K-ZSL: Resources for Knowledge-driven Zero-shot Learning

Yuxia Geng
gengyx@zju.edu.cn
Zhejiang University
Hangzhou, China

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

Zhuo Chen
zhuo.chen@zju.edu.cn
Zhejiang University
Hangzhou, China

Jeff Z. Pan
https://knowledge-representation.org/j.z.pan/
University of Edinburgh
Edinburgh, United Kingdom

Zonggang Yuan
yuanzonggang@huawei.com
NAIE CTO Office,
Huawei Technologies Co., Ltd.
Nanjing, China

Huajun Chen
huajunsir@zju.edu.cn
College of Computer Science & HIC,
Zhejiang University
AZFT Knowledge Engine Lab

ABSTRACT

External knowledge (a.k.a side information) plays a critical role in zero-shot learning (ZSL) which aims to predict with unseen classes that have never appeared in training data. Several kinds of external knowledge such as text and attribute have been widely investigated, but they alone are limited with incomplete semantics. Therefore, some very recent studies propose to use Knowledge Graph (KG) due to its high expressivity and compatibility for representing kinds of knowledge. However, the ZSL community is still in short of standard benchmarks for studying and comparing different KG-based ZSL methods. In this paper, we proposed 5 resources for KG-based research in zero-shot image classification (ZS-IMGC) and zero-shot KG completion (ZS-KGC). For each resource, we contributed a benchmark and its KG with semantics ranging from text to attributes, from relational knowledge to logical expressions. We have clearly presented how the resources are constructed, their statistics and formats, and how they can be utilized with cases in evaluating ZSL methods' performance and explanations. Our resources are available at https://github.com/China-UK-ZSL/Resources_for_KZSL.

KEYWORDS

Zero-shot Learning, Knowledge Graph, Image Classification, Knowledge Graph Completion, Ontological Schema, Semantic Embedding

ACM Reference Format:
Yuxia Geng, Jiaoyan Chen, Zhuo Chen, Jeff Z. Pan, Zonggang Yuan, and Huajun Chen. 2018. K-ZSL: Resources for Knowledge-driven Zero-shot Learning. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Supervised learning has achieved impressive performance in many domains such as natural language processing and computer vision. Its methods often require a large number of labeled training samples to achieve good performance, following a closed world assumption. Namely they predict with classes that have appeared in the training stage (i.e., seen classes). However, in real-world applications, new classes always emerge, and it often costs too much computation, human labour and time to address these new classes, by collecting labeled samples and training the model from the scratch. To this end, Zero-shot Learning (ZSL), which aims at predicting with classes that have no training samples (i.e., unseen classes), was proposed and has been widely investigated in the past decade [3, 30, 40, 43].

Existing ZSL methods usually learn a knowledge transfer model in the following paradigm. They collect and/or construct *external knowledge* (a.k.a *side information*) which describes prior semantic relationships between classes, embed all the classes with the external knowledge, establish a mapping of inter-class relationship from the class embedding space to the sample space, and transfer model parameters (e.g., features) learned from samples of seen classes to a new model that can predict for unseen classes. Widely investigated external knowledge includes class textual information (e.g., names and descriptions) [10, 31] and class attributes (e.g., annotations) [22]. However, each kind of such external knowledge fails to accurately or fully express inter-class relationships.

Recently, KG has attracted wide attention as external knowledge in ZSL, where KG structural knowledge is exploited. For example, [20, 41] incorporate hierarchical inter-class relationships from WordNet [24]; [12, 26, 34, 44] exploit relational class knowledge from common sense KGs such as ConceptNet [37]. ZSL performance is often significantly improved when these KGs are well utilized. Besides, KG can also be used to represent many other kinds of traditional side information such as human annotation and text description [13, 23], due to its high compatibility in representing and integrating different knowledge. However, although various KGs have been exploited by the current methods, there is still a concern on semantics completeness especially in distinguishing fine-grained classes. Very few methods have been developed that can jointly utilize multiple kinds of knowledge in a KG, while other kinds of KG semantics such as logical expressions have not been investigated yet. More importantly, existing works all build their own KGs for evaluation, and the community lacks standard and unified benchmarks for comparing different KG-based ZSL methods under settings with ranging semantics.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/1122445.1122456

In this work, we construct systemic resources for KG-based ZSL research. The resources include 5 benchmarks and their KGs for two different but typical tasks: zero-shot image classification (ZS-IMGC) and zero-shot knowledge graph completion (ZS-KGC). The KGs contain different kinds of external knowledge, including not only typical side information such as attribute, text and hierarchy, but also common sense relational facts and logical expressions, with the goal of providing ranging semantics settings for investigating different KG-based ZSL methods. In addition to resource construction details, and resource statistics and accessibility information, we also demonstrated the high usability of these resources in evaluating the performance and explanation of two state-of-the-art KG-based ZSL methods and one classic ZSL method, and proposed a baseline method to utilize logical expressions in ZS-KGC.

