Disentangled Ontology Embedding for Zero-Shot Learning

Citation for published version:
Geng, Y, Chen, J, Zhang, W, Xu, Y, Chen, Z, Z. Pan, J, Huang, Y, Xiong, F & Chen, H 2022, Disentangled Ontology Embedding for Zero-Shot Learning. in Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. KDD ’22, Association for Computing Machinery, Inc, New York, NY, USA, pp. 443–453, The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington D.C., District of Columbia, United States, 14/08/22. https://doi.org/10.1145/3534678.3539453

Digital Object Identifier (DOI):
10.1145/3534678.3539453

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published in:
Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
Disentangled Ontology Embedding for Zero-shot Learning

Yuxia Geng
gengyx@zju.edu.cn
College of Computer Science and Technology, Zhejiang University
Hangzhou, China

Jiaoyan Chen
jiaoyan.chen@cs.ox.ac.uk
Department of Computer Science, University of Oxford
Oxford, United Kingdom

Wen Zhang
zhang.wen@zju.edu.cn
School of Software Technology, Zhejiang University
Ningbo, China

Yajing Xu
yajingxu@zju.edu.cn
School of Software Technology, Zhejiang University
Ningbo, China

Zhuo Chen
zhuo.chen@zju.edu.cn
Department of Computer Science and Technology, Zhejiang University
Hangzhou, China

Jeff Z. Pan
j.z.pan@ed.ac.uk
School of Informatics, The University of Edinburgh
Edinburgh, United Kingdom

Yufeng Huang
huangyufeng@zju.edu.cn
School of Software Technology, Zhejiang University
Ningbo, China

Feiyu Xiong
feiyu.xfy@alibaba-inc.com
Alibaba Group
Hangzhou, China

Huajun Chen∗
huajunsir@zju.edu.cn
College of Computer Science and Technology, Zhejiang University
ZJU-Hangzhou Global Scientific and Technological Innovation Center
Alibaba-Zhejiang University Joint Institute of Frontier Technologies

ABSTRACT

Knowledge Graph (KG) and its variant of ontology have been widely used for knowledge representation, and have shown to be quite effective in augmenting Zero-shot Learning (ZSL). However, existing ZSL methods that utilize KGs all neglect the intrinsic complexity of inter-class relationships represented in KGs. One typical feature is that a class is often related to other classes in different semantic aspects. In this paper, we focus on ontologies for augmenting ZSL, and propose to learn disentangled ontology embeddings guided by ontology properties to capture and utilize more fine-grained class relationships in different aspects. We also contribute a new ZSL framework named DOZSL, which contains two new ZSL solutions based on generative models and graph propagation models, respectively, for effectively utilizing the disentangled ontology embeddings. Extensive evaluations have been conducted on five benchmarks across zero-shot image classification (ZS-IMGC) and zero-shot KG completion (ZS-KGC). DOZSL often achieves better performance than the state-of-the-art, and its components have been verified by ablation studies and case studies. Our codes and datasets are available at https://github.com/zjkug/DOZSL.

∗Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD ’22, August 14–18, 2022, Washington, DC, USA.
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9385-0/22/08...
https://doi.org/10.1145/3534678.3539453

CCS CONCEPTS
- Computing methodologies → Artificial intelligence.

KEYWORDS
Zero-shot Learning, Ontology, Knowledge Graph, Disentangled Representation Learning

ACM Reference Format:
Yuxia Geng, Jiaoyan Chen, Wen Zhang, Yajing Xu, Zhuo Chen, Jeff Z. Pan, Yufeng Huang, Feiyu Xiong, and Huajun Chen. 2022. Disentangled Ontology Embedding for Zero-shot Learning. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD ’22), August 14–18, 2022, Washington, DC, USA. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539453

1 INTRODUCTION

Zero-shot Learning (ZSL), which enables models to predict new classes that have no training samples (i.e., unseen classes), has attracted a lot of research interests in many machine learning tasks, such as image classification [7, 36], relation extraction [20] and Knowledge Graph (KG) completion [25, 31]. To handle these unseen classes, most existing ZSL methods adopt a knowledge transfer strategy: transferring samples, sample features or model parameters from the classes that have training samples (i.e., seen classes) to these unseen classes, with the guidance of some auxiliary information which usually depicts the relationships between classes. For example, in zero-shot image classification (ZS-IMGC), some studies utilize visual attributes of objects to transfer image features learned from seen classes to unseen classes and build classifiers for the later [19, 37]. Other popular auxiliary information includes class’s literal name [7], textual descriptions [25, 40] and so on.

Recently, more and more studies leverage KG [14, 24], an increasingly popular solution for managing graph structured data,
to represent complex auxiliary information for augmenting ZSL [2]. KGs that are composed of relational facts can model diverse relationships between classes. For example, Wang et al. [34] incorporate class hierarchies from a lexical KG named WordNet [23]; works such as [10, 26] explore common sense class knowledge from ConceptNet [28]. As a kind of KGs, ontologies, also known as ontology schemas when they act as parts of KGs for meta information, can represent more complex and logical inter-class relationships. For example, Chen et al. [3] use an ontology in OWL1 to express the compositionality of classes; Geng et al. [9] define the domain and range constraints of KG relations using ontological schemas, as shown in Figure 1 (b). In addition, ontologies are also able to represent and integrate traditional auxiliary information such as attributes and textual descriptions. For example, as Figure 1 (a) shows, animal visual attributes with binary values can be represented in graph with the attributes transformed into entities.

To exploit these KGs, two ZSL paradigms have been widely investigated. One is a pipeline including two main steps. Firstly, the KG is embedded, based on which the ZSL classes that are already aligned with KG entities are represented using vectors with their relationships kept in the vector space. Second, a compatibility function between the class vector and the sample input (or features) is learned. It can either be a mapping function, which projects the sample input and the class vector into the same space such that a testing sample can be matched with an arbitrary class via e.g., Euclidean distance [3, 7, 20], or a generative model, which generates labeled samples or features for unseen classes [9, 25]. The other paradigm is based on graph information propagation. It often uses Graph Neural Networks (GNNs) to propagate classifier parameters or sample features from nodes of seen classes to nodes of unseen classes [4, 16, 34]. Methods of both paradigms, together with KGs, always lead to state-of-the-art performance on many ZSL tasks.

Nevertheless, existing methods of both paradigms still have big space for improvement. In a real-world KG, an entity is often linked to other entities for knowledge of different aspects. For example, Kobe Bryant is connected to NBA teams for his career knowledge, and connected to his daughters for family knowledge. This also happens in those KGs (especially ontologies) used for augmenting ZSL. As shown in Figure 1 (a), Zebra is connected to Horse via rdfs:subClassOf for knowledge on taxonomy, and connected to Tiger and Panda via imgc:hasAttribute for knowledge on visual characteristics. Thus the vector representation of Zebra should be closer to Horse than Tiger and Panda considering the aspect of taxonomy, and be closer to Tiger and Panda than Horse considering the visual characteristics. The existing KG-based ZSL methods all neglect this important KG characteristic on entanglement, which prevents them from capturing more fine-grained inter-class relationships in different aspects, and limits their performance.

