network concatenates and processes the attention-aware triplet entity features to compute the likelihood of relation types. Experimental results have shown that TSAE performs better than baselines in standard metrics. Furthermore, we investigate the model sensitivity to training parameters. A case study illustrates the effectiveness and efficiency of TSAE in completing knowledge graphs. TSAE is practical for applications where reasonableness, simplicity, and interpretability are important.

KGs are often sparse, and TSAE might struggle with less frequent (long-tail) entities because they tend to receive less attention than frequent entities. It needs to be investigated by incorporating adaptive attention mechanisms that give more weight to less frequent entities. TSAE can be prone to overfitting, especially when dealing with noisy or corrupted KGs. Future research can focus on regularization techniques, such as dropout or weight decay to mitigate overfitting in attention-based models. Many real-world KGs are dynamic, and the information changes over time. TSAE might not naturally handle temporal aspects. It is necessary to explore ways

**Table 7** Case study on relation prediction in Hit@5 on YAGO3-10 data set

| Triplet Entity Pair | Predicted Relation Type |
|---|---|
| RajivGandhiInternationalAirport, KuwaitInternationalAirport | (1) ***isConnectedTo***; (2) *isLocatedIn*; (3) *wasBornIn*; (4) *isInterestedIn*; (5) *owns* |
| HolyRomanEmpire, ItalianWarof1536-38 | (1) ***participatedIn***; (2) *isKnownFor*; (3) *hasCurrency*; (4) *happenedIn*; (5) *diedIn* |
| EdwinCongo, Male | (1) ***hasGender***; (2) *wasBornIn*; (3) *diedIn*; (4) *isAffiliatedTo*; (5) *playsFor* |
| Yahoo!, SunnyvaleCalifornia | (1) ***isLocatedIn***; (2) *isAffiliatedTo*; (3) *playsFor*; (4) *wasBornIn*; (5) *isLeaderOf* |
| Leonid_Musin, Moscow | (1) *isAffiliatedTo*; (2) *livesIn*; (3) ***wasBornIn***; (4) *happenedIn*; (5) *dealsWith* |
| Vinay_Pathak, Johnny_Gaddaar | (1) *influences*; (2) *worksAt*; (3) *hasNeighbor*; (4) ***actedIn***; (5) *isInterestedIn* |

to incorporate temporal information by combining attention mechanisms with other techniques, such as adversarial learning, reinforcement learning, and transfer learning.

**Acknowledgements** The authors thank the Department of Computer Science and Engineering, COEP Technological University, for supporting us in using the GPU server facility purchased under TEQIP-III (a World Bank initiative).

**Author Contributions** Conceptualization: Anish Khobragade, Shashikant Ghumbre; Methodology: Anish Khobragade; Validation, formal analysis, and investigation: Anish Khobragade, Shashikant Ghumbre; Resources: Vinod Pachghare; Article original draft preparation: Anish Khobragade; Article review and editing: Anish Khobragade, Vinod Pachghare; Supervision: Shashikant Ghumbre, Vinod Pachghare. All authors have read and agreed to the published version of the manuscript.

**Funding** No funding was received to carry out this study.

**Code Availibility** The code and data used in this study will be made available from the corresponding author on reasonable request.

## Declarations

**Ethical and informed consent for data used** The datasets used for this experiment are publicly available by the respective organizations/authors to further improve the knowledge graph research field. Thus, informed consent is not required to use the dataset. References and citations to relevant datasets are included in the manuscript.

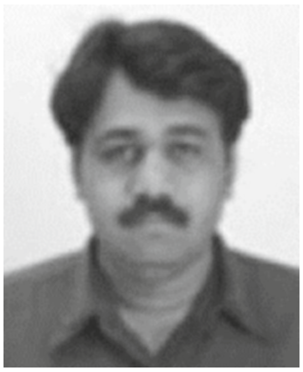
**Conflict of interests** The authors declare no conflict of interest.