Note an old version of these resources have been used in evaluating our method OntoZSL [13]. This paper focuses on the resources, introducing their construction, statistics and use cases, and making them more accessible. The resources have also been substantially extended with logical expressions, new common sense knowledge, additional use cases, more formal KG representation and so on.

2 BACKGROUND AND RELATED WORK

2.1 Knowledge Graph

Knowledge Graph (KG) is famous for knowledge representation, integration and reasoning. It has been playing a critical role in a variety of applications, including search engines, recommendation systems, question answering, personal assistants, and so on [19]. A KG is often largely composed of relational facts in the form of RDF triple (s, r, o), where s represents a subject entity, o represents an object entity, and r is a relation (a.k.a. object property). All these triples compose a multi-relational graph whose nodes correspond to entities and edges are labeled by relations. A KG also contains RDF triples that represent KG literals and meta information such as entity attributes and textual definition, via built-in or bespoke data and annotation properties such as *rdfs:label* and *rdfs:comment*. In addition to these facts (data), KGs are often accompanied by an ontological schema in RDFS[1] or OWL[2], which often defines entities' concepts (a.k.a. classes), properties, concept and property hierarchies, constraints (e.g., relation domain and range), and logical expressions such as relation composition. RDF, RDFS and OWL have a number of built-in vocabularies for representing these knowledge, such as *rdfs:subClassOf*, *rdf:type* and *owl:disjointWith*.

2.2 Zero-shot Image Classification

Image classification is a critical task in computer vision. Zero-shot image classification (ZS-IMGC) refers to predict images with new classes that have no labeled training images. In the literature of ZS-IMGC, case studies range from classifying general objects [8, 9] to classifying (fine-grained) objects in specific domains such as animals [22, 43], birds [42], and flowers [28]. Please see [41] for a comprehensive survey on ZS-IMGC studies.

To address new classes, some early ZS-IMGC works employ class attributes as side information, which describe objects' visual characteristics about e.g., colors and shapes, to model the relationships between classes. However, the attribute ignores the direct associations between classes, cannot represent complicated relationship and usually needs human labour for annotation. Some other works adopt the word embeddings of class names [10, 29], or the sentence embeddings and textual features of class descriptions [32] to model the inter-class relationship. Although such text is easy to access, it cannot represent logical or quantitative semantics, and is often quite noisy containing many irrelevant words.

Recently, several methods model the inter-class relationships via KG, with promising results achieved. [20, 41] adopt WordNet to represent the hierarchy of classes of images from ImageNet; [12, 26, 34] propose to use the common sense KG ConceptNet to introduce more relational knowledge; [14] extracts knowledge from DBpedia as a complement of the WordNet class hierarchy. However, all these KG-based ZS-IMGC studies are still preliminary in terms of both semantic sufficiency in the methodology and benchmarking in the evaluation. To bridge the gap of benchmarking and support research in utilizing different external knowledge, in this work, we contributed three resources, each of which can support ranging external knowledge settings with a KG that has incorporated not only class hierarchy, text and attributes, but also common sense class knowledge and logical relationships between classes.

2.3 Zero-shot Knowledge Graph Completion

KGs such as Wikidata and DBpedia mostly face the challenge of incompleteness, and thus KG completion (KGC), which is often defined to predict the subject, relation or object of a missing relational fact (triple), has been widely investigated. The KGC methods usually first embed entities and relations into vectors by e.g., geometric learning and Graph Neural Networks (GNNs), and then discover the missing facts in the vector space [33]. However, these methods can only predict entities and relations that have been associated in some training facts, but cannot address newly-added (unseen) entities and relations, which are quite common as KGs are often evolving. To this end, zero-shot KGC (ZS-KGC), which is to predict facts with entities or relations that have never appeared in the training facts, was recently proposed and has achieved quite much attention in recent years [17, 31, 36, 38]. Currently we consider unseen relations, and will extend it with unseen entities in the future.