In this work, we focused on augmenting ZSL by ontologies, proposed to investigate Disentangled Ontology embeddings and developed a general ZSL framework named DOZSL. DOZSL first learns multiple disentangled vector representations (embeddings) for each class according to its semantics of different aspects defined in an ontology, where a new disentangled embedding method with ontology property-aware neighborhood aggregation and triple scoring is proposed, and then adopts an entangled ZSL learner, which builds upon a Generative Adversarial Network (GAN)-based generative model and a Graph Convolutional Network (GCN)-based graph propagation model, respectively, to incorporate these disentangled class representations. To apply the generative model, we concatenate the disentangled representations; while to apply the propagation model, we generate one graph for semantics of one aspect with the disentangled representations. We evaluate DOZSL with five datasets of zero-shot image classification (ZS-IMGC) and zero-shot KG completion (ZS-KGC). See Figure 1 for segments of the ontology for one IMGC dataset and the ontological schema for a KG to complete. In summary, our contributions are the following:

- To the best of our knowledge, this is among the first to investigate disentangled semantic embeddings for ZSL.
- A property guided disentangled embedding method is developed for ontologies used in ZSL, and a general ZSL framework named DOZSL, which is able to support both generative models and propagation models, is proposed.
- The work includes extensive evaluation, where DOZSL often outperforms the baselines including the state-of-the-art methods on five datasets of two tasks, and the effectiveness of DOZSL’s components is verified by ablation studies.

2 PRELIMINARIES AND RELATED WORK

2.1 Zero-shot Learning

2.1.1 Zero-shot Image Classification (ZS-IMGC). ZS-IMGC has been thoroughly studied in Computer Vision for image classification with new classes whose images are not seen during training. Formally, let $D_{tr} = \{(x, y)|x \in X_s, y \in Y_s\}$ be the training set, where $x$ is the CNN features of a training image and $y$ is its class in $Y_s$, which is a set of seen classes, and $D_{te} = \{(x, y)|x \in X_u, y \in Y_u\}$ be the testing set, where $Y_u$ is a set of unseen classes, has no overlap with $Y_s$. Given $D_{tr}$ and some auxiliary information $A$ for describing the relationships between seen and unseen classes, ZS-IMGC aims to learn a classifier for each unseen class. There are often two evaluation settings: standard ZSL which recognizes the testing samples in $X_u$ by only searching in $Y_u$ and generalized ZSL which recognizes the testing samples in $X_u \cup X_s$ by searching in $Y_u \cup Y_s$.

1 Web Ontology Language (https://www.w3.org/TR/owl-features/)
Widely used auxiliary information includes class attributes [19, 37, 38], textual information [7, 40] and KGs [6, 9, 26, 34]. To support ZSL, they are often embedded to generate one semantic vector for each class, such as binary/numerical attribute vectors, pre-trained word embeddings, learnable sentence embeddings, and KG embeddings. Next, a compatibility function between the class vectors and the vector representations of samples is often learned to conduct knowledge transfer. Mapping function is a typical practice, which maps the image features to the space of class vector [3, 7, 19] or vice versa [39] or to a shared common space [8]. However, all of these mappings are trained by seen data, and thus have a strong bias towards seen classes during prediction, especially in generalized ZSL. Recently, thanks to generative models such as GANs [12], several methods [37, 40] have been proposed to synthesize samples (or features) for unseen classes conditioned on their class vectors. This converts the ZSL problem to a standard supervised learning problem with the aforementioned bias issue alleviated.

Besides, to explicitly exploit the structural inter-class relationships that exist in a KG, some ZSL works explore a graph information propagation strategy. In these works, classes are often aligned with KG entities, and a powerful GNN such as GCN [17] is then trained to output a classifier (i.e., a class-specific parameter vector) for each class, through which the classifiers of unseen classes are approximated by aggregating the classifiers of seen classes. One typical work is by Wang et al. [34], the subsequent works adopt similar ideas but vary in optimizing the graph propagation [11, 16]. Especially, some of them consider the multiple types of relations in the KGs by developing multi-relational GCN [4], or splitting the multi-relation KGs into multiple single-relation graphs and applying several parameter-shared GCNs to propagate features [32].

### 2.1.2 Zero-shot KG Completion (ZS-KGC)

In this task, a KG composed of relational facts is to be completed. It is denoted as \( G = (E, R, T) \), where \( E \) is a set of entities, \( R \) is a set of relations, and \( T = \{(h, r, t) | h, t \in E; r \in R\} \) is a set of relational facts in form of RDF triple. The completion is to predict a missing but plausible triple with two of \( h, r, t \) given. Typical KGC methods first embed entities and relations into vector spaces (i.e., \( x_h, x_r, \) and \( x_t \)) and conduct vector computations to discover missing triples. The embeddings are trained by existing triples and assume all testing entities and relations into vector spaces (i.e., head entity types) and ranges (i.e., tail entity types) of KG relations. Note we sometimes also call concept as concept node in introducing ontology embedding.

Ontology properties can be either built-in properties of RDFS, such as `rdfs:subClassOf` and `rdfs:subPropertyOf`, or user defined for a specific task, such as `imgc:hasAttribute`. Figure 1 shows two ontology segments for ZS-IMGC and ZS-KGC. The triple (`Zebra`, `imgc:hasAttribute`, `Stripe`) means that an animal class `Zebra` has an attribute `Stripe` in decoration, while the triple (`radioStation_in_city`, `rdfs:subPropertyOf`, `has_office_in_city`) means that the KG relation `radioStation_in_city` is a subrelation of `has_office_in_city`. It is worth mentioning that properties are also often defined with hierarchies, as the concepts. One general property is often defined for semantics of one aspect, and then more sub-properties are defined for more fine-grained semantics. Thus we can often easily find out relevant properties for different semantic aspects of an ontology by simple visualization of the property hierarchies.

In our ZS-KGC case study, we adopt ontologies developed in [9] as the auxiliary information for completing relational facts of their corresponding KGs in the zero-shot setting, where KG relations are modeled as ontology concepts and their meta-relationships are modeled by ontology properties. Our DOZSL framework contains a disentangled ontology encoder to learn disentangled representations for all concept nodes in an ontology, through which the fine-grained inter-concept relationships can be figured out and well utilized in downstream zero-shot learning and prediction steps.

### 2.3 Disentangled Representation Learning

The goal of disentangled representation learning is to learn embedding including various separate components behind the data. In the field of the graph, DisenGCN [22] is the first work tending to learn disentangled node representations, which uses a neighborhood routing mechanism to identify the latent factor that may have caused the link from a given node to one of its neighbors. However, it mainly focuses on homogeneous graphs with a single relation type. To process graphs with more diverse relation types, DisenE [18] and DisenKGAT [35], which leverage an attention mechanism and a dynamic assignment mechanism, respectively, disentangle the set of entities is considered following previous works, i.e., each entity that appears in the testing set has appeared during training.
As shown in Figure 2, DOZSL includes two core modules: Disentangled Ontology Encoder learning disentangled ontology embeddings, and Entangled ZSL Learner utilizing the embeddings for generation-based and propagation-based ZSL methods.