## References

1. Ali M, Berrendorf M, Hoyt CT et al. (2021) PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings. J Mach Learn Res 22(82):1–6
2. Bordes A, Usunier N, Garcia-Duran A et al. (2013) Translating embeddings for modeling multi-relational data. Adv Neural Infor Process Sys 26
3. Cai L, Wang WY (2018) Kbgan: Adversarial learning for knowledge graph embeddings. In: Proc. 16th Annual Conf. NAACL Human Lang. Tech
4. Destandau M, Fekete JD (2021) The missing path: Analysing incompleteness in knowledge graphs. Inf Visualizat 20(1):66–82
5. Dettmers T, Minervini P, Stenetorp P et al (2018) Convolutional 2d knowledge graph embeddings. In: Proc. AAAI Conf. on AI
6. Ebisu T, Ichise R (2019) Generalized translation-based embedding of knowledge graph. IEEE Trans Knowl and Data Engg 32(5):941–951
7. Hou X, Ma R, Yan L et al. (2023) T-gae: A timespan-aware graph attention-based embedding model for temporal knowledge graph completion. Inf Sci 119225
8. Hsu PY, Chen CT, Chou C et al. (2022) Explainable mutual fund recommendation system developed based on knowledge graph embeddings. Appl Intell 1–26
9. Huang J, Zhang T, Zhu J et al (2021) A deep embedding model for knowledge graph completion based on attention mechanism. Neural Comput Appl 33(15):9751–9760
10. Ji G, He S, Xu L et al. (2015) Knowledge graph embedding via dynamic mapping matrix. In: Proce. 7th Int. Joint Conf. NLP, pp 687–696
11. Ji K, Hui B, Luo G (2020) Graph attention networks with local structure awareness for knowledge graph completion. IEEE Access 8:224860–224870
12. Jiang X, Wang Q, Wang B (2019) Adaptive convolution for multi-relational learning. Proc. NACACL, Human Lang. Tech., pp 978–987
13. Khobragade A, Mahajan R, Langi H et al (2022) Effective negative triplet sampling for knowledge graph embedding. Jour Info and Optim Sci 43(8):2075–2087
14. Khobragade A, Ghumbre S, Pachghare V (2023) Infer the missing facts of d3fend using knowledge graph representation learning. Int J Web Inf Syst
15. Khobragade AR, Ghumbre SU (2022) Study and analysis of various link predictions in knowledge graph: A challenging overview. Intell Decis Tech 16(4):653–663
16. Li C, Peng X, Niu Y et al (2021) Learning graph attention-aware knowledge graph embedding. Neurocomputing 461:516–529
17. Li Q, Wang D, Feng S et al (2021) Global graph attention embedding network for relation prediction in knowledge graphs. IEEE Tran Neural Netw and Lear Sys 33(11):6712–6725
18. Li W, Zhang X, Wang Y et al (2019) Graph2seq: Fusion embedding learning for knowledge graph completion. IEEE Access 7:157960–157971
19. Li W, Peng R, Li Z (2021) Knowledge graph completion by jointly learning structural features and soft logical rules. IEEE Trans Knowl Data Eng
20. Li Z, Liu H, Zhang Z et al (2021) Learning knowledge graph embedding with heterogeneous relation attention networks. IEEE Tran Neural Netw and Lear Sys 33(8):3961–3973
21. Lin Y, Liu Z, Sun M et al. (2015) Learning entity and relation embeddings for knowledge graph completion. In: Proc. AAAI Conf. on AI
22. Liu S, Tan N, Yang H et al (2021) An intelligent question answering system of the liao dynasty based on knowledge graph. Inter Jour Computat Intelli Sys 14:1–12
23. Maddalena L, Giordano M, Manzo M, et al (2022) Whole-graph embedding and adversarial attacks for life sciences. In: Trends in Biomathematics: Stability and Oscillations in Environmental, Social, and Biological Models: BIOMAT 2021. Springer, p 1–21
24. Mahdisoltani F, Biega J, Suchanek F (2014) Yago3: A knowledge base from multilingual wikipedias. In: Proc. 7th Conference on CIDR, CIDR Conference
25. Nathani D, Chauhan J, Sharma C et al. (2019) Learning attention-based embeddings for relation prediction in knowledge graphs. In: Proc. 57th Annual Meeting of the ACL, pp 4710–4723
26. Nguyen DQ, Nguyen TD, Nguyen DQ et al (2018) A novel embedding model for knowledge base completion based on convolutional neural network. Proc. NACACL, Human Lang. Tech., pp 327–333
27. Nickel M, Tresp V, Kriegel HP et al. (2011) A three-way model for collective learning on multi-relational data. In: Proc. ICML, pp 3104482–3104584
28. Paszke A, Gross S, Massa F et al. (2019) Pytorch: An imperative style, high-performance deep learning library. In: Proceedings of Neural Information Processing Systems. Curran Associates Inc., pp 8024–8035
29. Song D, Zhang F, Lu M et al (2021) Dtranse: Distributed translating embedding for knowledge graph. IEEE Trans Parallel and Distributed Sys 32(10):2509–2523
30. Song HJ, Park SB (2018) Enriching translation-based knowledge graph embeddings through continual learning. IEEE Access 6:60489–60497
31. Trouillon T, Welbl J, Riedel S et al (2016) Complex embeddings for simple link prediction. Proc. Int. Conf. ML, PMLR, pp 2071–2080
32. Veličković P, Cucurull G, Casanova A et al. (2018) Graph attention networks. In: Proc. 6th Int. Conf. Learn. Represent