There are relatively few ZS-KGC studies that aim at addressing unseen relations. Qin et al. [31] leveraged the features learned from relation textual descriptions, and extracted two benchmarks from NELL and Wikidata for evaluation. As in ZS-IMGC, text side information is noisy, with irrelevant words and ambiguous meanings. To support further studies for developing and comparing ZS-KGC methods that can utilize different kinds of external knowledge, we propose two ZS-KGC resources, each of which is associated with one KG composed of relational facts (i.e., **data graph**) as the target for completion (fact prediction), and one ontological schema (i.e., **schema graph**) as external knowledge. For the schema graph, we adopt some vocabularies in RDFS, such as *rdfs:domain*, *rdfs:range*, *rdfs:subPropertyOf* and *rdfs:comment*, to define domain and range constraints, hierarchy and textual description, etc for relations, and some vocabularies in OWL, such as *owl:inverseOf*, *owl:propertyChainAxiom* and *owl:SymmetricProperty*, to define logics such as relation inversion and relation composition.

[1] https://www.w3.org/TR/rdf-schema/
[2] https://www.w3.org/TR/owl-features/

Table 1: Statistics of ZS-IMGC benchmarks in terms of granularity (Gran.), number of attributes (#Att.), classes and images. S/U denote seen/unseen classes.

Datasets	Gran.	#Att.	#Classes	#Images		
					Training	Testing
			Total (S/U)	Total	S/U	S/U
ImNet-A	fine	85	80 (28/52)	77,323	36,400/0	1,400/39,523
ImNet-O	fine	40	35 (10/25)	39,361	12,907/0	500/25,954
AwA	coarse	85	50 (40/10)	37,322	23,527/0	5,882/7,913

3 RESOURCE CONSTRUCTION FOR ZS-IMGC

3.1 Images and Classes

We extract two benchmarks named ImNet-A and ImNet-O from ImageNet which is a large-scale image database organized according to the WordNet taxonomy [8]. Each class in ImageNet is matched to a WordNet node, and has hundreds and thousands of images. Due to a large number of hierarchical classes and a huge number of images, ImageNet is widely adopted in computer vision research as well as in ZSL research.

We focus on fine-grained ImageNet classes under several class families (groups) such as vehicles and dogs. In each class family, 1) seen classes are the classes in the ImageNet 2012 1K subset that is often used to train CNNs e.g., ResNet [18], while unseen classes are one-hop away in the class hierarchy; 2) the total number of seen and unseen classes is more than 5; 3) the connection between seen and unseen classes are dense; and 4) each class is associated with a Wikipedia page such that more information about this class can be accessed. In the end, we extracted 28 seen classes and 52 unseen classes from 11 families all about animals (e.g., *bees* and *foxes*) for a benchmark named ImNet-A. We also extracted a benchmark named ImNet-O for general object classification from 5 irrelevant class families (e.g., *snack food* and *fungi*). Table 1 shows detailed statistics of ImNet-A and ImNet-O.

In addition, we also re-use a very popular ZS-IMGC benchmark named Animals with Attributes (AwA) [43]. AwA is a coarse-grained dataset for animal classification that contains $37,322$ images from 50 animal classes, all of which can be matched to WordNet nodes. The original AwA benchmark has no KG, while in this work, we build a KG as its external knowledge.

3.2 External Knowledge and KG Construction

We collect different kinds of external knowledge, including class hierarchy, attribute, text, relational fact and logical expression, for ImNet-A, ImNet-O and AwA. For each benchmark, we then construct one KG which integrates all these external knowledge.

3.2.1 Class Hierarchy. We first extract the class hierarchy from WordNet whose class nodes are connected via the *super-subordinate* relation (a.k.a hyponymy or hypernomy relation). The class hierarchy is used as our KG backbone, and is formally represented by the RDFS vocabulary *rdfs:subClassOf*, as Fig. 1 shows. Each class's IRI (Internationalized Resource Identifier) is created following its original WordNet id, e.g., *zebra* from AwA has the IRI *AwA:n02391049*. The prefix "*AwA*" here refers to an ad-hoc namespace of our KG. Since WordNet contains a very large taxonomy, we extract a subset that covers all the benchmark classes and all their ancestors, using the WordNet interface in the Python package NLTK[3].