3 METHODOLOGY

As shown in Figure 2, DOZSL includes two core modules: Disentangled Ontology Encoder learning disentangled ontology embeddings, and Entangled ZSL Learner utilizing the embeddings for generation-based and propagation-based ZSL methods.

3.1 Disentangled Ontology Encoder

In DOZSL, the embedding of each concept node is disentangled into multiple distinct components as $c = [c^1, c^2, ..., c^K]$, where $K$ is the component numbers, $c^k \in \mathbb{R}^d$ represents the $k$-th component encoding semantics of one aspect of $c$ and $d$ is the embedding size.

To learn disentangled embedding for each concept, we first aggregate information from its graph neighborhoods that characterize it. In the aggregation of each component for a concept, only a subset of neighbors actually carries valuable information since each component represents a specific semantic aspect. To identify the aspect-specific subset, we follow the attention-based neighborhood routing strategy in previous works [22, 35]. Also, considering the various relation types in the ontologies, we propose a property-aware attention mechanism. Specifically, for the $k$-th aspect, the attention value of one neighbor $c_j$ of concept $c_i$ is computed by the similarity of the $k$-th component embeddings of $c_j$ and $c_i$ in the subspace of their connection property $p$ following the assumption that when a neighbor contributes more to $c_i$ in the aggregation, their property-aware representations are more similar, formally:

$$a^k_{(c_i, p, c_j)} = \text{softmax}(h^l_{i, k, p}^T h^j_{j, k, p})$$

$$h^l_{i, k, p} = h^l_{i, k} \circ W_p, \quad h^j_{j, k, p} = h^j_{j, k} \circ W_p$$

where $l \in \{0, 1, ..., L - 1\}$ with $L$ as the number of aggregation layers, $h^l_{i, k, p}$ is the $k$-th component embedding of $c_i$ w.r.t. property $p$ in the $l$-th aggregation layer, $\circ$ denotes the Hadamard product, and $W_p$ is a learnable projection matrix of $p$ for projecting $c_i$’s $k$-th component embedding $h^l_{i, k}$ into the property specific subspace. $\mathcal{N}(i) = \{(c_j, p')|(c_j, p', c'_j) \in \mathcal{T}_o\}$ is the set of pairs of neighboring concept nodes and properties of $c_i$, which also includes $c_i$ itself with a special self-connection property $p_s$. $\mathcal{T}_o$ is the ontology triple set. A dot-product similarity is adopted here.

With attention values, we separately aggregate the neighborhood information for representing each component and also update the property embedding after each aggregation as:

$$h^{l+1}_{i, k} = \sigma\left( \sum_{(c_j, p, c_j) \in \mathcal{T}_o} a^k_{(c_i, p, c_j)} \phi(h^l_{j, k, p}) \right), \quad h^0_{i, k} = h^l_{i, k}$$

where $h^l_{i, k}$ is the embedding of property $p$ in the $l$-th layer. $\phi$ is the layer-specific linear transformation matrix for $p$, $\psi$ is a combination operator for fusing the information of neighboring concept nodes and property edges. Here, we refer to CompGCN [30] to implement it via e.g. vector multiplication. $h^l_{i, k}$ is randomly initialized, and $h^L_{i, k}$ is outputted at last layer which has encoded the neighborhood information specific to aspect $k$. We make $c^k_i = h^L_{i, k}$ for simplicity.

To further improve the disentanglement, we propose to refine the semantics of each disentangled component embedding of concepts according to their associated properties. It is inspired by the characteristic of knowledge in ontologies, i.e., ontology properties are often represented with hierarchies, thus one general property can always be selected for representing one distinct semantic aspect of a concept; for example, the properties imgc:hasAttribute and rdfs:subClassOf in Figure 1 represent the semantics on animal visual characteristics and taxonomy, respectively.

To achieve this goal, we (i) select a set of properties for aspects of the semantics of the concepts to encode (e.g., imgc:hasAttributes for visual characteristics in the ontology for IMGC) and set the number of disentangled components to be the number of selected properties, and (ii) design a property guided triple scoring mechanism extracting property-specific components to constitute a valid ontology triple. Specifically, for an ontology triple $(c_i, p_k, c_j)$, we extract the $k$-th components of $c_i$ and $c_j$ with respective to property $p_k$, and leverage the score function on KG embedding methods to calculate the triple score with the extracted components. In this way, we accurately endow each component embedding with a specific semantic meaning w.r.t properties. Here, the score function of TransE [1] is adopted to compute the triple score as:

$$q(c_i, p_k, c_j) = f(-||c^k_i + p_k - c^k_j||)$$

where $c^k_i$ and $c^k_j$ denote the extracted component embeddings of concepts $c_i$ and $c_j$ respectively, and $p_k$ represents the embedding of property $p_k$. $f$ is the logistic sigmoid function. A higher score indicates a stronger relatedness between $c^k_i$, $p_k$, and $c^k_j$. Finally, we use the standard cross entropy with label smoothing to train the whole disentangled ontology encoder as:

$$\mathcal{L} = -\frac{1}{B} \frac{1}{|C|} \sum_{(c_i, p_k) \in \text{batch}} \sum_{n} \left( t_n \cdot \log(q(c_i, p_k, c_j)) + (1 - t_n) \cdot \log(1 - q(c_i, p_k, c_j)) \right)$$

where $B$ is the batch size, $C$ is the concept node set of the ontology, $t_n$ is the label of the given query $(c_i, p_k)$, whose value is 1 when the triple $(c_i, p_k, c_j)$ holds and 0 otherwise.

3.2 Entangled ZSL Learner

With the disentangled ontology embeddings, we next show how to utilize them for ZSL. Specifically, we develop two kinds of methods. In consideration of the effectiveness of GANs in learning the compatibility between class vectors and their samples, the first method is generation-based leveraging GANs to generate discriminative
samples for classes (each of which corresponds to an ontology concept). The other is propagation-based propagating features among classes based on the disentangled graphs generated from the original ontology.

### 3.2.1 Generation-based

We first get the embedding of each class by concatenating all K component embeddings of its corresponding ontology concept (i.e., $c_i = [e_i^1, e_i^2, \ldots, e_i^K]$), and then adopt a typical scheme of GAN for feature generation. Specifically, the GAN consists of three networks: a generator $G$ synthesizing sample features for a class from random noises conditioned on its embedding; a feature extractor $E$ providing the real sample features; and a discriminator $D$ distinguishing the generated features from the real ones. We generate sample features instead of raw samples for both higher accuracy and efficiency, as in many works [9, 25, 37].

Formally, for a class $c_i$, the generator $G$ takes as input its embedding and a random noise vector $z$ sampled from Normal distribution, and generates its features: $\hat{x} = G(z, c_i)$. The loss of $G$ is defined as:

$$L_G = -E[D(\hat{x})] + \lambda_1 L_{clu}(\hat{x}) + \lambda_2 L_R$$

where the first term is the Wasserstein loss, the second term is a supervised classification loss for classifying the synthesized features, and the third is for regularizing the mean of generated features of each class to be the mean of its real features. The latter two both encourage the generated features to have more inter-class discrimination. $\lambda_1$ and $\lambda_2$ are the corresponding weight coefficients.