33. Wang J, Zhang Z, Shi Z et al (2022) Duality-induced regularizer for semantic matching knowledge graph embeddings. IEEE Trans Pattern Ana and ML 45(2):1652–1667
34. Wang Z, Zhang J, Feng J et al. (2014) Knowledge graph embedding by translating on hyperplanes. In: Proc. AAAI Conf. on AI
35. Wangde F, Khobragade A, Shinde O (2022) Analysis of translational and tensor factorization knowledge graph embedding models. In: Proc. 6th Int. Conf. Comput., Comm., Cont. Autom., IEEE, pp 1–5
36. Yang B, Yih Wt, He X et al. (2014) Embedding entities and relations for learning and inference in knowledge bases. In: Proc. ICLR
37. Yu R, Wang L, Xin Y et al. (2023) A gated graph attention network based on dual graph convolution for node embedding. Applied Intelligence pp 1–14
38. Zhang R, Trisedya BD, Li M et al (2022) A benchmark and comprehensive survey on knowledge graph entity alignment via representation learning. VLDB J 31(5):1143–1168
39. Zhang S, Sun Z, Zhang W (2020) Improve the translational distance models for knowledge graph embedding. J Intell Inf Syst 55:445–467
40. Zhang Z, Li Z, Liu H et al (2020) Multi-scale dynamic convolutional network for knowledge graph embedding. IEEE Trans Know and Data Engg 34(5):2335–2347
41. Zhang Z, Huang J, Tan Q (2022) Association rules enhanced knowledge graph attention network. Knowl-Based Syst 239:108038

**Anish Khobragade** received the MTech. degree from the Government College of Engineering Amravati, India, in 2018. He is currently pursuing the Ph.D. degree at Savitribai Phule Pune University. He is currently an Assistant Professor with the Department of Computer Science and Engineering, COEP Technological University, Pune, India. His current research interests include artificial intelligence, knowledge graph, and representation learning.



**Shashikant Ghumbre** received a Ph.D. in Computer Engineering from the Savitribai Phule Pune University in 2012. Since June 2016, he has been a Professor with the Government College of Engineering and Research, Avasari Khurd, Pune, and previously worked as an Associate Professor at the College of Engineering, Pune, India. His research addresses artificial intelligence for medical disease diagnosis. He has authored various articles in international journal papers and conference proceedings.

**Vinod Pachghare** received a Ph.D. in Computer Engineering from the Savitribai Phule Pune University in 2013. He is an Associate Professor in the Department of Computer Science and Engineering, College of Engineering, Pune, India. He has authored research articles in various international journals. His area of research is network security. He is an investigator for the Information Security Education and Awareness [ISEA] Project, Ministry of Information Technology, Govt. of India. He was a Principal Investigator for a research project, "Wireless IDS," sponsored by AICTE, India.