3.2.2 Class Attribute. Based on this structure, we then inject the annotated attributes of seen and unseen classes. Before injection, there is a need to establish the hierarchy of attributes. This is because some attributes describe the same aspect of objects. For example, attributes like *black*, *white* and *red* all describe the appearance color of objects, while *head*, *tail* and *claws* all describe the body parts of animals. The categorization of attributes on the one hand models richer relationships among attributes, and on the other hand, it is helpful for defining the relations between classes and attributes. With reference to the concept categorization in WordNet, we manually gather the attributes into different groups. For example, for the KG of AwA, we gather 17 attributes into the group about body parts, and 8 attributes into the group about colors.

We represent these attributes and attribute groups as nodes in KG, each of them also has a namespace specified by the dataset to which it belongs and a unique id defined by ourselves, as shown in Fig. 1. Given the data of attribute hierarchy, we connect attribute nodes to their group nodes via the relation *rdfs:subClassOf*, and define the relation edges from class nodes to attribute nodes according to the group to which the connected attribute belongs. For example, for class *zebra* and its one attribute *tail*, a relation edge named *AwA:hasBodyPart* is defined.

Regarding the collection of attribute annotation data, for AwA, we use its published attributes which are annotated by experts. Each AwA class has an associated binary-valued or continuous-valued attribute vector. To represent the annotated attributes of classes in graph without losing information, we adopt the binary-value vector and extract attributes whose corresponding vector slots are 1 for each class. For ImNet-A/O, we manually annotate attributes for the classes as the attributes of ImageNet classes are not available. Briefly, we prepare a list of related attributes gathered from Wikipedia pages and attribute annotations from other ZS-IMGC dataset such as AwA, and invite volunteers to assign $3 \sim 6$ attributes for each class with 25 images as references. Each class is independently reviewed by 3 volunteers and the final decision is made by voting. The statistics of attribute annotations of these three datasets are listed in Table 1.

3.2.3 Class Text. In addition to the structured triples, we also introduce the textual contents of KG entities (i.e., classes, attributes and attribute groups). We choose their natural language names, considering that some classes are hierarchically related and their names are similar. For example, classes *red fox*, *grey fox* and *kit fox* are children of class *fox*. The names of classes can also be looked up by NLTK WordNet interface. Finally, these texts are represented in the graph using relations e.g., *rdfs:label*, as Fig. 1 shows.

3.2.4 Relational Fact. We also access more relational class knowledge from a large scale common sense KG named ConceptNet [37] whose knowledge is collected from multiple resources including WordNet, DBpedia, etc. We use the English subgraph of its latest version (5.7)[4], which has over 3.4 million triples and around 1.8 million nodes in total. However, a lot of its knowledge is irrelevant

[3]https://www.nltk.org/howto/wordnet.html

[4]https://github.com/commonsense/conceptnet5/wiki/Downloads

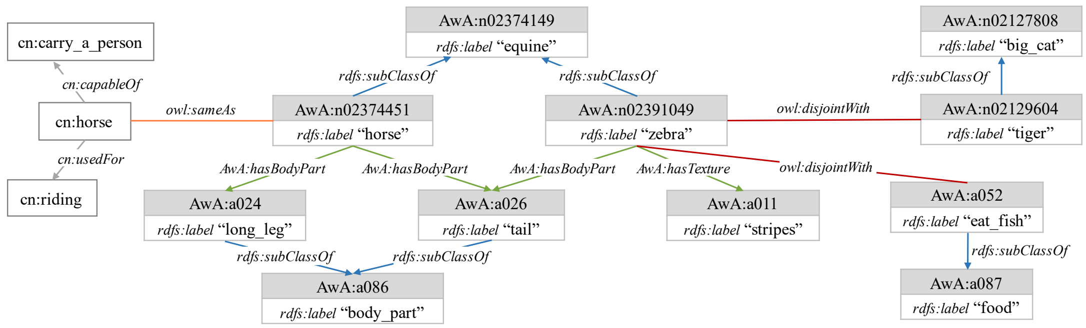


Figure 1: A snapshot of the KG of AwA. The prefixes *AwA* and *cn* are two ad-hoc namespaces of the KG.

Table 2: Statistics of different kinds of entities, relations and triples in the KGs of ZS-IMGC. "Hie.", "Att." and "CN" are short for Hierarchy, Attribute and ConceptNet, respectively.