The discriminator $D$ takes as input the synthesized features $\hat{x}$ from $G$ and the real features $x$ from $E$. Its loss is defined as:

$$L_D = E[D(x, c_i)] - E[D(\hat{x})] - \beta E[\|\nabla_{\hat{x}}D(\hat{x})\|_p - 1]^2$$

where the first two terms approximate the Wasserstein distance of the distributions of $x$ and $\hat{x}$. The last term is the gradient penalty to enforce the gradient of $D$ to have unit norm in which $\hat{x} = x + (1-\epsilon)\delta x$ with $\delta \sim U(0, 1)$. $\beta$ is the weight coefficient.

In view of the different data form in different ZSL tasks, we adopt different feature extractor $E$. For ZS-IMGC, we employ ResNet101 [13] to extract features of images following previous works [36]; and for ZS-KGC, we follow [9, 25] to learn cluster-structured features for KG relations. In general, $E$ is trained in advance with only samples of seen classes, and is fixed during adversarial training. Also, our framework is compatible to different feature extractors.

With well trained GAN, we use generator $G$ to synthesize features and train task-specific prediction models for unseen classes.

In ZS-IMGC, we train a softmax classifier for each unseen class to classify its testing images; in ZS-KGC, a testing triple is completed by calculating the similarity between the generated embedding of the relation $r$ and the joint embedding of the entity pair $(h, t)$.

### 3.2.2 Propagation-based

With disentangled concept embeddings, more fine-grained relatedness between concepts could be utilized. Therefore, as shown in Figure 2, we generate one semantic graph for each component, where nodes correspond to the classes (relations in KGC) in the dataset and edges are generated by calculating the cosine similarity between the component embeddings of two class nodes, and conduct graph propagation on it to transfer features between classes under each semantic aspect. The initialized node features are the class’s component embedding. Formally, we represent the $k$-th semantic graph as $G_k(A_k, S_k)$, where $S_k \in \mathbb{R}^{m \times K}$ is the input feature matrix of graph nodes, and $A_k \in \mathbb{R}^{m \times m}$ is the graph adjacency matrix indicating the connections among $m$ classes defined as below, $\tau$ denotes the similarity threshold.

$$A_k(i, j) = \begin{cases} 1 & \text{if } \text{sim}(e_i^k, e_j^k) > \tau \\ 0 & \text{otherwise} \end{cases}$$

Since $G_k$ is a graph with one single relation, we use GCN for feature propagation. Each graph convolutional layer performs as:

$$H_k^{l+1} = \sigma(\tilde{A}_k H_k^l \Phi_k^l)$$

where $\tilde{A}_k$ is the normalized adjacent matrix, and $\Phi_k^l$ is a layer-specific weight matrix shared among all semantic graphs. $H_k^0 = S_k$.

For each semantic graph, the GCN outputs a set of node embeddings $Z_k \in \mathbb{R}^{m \times F}$, through which we can obtain a set of classifiers $\tilde{W} = \{W_k \mid Z_1, Z_2, \ldots, Z_K\}$, where $\varphi$ is a fusion function. In our experiments, we implement $\varphi$ by averaging: $\tilde{W} = \frac{1}{K} \sum Z_k$, or linear transformation: $\tilde{W} = W_l(\{Z_1, Z_2, \ldots, Z_K\})$ where $W_l \in \mathbb{R}^{F \times m}$ is a trainable transformation matrix. Then, following [16, 32, 34], we compute the Mean Square Error between the fused classifiers and the ground-truth classifiers as loss function:

$$L_{GCN} = \frac{1}{|W_l|} \sum_{w \in \tilde{W}} (\tilde{w} - gt(\tilde{w}))^2$$

where $\tilde{W} \subset \tilde{W}$ is the set of classifiers of the seen classes, $gt(\tilde{w})$ denotes the corresponding ground-truth. Different from the traditional classifier which is a network trained using labeled samples,
the classifier here is actually a real-valued vector that represents the class-specific features, and is obtained by averaging the features of all the training samples of one class in our paper. The sample features are also extracted via the feature extractor $E$ mentioned in Section 3.2.1. By using these ground-truth seen classifiers to supervise the training of GCNs, classifiers of the unseen classes can be learned by aggregation. During prediction, for an input testing sample, we first extract its features using the same feature extractors, and then perform classification or completion by calculating the similarity between the learned classifiers and the extracted features.

4 EVALUATION

4.1 Experiment Settings

4.1.1 Datasets and Ontologies. For ZS-IMGC, we use a popular benchmark named Animals with Attributes (AwA) [36] and two benchmarks ImNet-A and ImNet-O extracted from ImageNet by Geng et al. [9]. AwA is for coarse-grained animal image classification with 50 classes and 37,322 images. ImNet-A is for more fine-grained animal image classification and ImNet-O is for fine-grained general object classification. The classes are split into a seen set and an unseen set, following [36]. For ZS-KGC, we use two KGs provided in [25] for completion, i.e., NELL-ZS and Wiki-ZS extracted from NELL and Wikidata, respectively. In each KG, the relations are split into a training set with seen relations, a validation set and a testing set with unseen relations, following [25]. Accordingly, their associated triples compose a training set, a validation set and a testing set. It is ensured that all entities are seen.

Each dataset has an ontology as its auxiliary information. We use the ontologies developed in [9] and take the latest version released in [10]. For ZS-IMGC, the ontologies contain class hierarchies (taxonomy), class visual attributes and attribute hierarchies. In our property guided disentangled embedding, we select two general properties: `rdfs:subClassOf` for semantic aspect on taxonomy, and `imgc:hasAttribute` for semantic aspect on visual characteristics. For ZS-KGC, the ontologies contain type constraints of the head and tail entities of relations, represented by properties `rdfs:domain` and `rdfs:range`, relation hierarchies represented by property `rdfs:subProperty`, and type hierarchies represented by property `rdfs:subClassOf`. These four properties are selected as general properties used in ontology encoder. See Table 1 for detailed statistics.

4.1.2 Variants of DOZSL and Baselines. In disentangled ontology encoder, we compare two settings for component embeddings that are fed to score triple (Eq. (4)): aggregating neighborhood information (Eq. (1) and (3)), and randomly initializing component embeddings without neighborhood aggregation. This leads to two DOZSL variants. Meanwhile, they can be combined with two downstream ZSL methods: generation-based with GAN and propagation-based with GCN. Thus we have four DOZSL variants and denote them as “DOZSL(X+Y)”, where X can be AGG (neighborhood aggregation) and RD (random initialization), Y can be GAN and GCN.

The baselines include those generation-based and propagation-based ZSL methods that often achieve state-of-the-art performance on many ZSL datasets. OntoZSL [9] is a generation-based method that uses GANs to synthesize samples, where we take TransE as its ontology encoder for a fair comparison. DGP [16] is a propagation-based method using a two-layers GCN which only supports single-relation graphs. To deal with the multi-relation ontology graph, we take the method proposed in [32] as a baseline. Meanwhile, two relation-aware GNNs, RGCN [27] and CompGCN [30], are also used to implement another two propagation-based ZSL baselines. We also consider different disentangled and non-disentangled semantic embedding methods for more baselines. For non-disentangled embedding, we choose classical TransE and RGAT which also performs attentive relation-aware graph aggregation. For disentangled embedding, we choose two state-of-the-art methods DisenE [18] and DisenKGAT [35]. These embedding methods can also be combined with GAN-based and GCN-based ZSL learners as in DOZSL, leading to baselines such as "DisenKGAT+GAN". Note "TransE+GAN" is equivalent to OntoZSL.

4.1.3 Evaluation Metrics. For ZS-IMGC, we report macro accuracy following [36], where accuracy of each class is first calculated with its testing images, and the accuracies of all testing classes are then averaged. For standard ZSL testing, we compute accuracy on all unseen classes, denoted as $acc$; while for generalized ZSL testing, we first calculate accuracy for all the seen classes and all the unseen classes separately, denoted as $acc_s$ and $acc_u$, respectively, and then report a harmonic mean $H = (2 \times acc_s \times acc_u) / (acc_s + acc_u)$.

Our ZS-KGC task is to predict the tail entity $t$ given a head entity $h$ and an unseen relation $r_u$. Thus for the input of a testing triple $(h, r_u, t)$, we rank a set of candidate entities according to their predicted scores of being the tail entity, and see the rank of the ground truth tail entity — the smaller rank, the better performance.

As in most KGC works, we report Mean Reciprocal Ranking (MRR) and hit@$k$ (i.e., the ratio of testing samples whose ground truths are ranked in the top-$k$ position). $k$ is set to 1, 5, 10. Different from ZS-IMGC where predicting the class label of an image tends to be confused by other classes, the prediction for a seen relation in ZS-KGC is relatively independent of the prediction for an unseen relation. Thus the generalized ZSL testing setting in ZS-KGC, which is a simple addition of normal KGC, is not considered in our paper.

4.2 Main Results

4.2.1 ZS-IMGC. The results are reported based on these settings. For ontology encoder, we set the component embedding size and the property embedding size to 100. $K$ is set to 2 (corresponding to `rdfs:subClassOf` and `imgc:hasAttribute`) for all DOZSL(RD) variants,
Table 2: Accuracy and $H$ (%) of ZS-IMGC on AwA, ImNet-A and ImNet-O. MRR and hit@k (%) of ZS-KGC on NELL-ZS and Wiki-ZS. The best results in a method category (resp. in the whole column) are in bold (resp. underlined). TransE+GAN equals OntoZSL.

| Category | Methods          | AwA acc | ImNet-A acc | ImNet-O acc | NELL-ZS hit@10 | Wiki-ZS hit@10 |
|----------|------------------|--------|-------------|-------------|----------------|----------------|
| Generation | TransE+GAN       | 58.28  | 54.77      | 39.44 | 31.61 | 31.93 | 27.82 |
|           | RGAT+GAN         | 63.95  | 57.52      | 39.20 | 31.17 | 35.13 | 27.68 |
|           | DisenE+GAN       | 59.40  | 49.47      | 33.60 | 29.96 | 31.62 | 26.69 |
|           | DisenKGAT+GAN    | 61.81  | 54.41      | 35.90 | 31.09 | 34.94 | 27.33 |
|           | DOZSL(RD+GAN)    | 52.35  | 46.91      | 37.12 | 30.18 | 34.48 | 28.57 |
|           | DOZSL(AGG+GAN)   | 66.36  | 57.62      | 40.26 | 32.82 | 36.00 | 28.74 |
| Propagation | DGP              | 59.03  | 28.97      | 35.72 | 29.98 | 34.89 | 29.76 |
|           | Wang et al. [32] | 43.81  | 42.13      | 34.33 | 21.95 | 32.73 | 26.86 |
|           | RGCN-ZSL         | 44.90  | 24.95      | 37.36 | 33.01 | 31.19 | 23.39 |
|           | CompGCN-ZSL      | 53.46  | 29.33      | 38.34 | 29.01 | 28.95 | 27.35 |
|           | TransE+GCN       | 63.36  | 36.15      | 36.69 | 22.12 | 33.16 | 24.72 |
|           | RGAT+GCN         | 58.83  | 37.35      | 37.53 | 31.27 | 35.47 | 28.49 |
|           | DisenE+GCN       | 58.34  | 50.86      | 32.56 | 27.76 | 32.02 | 26.33 |
|           | DisenKGAT+GCN    | 61.24  | 37.43      | 37.55 | 32.27 | 35.92 | 29.50 |
|           | DOZSL(RD+GCN)    | 62.79  | 52.74      | 36.01 | 30.29 | 33.66 | 31.19 |
|           | DOZSL(AGG+GCN)   | 63.88  | 44.52      | 38.69 | 32.12 | 37.42 | 31.77 |

but to 5 for all DOZSL(AGG) variants since two reverse properties and a self-connection property are added during aggregation. The initial learning rate is set to 0.001. The number of the aggregation layer for DOZSL(AGG) varies is set to 1.

For ZSL learner, we employ ResNet101 to extract 2,048-dimensional image features. It is ensured that unseen classes of all the three datasets have never appeared in training ResNet101. Regarding GAN, the generator and discriminator both consist of two fully connected layers with 4,096 hidden units; their learning rates are set to 0.0001; the dimension of noise vector $z$ is set to 100; $\lambda_1$, $\lambda_2$ and $\beta$ are set to 0.01, 5 and 10, respectively. Regarding GCN, the classifier vector $\boldsymbol{v}$ is set to 2,048; 2 convolutional layers with a hidden dimension of 2,048 are used; the learning rate is set to 0.001. As for the optimum similarity threshold for creating semantic graphs, we provide a detailed evaluation in Section 4.3.

For baselines DisenE and DisenKGAT, we test different $K$ values and report the better ones in the main body, and attach the complete results in Appendix A. More details please see our released codes.

**Overall Results.** The results are shown in the left side of Table 2. We can see DOZSL always achieves the best performance on AwA and ImNet-O, no matter what downstream ZSL learners are applied (+GAN or +GCN). On ImNet-A, DOZSL is still the best in most cases. Although DOZSL does not outperform RGCN-ZSL on the metric of $H$, the result is still comparable.

**Results on ZSL Learners.** Using either GAN or GCN can make our framework perform better than the baselines. Especially, when the input ontology embedding is fixed, we can often select one of them for better performance. For example, on AwA, using GAN with DOZSL(RD+GAN) achieves even higher performance than DisenE+GAN and DisenE+GCN. Moreover, our DOZSL variants with GCN perform better than previous propagation-based ZSL methods in most situations, illustrating that our method can more effectively capture the structural class relationships in ontologies.