Datasets	# Entities				# Relations			# Triples						
	Total	Class	Att.	CN	**Total**	Att.	CN	**Total**	Hie.	Att.	Literal	sameAs	CN	disjointWith
ImNet-A	8,920	111	103	8,706	41	17	21	10,461	210	335	214	156	9,546	/
ImNet-O	3,148	59	52	3,037	31	6	22	3,990	110	110	111	93	3,566	/
AwA	9,195	100	102	8,993	42	15	23	14,112	197	1,562	202	182	10,546	1,423

to our benchmarks. Therefore, we choose to extract a related subgraph from it. Specifically, we align the classes and attributes to entities of ConceptNet and query their 1-hop neighbors.

Since entities in ConceptNet are words and phrases of natural language, we use the literal names of classes and attributes and adopt string matching for alignment. For example, a class *zebra* is aligned to *c/en/zebra* in ConceptNet. For some attributes that have no corresponding entities due to the different word forms, we first lemmatize them. For example, an attribute *spots* is lemmatized to *spot* that can be found in ConceptNet. Besides, we also find that some entities in ConceptNet share the same prefix, e.g., the *c/en/zebra* and *c/en/zebra/n/wn/animal*. Targeting this, we merge them using a custom namespace "*cn*", e.g., the above two entities are merged as *cn:zebra*, and extract the union of these entities' neighborhood. To be unified, other entities are also represented with this namespace. As a result, for the aligned pairs, we use a vocabulary *owl:sameAs* defined in OWL to relate them in the graph. According to the statistics in Table 2, we find that there are still some classes or attributes that have no corresponding ConceptNet entities, it may be because they are fined-grained concepts which are not included in ConceptNet yet. We choose to skip them and leave their knowledge extraction as future work. In addition, to reduce the noise during neighborhood extraction, we ignore the triples with irrelevant relations, e.g., Synonym, Antonym, SymbolOf, NotCapableOf and NotHasProperty.

However, to use the extracted subgraph, there are still some issues to be addressed. One issue is that the relations extracted from ConceptNet may have the same semantics with the relations previously defined. For example, *cn:isA* and *rdfs:subClassOf* both indicate the concept hierarchy. Targeting this, we unify them into *rdfs:subClassOf*. The other issue is that the knowledge extracted from ConceptNet may already exist. An example is (*cn:squirrel*, *cn:LocatedNear*, *cn:tree*), which is already modelled by the attribute triple: (*AwA:squirrel*, *AwA:hasHabitat*, *AwA:tree*). To solve this, we extract the subjects and objects of these triples to generate a set of tuples (s, o), and filter out ConceptNet triples with repetitive tuples.

3.2.5 Logical Expression. In ZS-IMGC, we found that some classes that belong to different families and look greatly different have many identical attributes. For example, animal classes *zebra* and *tiger* both have attributes like *stripes*, *tail* and *muscle*. During inference, *tiger* may provide a significant contribution to the feature learning of *zebra* due to too many shared attributes between them. Although their parent classes (i.e., *equine* and *big cat*) and name information have been introduced, more direct information to distinguish them would benefit the model and should be investigated. One kind of semantics that can be expressed by a KG for further augmenting ZSL is the logical relationship defined using the OWL vocabulary. For the ZS-IMGC resources, we added the disjointness axioms between classes using the built-in property *owl:disjointWith*. In the above example, we define the disjointness between *zebra* and *tiger* to constrain that an images of *zebra* cannot simultaneously be the instance of *tiger* so that avoiding the misclassification, as shown in Fig. 1. We also define the disjointness between classes and attributes. For example, the fact "Zebra doesn't eat fish" means the disjointness between class *zebra* and attribute *eat fish*.

Considering that in ImNet-A/O, the overlap of attributes of classes in different families is low, we mainly set the disjoint constraints for classes and attributes in AwA. For the disjointness between different classes, we first generate a candidate set by counting the number of shared attributes. To be more specific, for a pair of classes that belong to different families, if their shared attributes are more than 2/3 of the attributes of class that has fewer attributes, we set a candidate disjoint relationship between them. Then, to ensure the correctness of these disjoint pairs, we invite volunteers for a check. For defining the disjointness between classes and attributes, we use the continuous-valued attribute vectors of AwA, and each class is disjoint with attributes whose vector values are 0.

Table 3: Statistics of ZS-KGC datasets. "Tr/V/Te" is short for training/validation/testing. "#Ent." denotes the number of entities.