4.2.2 ZS-KGC. For ontology encoder, we re-use the settings in ZS-IMGC. The dimension of component embedding and property embedding is set to 200. $K$ is 4 for DOZSL(RD) and is 9 for DOZSL(AGG) considering the reverse properties and the self-connection property. The feature extractors are pre-trained to extract 200-dimensional and 100-dimensional relation features for NELL-ZS and Wiki-ZS, respectively, following the settings in [9, 25], with TransE-based embeddings as the input. For ZSL learner, we also employ the same GAN and GCN architectures as in ZS-IMGC, but use some different settings. Regarding the GAN for NELL-ZS, the generator has 250 hidden units, while the discriminator has 200 hidden units. Regarding the GAN for Wiki-ZS, the corresponding unit numbers are 200 and 100. For both datasets, the noise vector size is set to 15; $\lambda_1$, $\lambda_2$ are set to 1 and 3, respectively. Regarding GCN, the classifier vector size is 200 for NELL-ZS and 100 for Wiki-ZS. As in ZS-IMGC, the selection of similarity thresholds for creating semantic graphs is evaluated in Section 4.3; different $K$ values are tested for DisenE and DisenKGAT with the optimum performance reported in Table 2 and the complete results attached in Appendix A.

**Overall Results.** The results are presented in the right of Table 2. On NELL-ZS, our method achieves the best on hit@10 and hit@5, DOZSL(RD+GAN) and DOZSL(RD+GCN) are both very competitive to the baseline RGCN-ZSL and better than other baselines on hit@1 and MRR. On Wiki-ZS, two baselines RGAT+GAN and RGCN-ZSL
We conduct extensive ablation studies to analyze the impact of the only exception of DOZSL(RD) and DOZSL(AGG) is narrowed, and DOZSL(RD) even performs better on some metrics, which can be attributed to the following reasons. (1) The neighborhood information of concepts in ZS-IMGc task is richer than that in ZS-KGC task, especially for the concepts in NELL-ZS’s ontology. Statistically, the average number of surrounding neighbors for NELL-ZS is around 3.4, while the number for ImNet-A is around 5.9. (2) The properties in the ontologies of ZS-IMGc task such as imgc:hasAttribute are 1-N; while most properties in the ontologies of ZS-KGC task are 1-1. The embedding methods that ignore aggregating the neighborhood are often not good at handling these 1-N properties. Besides, in comparison with the disentangled and non-disentangled baselines, our methods always have superior performances, i.e., on most metrics, DOZSL(AGG) performs better than RGAT and DisenKGAT, and DOZSL(RD) outperforms DisenE and TransE by a large margin.

**Results on ZSL Learners.** Given the same ontology embeddings, we find the performance varies from one ZSL learner to another. The GCN-based learner usually performs better than the GAN-based one on NELL-ZS, while the GAN-based learner reversely performs better on Wiki-ZS. This motivates us to conduct an in-depth analysis about the interaction between the datasets and the ZSL methods, so that making a more suitable selection for better performance. Moreover, RGCN-ZSL also shows the promising ability of relation-aware GNNs on ZS-KGC task.

### 4.3 Ablation Studies

We conduct extensive ablation studies to analyze the impact of different factors in DOZSL, including the property guided triple scoring, the neighborhood aggregation, the similarity threshold for constructing semantic graphs and the classifier fusion.

**Property Guided Triple Scoring.** We replace the property guided triple scoring in DOZSL(RD) and DOZSL(AGG) by the widely-adopted attentive triple scoring and keep the same setting of $K$. This leads to two new variants, denoted as DOZSL(RD\(_{atten}\)) and DOZSL(AGG\(_{atten}\)), respectively. These variants’ results with GAN are reported in Table 3, the results with GCN are attached in Appendix B. We can find that DOZSL(RD\(_{atten}\)) and DOZSL(AGG\(_{atten}\)) always obtain dramatically worse results than DOZSL(RD) and DOZSL(AGG), respectively, on all the datasets of the two tasks, with the only exception of DOZSL(RD\(_{atten}\) + GAN) on AwA. These results illustrate the efficiency of our proposed property guided triple scoring. The except may be due to the imbalanced associated triples of different properties in AwA’s ontology: imgc:hasAttribute has 1,562 associated triples, which can well train its corresponding component, while rdfs:subClassOf has only 197 associated triples, making its corresponding component under fitted. The two components are concatenated and fed to GANs together, thus they may influence each other. In contrast, the GCN-based method, which performs independent feature propagation in isolated semantic graphs, suffers less from the imbalance issue.

**Neighborhood Aggregation.** In DOZSL, we aggregate information from all the neighboring concepts in the ontology, with an attention mechanism for combination. Here, we want to test a more straightforward solution, i.e., aggregating information from a neighborhood subset which only includes concepts that are connected by the property corresponding to the embedding component. This leads to new variants denoted by DOZSL(AGG\(_{sub}\)). The results with GAN are shown in Table 3, the results with GCN are in Appendix B. In comparison with DOZSL (AGG), DOZSL(AGG\(_{sub}\)) performs worse on most metrics across two tasks, except for DOZSL(AGG\(_{sub}\) + GAN) on ImNet-O w.r.t. $H$ and DOZSL(AGG\(_{sub}\)*GCN) on NELL-ZS. The overall worse results of DOZSL(AGG\(_{sub}\)) indicate that learning a component embedding should (attenitively) aggregate all the neighboring concepts rather than select a part of them according to the specific properties. The exceptions may be due to the simple neighborhoods in NELL-ZS and ImNet-O and/or the independent propagation in each semantic graph.

**Similarity Threshold and Classifier Fusion.** We compare different similarity thresholds ranging from 0.85 to 0.999 for constructing semantic graphs, and compare different classifier fusion functions, under different ontology encoding methods. The results are reported in Figure 4 in Appendix C, from which we can find that the optimum similarity threshold varies when different ontology encoding methods are used, and the two fusion functions — Average and Linear Transformation both positively contribute to the learning of the classifier. Please see Appendix C for more details.

### 4.4 Case Study

We use examples from NELL-ZS to analyze disentanglement of concept embeddings we learned. In the left of Figure 3, we visualize the component embeddings of KG relations learned from NELL-ZS’s ontology by DOZSL(RD), where different colors indicate different components. We can find that (i) the embeddings are clustered into different groups under each component’s subspace, and (ii) the component embeddings of each relation are divided into different clusters across different components. These observations illustrate that (i) our method indeed captures the semantically similarity among relation concepts under each semantic aspect and (ii) different relatedness is presented across different aspects.

Also, to further verify that different components represent different semantic aspects, for each relation, we randomly select two neighbors from the cluster of each component. The right of Figure 3...
With the new disentangled embeddings, different semantic aspects of the relations, i.e., \( \text{rdfs:domain, rdfs:range, rdfs:subPropertyOf} \) and \( \text{rdfs:subClassOf} \), and we can also conclude that the semantic of one component is a fixed across different relations.