Datasets	#Ent.	#Relations Total (Tr/V/Te)	#Triples Total (Tr/V/Te)
NELL-ZS	65,567	181 (139/10/32)	188,392 (181,053/1,856/5,483)
Wikidata-ZS	605,812	537 (469/20/48)	724,928 (701,977/7,241/15710)

3.2.6 KG Overview. The statistics of the resulting KGs are shown in Table 2. Each KG is composed of RDF triples which are stored in a CSV file with three columns corresponding to subjects, relations and objects. The KG files can easily be accessed by Python libraries or be loaded into graph stores such as RDFox [27]. Regarding the images, we follow previous survey work [41] and provide ResNet features which are stored as a matrix, whose two dimensions correspond to feature vector length and image number, respectively.

4 RESOURCE CONSTRUCTION FOR ZS-KGC

4.1 KGs for Completion

We employ two ZS-KGC datasets (i.e., two sub-KGs for completion) proposed by [31]. They are NELL-ZS extracted from NELL[5] and Wikidata-ZS extracted from Wikidata[6]. In both datasets, relations are divided into two disjoint sets: a seen relation set $\mathcal{R}_s$ and an unseen relation set $\mathcal{R}_u$. In training, a set of facts (RDF triples) of the seen relations are used, while in testing, the model predicts the facts involving unseen relations in $\mathcal{R}_u$. A closed set of entities are considered in both datasets, which means each entity in the testing triples has already appeared in the training triples. Besides, a subset of training facts is left out as the validation set by removing all training facts of the validation relations. The statistics of these two datasets are shown in Table 3.

4.2 Ontological Schema

We build an ontological schema as side information for each dataset. It mainly includes semantics expressed by RDFS (concept hierarchy, relation hierarchy, relation domain and range), semantics expressed by OWL (relation characteristics and inter-relation relationships), and textual meta data (e.g., concept and relation names and descriptions). Note concept here refers to entity type/class. In this paper, the sub-KG for completion is also named as a data graph, while its ontology schema is represented as a schema graph.

4.2.1 Semantics in RDFS. Semantics by RDFS vocabularies act as the backbone of the schema graph. Different from the data graph where relations act as edges between nodes (i.e., predicates in RDF triples), in the schema graph, relations act as nodes (i.e., subjects or objects in RDF triples). Specifically, we use the following vocabularies: *rdfs:subPropertyOf*, *rdfs:domain*, *rdfs:range*, and *rdfs:subClassOf* to define relation semantics and generate corresponding triples:

- (r_1, *rdfs:subPropertyOf*, r_2), subproperty triple, states the hierarchical relationships between relations, i.e., relation r_1 is a subrelation of relation r_2;
- (r, *rdfs:domain*, C_s), domain triple, summarizes the subject entity type (i.e., subject concept) C_s of relation r;
- (r, *rdfs:range*, C_o), range triple, summarizes the object entity type (i.e., object concept) C_o of relation r;
- (C_i, *rdfs:subClassOf*, C_j), subclass triple, states the hierarchical relationships between entity types C_i and C_j.

A snapshot of the schema graph of NELL-ZS is shown in Fig. 2. As we can see, the schema graph's nodes are relations and entity concepts of the data graph. We also add dataset specific namespace for these nodes. Besides, the edge in the schema graph, i.e., the relationship between relations, is called as "meta-relation".

For the schema graph of NELL-ZS, the aforementioned semantics in RDFS are extracted from NELL's ontology. The ontology is saved and published as a CSV file[7] which has three columns corresponding to subjects, predicates and objects of RDF triples. From these triples, we extract domain and range triples according to the predicates "domain" and "range", respectively, and extract subproperty and subclass triples according to the predicate "generalizations". For the schema graph of Wikidata-ZS, the concept/relation hierarchy, relation domains and ranges can be accessed from Wikidata by a tookit implemented in Python[8]. Specifically, we look up a relation's super-relations by Wikidata property P1647 (subproperty of), look up a relation's domain concepts and range concepts by Wikidata property P2302 (property constraint) with constraints Q21503250 (type constraint) and Q21510865 (value-type constraint), respectively, and look up a concept's super-concepts by Wikidata property P279 (subclass of).

4.2.2 Semantics in Text. We further enrich the schema graphs with textual semantics of the nodes (i.e., relations and concepts), which usually act as important side information in addressing ZS-KGC [13, 31, 36]. For NELL-ZS, they also can be extracted from its original ontology file. Specifically, we use predicate "description" to extract the textual descriptions of relations and concepts. For Wikidata-ZS, we look up the textual names and textual descriptions of relations and concepts from Wikidata using properties *label* and *description*, respectively. These texts are represented in graph by RDFS vocabularies *rdfs:label* and *rdfs:comment*, leading to a literal-aware schema graph.

4.2.3 Semantics in OWL. We also introduce relation semantics from OWL documents, including semantic relationships between relations and characteristics of relations. We provide an overview illustration in Table 4. Next, we will introduce one by one.

Inverse Relationship. The inverse relationship between two relations is defined by *owl:inverseOf*. If r_1 is an inverse relation of r_2 or r_2 is an inverse relation of r_1, when a fact (e_1, r_1, e_2) holds, the fact (e_2, r_2, e_1) also holds, and vice versa. In making our ontological schemas for ZS-KGC, we introduce the inverse relationships between seen and unseen relations, and construct inverse triples in format of (r_1, *owl:inverseOf*, r_2). In predicting triples with unseen relations, the potential ZS-KGC models can benefit from the unseen relations' inverse relations which are involved in training triples. Since the inverse relations have been removed from NELL-ZS during dataset construction, We only extract inverse triples for Wikidata. For relations in Wikidata-ZS, we get their inverse relations via Wikidata property P1696 ("inverse property").

[5] http://rtw.ml.cmu.edu/rtw/
[6] https://www.wikidata.org/
[7] http://rtw.ml.cmu.edu/resources/results/08m/NELL.08m.1115.ontology.csv.gz
[8] https://pypi.org/project/Wikidata/

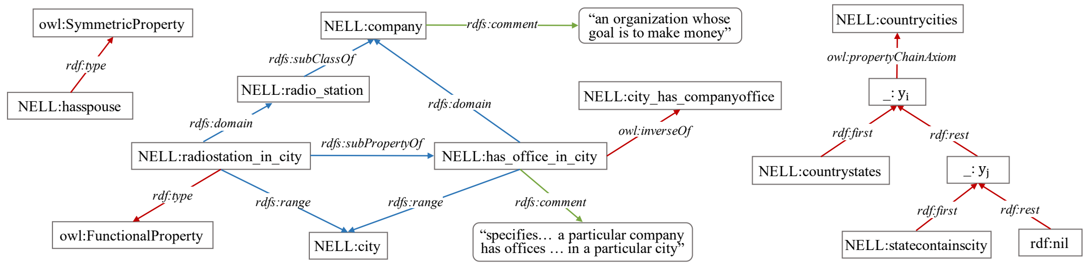


Figure 2: A snapshot of the constructed schema graph for ZS-KGC dataset NELL-ZS. $\_:y_i$ and $\_:y_j$ denote two blank nodes.

Table 4: Illustrations and statistics of inter-relation relationships and relation characteristics in the ontological schemas of NELL-ZS and Wikidata-ZS. "[$NELL$]" and "[$Wikidata$]" denote the example comes from NELL-ZS and Wikidata-ZS, respectively.

OWL Semantics	Formula	Example	Statistics NELL-ZS	Wikidata-ZS
Inversion	$(?x, r_1, ?y) \Leftrightarrow (?y, r_2, ?x)$	P802 (student) & P1066 (student of) [Wikidata]	0	39
Composition	$(?x, r_1, ?y) \wedge (?y, r_2, ?z) \Rightarrow (?x, r_3, ?z)$	countrystates ∧ statecontainscity ⇒ countrycities [NELL]	20	7
Symmetry	$(?x, r, ?y) \Rightarrow (?y, r, ?x)$	hasspouse [NELL]	20	25
Asymmetry	$(?x, r, ?y) \Rightarrow \neg(?y, r, ?x)$	subpartof [NELL]	24	11
Reflexivity	$(?x, r, ?x)$	animalpreyson [NELL]	2	0
Irreflexivity	$\neg(?x, r, ?x)$	P184 (doctoral advisor) [Wikidata]	46	15
Functional	$(?x, r, ?y) \wedge (?x, r, ?z) \Rightarrow ?y = ?z$	airportincity [NELL]	6	15
Inverse Functional	$(?x, r, ?y) \wedge (?z, r, ?y) \Rightarrow ?x = ?z$	statecontainscity [NELL]	16	1