5 CONCLUSION AND DISCUSSION

In this study, we focused on ontology augmented ZSL and proposed a novel property guided disentangled ontology embedding method. With the new disentangled embeddings, different semantic aspects of ZSL classes are figured out and more fine-grained inter-class relationships are extracted, through which the ontology can be better utilized. To integrate these disentangled embeddings, we also developed a general ZSL framework DOZSL, including a GAN-based generative model and a GCN-based propagation model. Extensive evaluations with ablation studies and case studies on five datasets of ZS-IMGC and ZS-KGC show that DOZSL often outperforms the state-of-the-art baselines and its components are quite effective. DOZSL is compatible to both ZSL learners developed by us, and they together lead to higher robustness and better performance. Meanwhile, the performance of DOZSL is less competitive to the state-of-the-art on one of the five datasets. This motivates us to take an in-depth analysis of this dataset and its ontology, and to develop more robust disentangled embedding methods and ZSL learners in the future. We also realize some relation-aware GNNs such as RGCN achieve quite promising results on some datasets. This motivates the future. We also realize some relation-aware GNNs such as RGCN achieve quite promising results on some datasets. This motivates the future.

5 CONCLUSION AND DISCUSSION

In this study, we focused on ontology augmented ZSL and proposed a novel property guided disentangled ontology embedding method. With the new disentangled embeddings, different semantic aspects of ZSL classes are figured out and more fine-grained inter-class relationships are extracted, through which the ontology can be better utilized. To integrate these disentangled embeddings, we also developed a general ZSL framework DOZSL, including a GAN-based generative model and a GCN-based propagation model. Extensive evaluations with ablation studies and case studies on five datasets of ZS-IMGC and ZS-KGC show that DOZSL often outperforms the state-of-the-art baselines and its components are quite effective.

DOZSL is compatible to both ZSL learners developed by us, and they together lead to higher robustness and better performance. Meanwhile, the performance of DOZSL is less competitive to the state-of-the-art on one of the five datasets. This motivates us to take an in-depth analysis of this dataset and its ontology, and to develop more robust disentangled embedding methods and ZSL learners in the future. We also realize some relation-aware GNNs such as RGCN achieve quite promising results on some datasets. This motivates us to study the propagation-based ZSL learner with these GNNs. Lastly, we will apply and evaluate DOZSL in other tasks such as open information extraction and visual question answering.

ACKNOWLEDGMENTS

This work is partially funded by NSFCU19B2027/91846204, the EP-SRC project ConCur (EP/V050869/1) and the Chang Jiang Scholars Program (J2019032).

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, et al. 2013. Translating Embeddings for Modeling Multi-relational Data. In NIPS. 2787–2795.
[2] Jiaoyan Chen, Yuxia Geng, Zhuo Chen, Ian Horrocks, Jeff Z Pan, et al. 2021. Knowledge-aware zero-shot learning: Survey and perspective. In IJCAI.
[3] Jiaoyan Chen, Freddy Lécué, Yuxia Geng, Jeff Z Pan, and Huajun Chen. 2020. Ontology-guided Semantic Composition for Zero-shot Learning. In KR 850–854.
[4] Jingjing Chen, Liangming Pan, et al. 2020. Zero-Shot Ingredient Recognition by Multi-Relational Graph Convolutional Network. In AAAI 10542–10550.
[5] Mingyang Chen, Wen Zhang, Yuxuan Zhu, Huajun Chen, et al. 2022. Meta-Knowledge Transfer for Inductive Knowledge Graph Embedding. In SIGIR.
[6] Zhuo Chen, Jiaoyan Chen, Yuxia Geng, Jeff Z Pan, Huajun Chen, et al. 2021. Zero-shot visual question answering using knowledge graph. In ISWC. 146–162.
[7] Andrada Trome, Gin S. Corrado, Jonathon Shenks, Samy Bengio, et al. 2013. DeViSE: A Deep Visual-Semantic Embedding Model. In NIPS 2121–2129.
[8] Zhenyong Fu, Tao Xiang, Elyor Kodirov, and Shaoang Gong. 2015. Zero-shot object recognition by semantic manifold distance. In CVPR. 2635–2644.
[9] Yuxia Geng, Jiaoyan Chen, Zhou Chen, Jeff Z Pan, Zhequan Ye, Huajun Chen, et al. 2021. OntoZSL: Ontology-enhanced Zero-shot Learning. In WWW 3325–3336.
[10] Yuxia Geng, Jiaoyan Chen, Zhou Chen, Jeff Z Pan, Zonggang Yuan, and Huajun Chen. 2021. Benchmarking Knowledge-driven Zero-shot Learning. CoRR (2021).
[11] Yuxia Geng, Jiaoyan Chen, Zhiqian Ye, Wei Zhang, and Huajun Chen. 2021. Explainable Zero-shot Learning via Attentive Graph Convolutional Network and Knowledge Graphs. Semantic Web (2021).
[12] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, Yoshua Bengio, et al. 2014. Generative Adversarial Nets. In NIPS 2672–2680.
[13] Kaming He, Xiayang Zhu, Shaoqiang Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In CVPR. 770–778.
[14] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d’Amato, Gerard de Melo, et al. 2021. Knowledge Graphs. (2021).
[15] Ian Horrocks. 2008. Ontologies and the semantic web. Commun. ACM 51, 12 (2008), 58–67.
[16] Michael Kampffmeyer, Yinbo Chen, Xiaoan Liang, et al. 2019. Rethinking Knowledge Graph Propagation for Zero-shot Learning. In CVPR. 11487–11496.
[17] Thomas N. Kif and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In ICLR (Poster).
[18] Xiaoyu Kou, Yankai Lin, Yuntao Li, Jiahao Xu, Peng Li, Jie Zhou, and Yan Zhang. 2020. DisEnt: Disentangling Knowledge Graph Embeddings. (2020).
[19] Christoph H. Lampert, Hannes Nickisch, and Stefan Harmeling. 2014. Attribute-Based Classification for Zero-Shot Visual Object Categorization. IEEE Trans. Pattern Anal. Mach. Intell. 36, 3 (2014), 453–465.
[20] Juan Li, Ruoxu Wang, et al. 2020. Logic-guided Semantic Representation Learning for Zero-shot Relation Classification. In COLING. 2967–2978.
[21] Xiayou Li, Zhe Xu, Kun Wei, and Cheng Deng. 2021. Generalized Zero-shot Learning via Disentangled Representation. In AAAI. 1966–1974.
[22] Jiaojian Mu, Peng Cui, Kun Kuang, Xin Wang, and Wenwu Zhu. 2019. Disentangled Graph Convolutional Networks. In ICML. 4212–4221.
[23] George A Miller. 1995. WordNet: a lexical database for English. Commun. ACM 38, 11 (1995), 39–41.
[24] J.Z. Pan, G. Vetro, J.M. Gomez-Perez, and H. Wu. 2016. Exploiting linked data and knowledge graphs for large organisations. Springer.
[25] Pengda Qin, Xin Wang, et al. 2020. Generative Adversarial Zero-shot Relational Learning for Knowledge Graphs. In AAAI 6873–6880.
[26] Abhinaba Roy, Deepanway Ghosal, et al. 2020. Improving Zero-shot Learning Baselines with Commonsense Knowledge. CoRR abs/2012.06263 (2020).
[27] Michael Sejr Schlichtkrull, Thomas N. Kif, Peter Bloem, et al. 2018. Modeling Relational Data with Graph Convolutional Networks. In ESWC. 593–607.
[28] Runhui Speer, Joshua Chiu, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multi-lingual Graph of General Knowledge. In AAAI 4444–4451.
[29] Komał K. Teru, Etienne Denis, and Will Hamilton. 2020. Inductive Relation Prediction by Subgraph Reasoning. 119 (2020), 9448–9457.
[30] Shikhar Vashisth, Somnath Sanyal, Vikram Niran, and Farzana P. Talukdar. 2020. Composition-based Multi-Relational Graph Convolutional Networks. In ICLR.
[31] Bo Wang, Tao Shen, et al. 2021. Structure-Augmented Text Representation Learning for Efficient Knowledge Graph Completion. In WWW 1757–1764.
[32] Jin Wang and Bo Jiang. 2021. Zero-Shot Learning via Contrastive Learning on Dual Knowledge Graphs. In IJCAI 885–892.
[33] Peifeng Wang, Jialong Han, et al. 2019. Logic Attention Based Neighborhood Aggregation for Inductive Knowledge Graph Embedding. In AAAI 7152–7159.
[34] Xiaolong Wang, Yufei Ye, and Abhinav Gupta. 2018. Zero-Shot Recognition via Contrastive Learning on Dual Knowledge Graphs. In IJCV 887–892.
[35] Jingjing Chen, Liangming Pan, et al. 2020. Zero-Shot Ingredient Recognition by Multi-Relational Graph Convolutional Network. In AAAI 10542–10550.
[36] Yuping Xie, Shengyuan Cai, and Zhaoyang Gao. 2021. Learning a Deep Embedding Model for Zero-shot Learning. In CVPR 3010–3019.
[37] YiZhe Zhu, Mohamed Elhoseiny, Bingchen Liu, et al. 2018. A Generative Adversarial Approach for Zero-Shot Learning From Noisy Texts. In CVPR. 1004–1013.
Table 4: Results (%) of DisenE and DisenKGAT with GAN w.r.t different K values. The better results are in bold.

| Methods | K   | AWA | ImNet-A | ImNet-O | NELL-ZS | Wiki-ZS |
|---------|-----|-----|---------|---------|---------|---------|
|         |     | acc | H       | acc     | H       | hit     | hit@5   | hit@1   | MRR     | hit@10  | hit@5   | hit@1   | MRR     |
| DisenE+GAN | 2   | 59.40 | 49.47  | 33.60   | 29.96  | 31.62  | 26.69  | 34.5     | 25.2     | 15.4     | 22.0     | 28.0     | 22.7     | 13.8     | 18.9     |
| DisenE+GAN | 4   | 44.59 | 41.69  | 24.27   | 23.79  | 21.62  | 21.80  | 34.8     | 29.1     | 15.3     | 22.2     | 27.2     | 22.0     | 14.2     | 18.8     |
| DisenKGAT+GAN | 2  | 60.20 | 54.08  | 35.90   | 31.09  | 34.94  | 27.33  | 35.9     | 29.5     | 15.7     | 22.9     | 27.5     | 22.0     | 13.7     | 18.6     |
| DisenKGAT+GAN | 4  | 61.81 | 54.41  | 31.35   | 28.57  | 31.58  | 27.13  | 35.0     | 28.7     | 16.1     | 22.5     | 27.6     | 22.4     | 13.8     | 18.8     |

Table 5: Results (%) of ablation studies with GCN. The better results in each group are in bold.

| Methods               | AWA | ImNet-A | ImNet-O | NELL-ZS | Wiki-ZS |
|-----------------------|-----|---------|---------|---------|---------|
|                       | acc | H       | acc     | H       | hit     | hit@5   | hit@1   | MRR     | hit@10  | hit@5   | hit@1   | MRR     |
| DOZSL(RD+GCN)         |     | 62.79   | 52.74  | 36.01   | 30.29  | 33.66  | 31.19  | 38.0     | 31.2     | 16.5     | 23.9     | 26.7     | 21.9     | 13.8     | 18.5     |
| DOZSL(RD_atten+GCN)   |     | 58.34   | 50.86  | 32.56   | 27.76  | 32.02  | 26.33  | 35.5     | 29.7     | 15.6     | 22.7     | 26.3     | 21.4     | 13.4     | 18.1     |
| DOZSL(AGG)+PRO        |     | 63.88   | 44.52  | 38.69   | 32.12  | 37.42  | 31.77  | 36.2     | 29.3     | 16.2     | 23.0     | 27.5     | 22.4     | 13.6     | 18.7     |
| DOZSL(AGG_sub+GCN)    |     | 54.00   | 32.00  | 36.18   | 27.55  | 31.47  | 26.54  | 35.3     | 28.8     | 15.6     | 22.3     | 27.5     | 22.1     | 13.7     | 18.7     |
| DOZSL(AGG)+Linear     |     | 63.66   | 33.19  | 35.03   | 26.63  | 35.37  | 31.11  | 36.9     | 30.0     | 15.8     | 23.1     | 27.1     | 21.8     | 13.4     | 18.3     |

Figure 4: Results of GCN-based DOZSL variants using different ontology encoders with different similarity thresholds and different classifier fusion functions. Best viewed in color.

A SENSITIVITY STUDY OF DISENE AND DISENKGAT

In this section, we study the sensitivity of the number of components $K$ used in the baselines DisenE [18] and DisenKGAT [35]. Specifically, we $K$ to 2 and 4, two values with which the baselines perform well, and experiment with the GAN-based learner. The results on the six datasets of the two ZSL tasks are presented in Table 4. We can find that DisenE gets higher performance on all
the three ZS-IMGC datasets and on Wiki-ZS when \( K = 2 \). It also gets better results on most metrics on NELL-ZS when \( K = 4 \). As for DisenKGAT, the optimum \( K \) values on AwA, ImNet-A, ImNet-O, NELL-ZS and Wiki-ZS are 4, 2, 2, 2, 4, respectively.

**B ABLATION STUDY OF THE ONTOLOGY ENCODER WITH GCN-BASED METHODS**

In this section, we report the results of ablation studies on the property guided triple scoring and the neighborhood aggregation in the disentangled ontology encoder when incorporating with GCN-based methods. The results are shown in Table 5.

**C ABLATION STUDY OF THE GCN-BASED LEARNER**

In this section, we study the impact of the similarity threshold and the classifier fusion function under different disentangled ontology embeddings, using all our evaluation datasets. The results are presented in Figure 4. Specifically, we report the results of the metric of \( acc \) (i.e., the standard ZSL testing setting) for ZS-IMGC task and the results of the metrics of \( hit@10 \) and \( MRR \) for ZS-KGC task. Moreover, the curve of the Average fusion function is decorated with circular, while the curve of the Linear Transformation fusion function is decorated with triangle. Different ontology encoding methods are presented in different colors.