Compositional Relationship. In an OWL ontology, a relation can be constructed by ordered composition of several other relations. For example, *uncleOf* is the composition of *brotherOf* and *parentOf*. It means if x is a brother of y and y is a parent of z, then x is an uncle of z. Formally, the semantics of relation composition in a KG can be defined as

$$(?x, r_1, ?y) \wedge (?y, r_2, ?z) \Rightarrow (?x, r_3, ?z) \quad (1)$$

where r_1, r_2 and r_3 denote three relations; $?x$, $?y$ and $?z$ denote three variables of entities. The composition is also simply denoted as $r_1 \wedge r_2 \Rightarrow r_3$, and is represented by a composition axiom in an OWL ontology (see the rightest part in Fig. 2 for a composition axiom that has been serized as RDF triples with blank nodes according to W3C OWL to RDF graph mapping standard[9]). In our ontologies for NELL-ZS and Wikidata-ZS, we define some relations especially some unseen relations as compositions of seen relations, where we limit the number of seen relations used for each composition to 2.

For NELL-ZS, we first extract a set of candidate relation compositions via checking relation domain and range. Specifically, for any three relations r_1, r_2 and r_3, if the range of r_1 is the domain of r_2, the domain of r_1 is also the domain of r_3 and the range of r_2 is also the range of r_3, then $r_1 \wedge r_2 \Rightarrow r_3$ is regarded as a candidate composition. These candidates can be extracted according to the schema of NELL defined by RDFS. Briefly, for each relation of NELL-ZS, we traverse all seen relation pairs and check whether the domains and ranges of the two seen relations and the current relation match the above condition.

Some candidate relation compositions extracted in the above step are not correct; for example, *motherofperson*∧*personalsoknownas* ⇒ *wifeof* does not match common sense. Targeting this, we manually check these candidates. Briefly, each candidate is independently reviewed by three volunteers (including one of the authors and two of our colleagues who are familiar with KGs and ontologies) and the final decision is made by voting. It is also allowed that volunteers can look up all the information about relations such as triples and descriptions during review.

However, the method of checking relation domain and range can not be well applied to Wikidata-ZS because most Wikidata-ZS relations have more than one domain or range constraints, which will result in too many candidates and put great pressure on manual assessment. Therefore, for Wikidata-ZS, we use AMIE [11], a KG rule mining system, to automatically extract a candidate set of logical rules in form of (1) from facts as candidate relation compositions, where each rule is associated with a confidence score. We then keep those rules whose scores are above 0.9. As in constructing relation compositions for NELL-ZS, we finally ask volunteers to manually check the remaining rules, and transform the correct ones into relation composition axioms for the schema of Wikidata-ZS.

Symmetric & Asymmetric Relations. In a KG, a relation r is symmetric if we have $(?y, r, ?x)$ given $(?x, r, ?y)$. One typical examle is *hasSpouse*. In contrast, a relation r is defined as asymmetric if $r(?y, r, ?x)$ is always false given $(?x, r, ?y)$. We add symmetric and asymmetric characteristics to some relations in our schemas of NELL-ZS and Wikidata-ZS, because they could be utilized by potential ZSL methods for addressing ZS-KGC with unseen relations. For example, when an unseen relation is symmetric, during prediction, some of its testing triples could be inferred according to some other triples that have been correctly predicted. The asymmetric property could reduce the searching space of candidate entities during prediction. Namely, entity y can be directly excluded from being the object of the tuple (x, r) without predicting if the triple (y, r, x)

[9]https://www.w3.org/TR/owl2-mapping-to-rdf/

Table 5: Number of relations, concepts, literals, meta-relations, and different axioms in the ontological schema.

Datasets	# relations	# concepts	# literals	# meta-relations	# subproperty	# domain	# range	# subclass	# relation characteristics
NELL-ZS	894	292	1,063	9	935	894	894	332	114
Wikidata-ZS	560	1,344	3,808	11	208	1,843	1,378	1,392	67

holds. In our resource, we mainly add asymmetric characteristics to relations that have identical domains and ranges.

To add symmetric and asymmetric characteristics to some relations in NELL-ZS, we use the predicate "anti-symmetric" defined in the original ontology file of NELL. Specifically, for symmetric relations, we first extract relations whose "anti-symmetric" values are false, and then select those with the same domain and range. Some of the resultant relations are still not symmetric, and we invite volunteers to filter out them. For asymmetric relations, they can be automatically extracted by the predicates "anti-symmetric" and "irreflexive" considering that a relation is asymmetric iff it is antisymmetric and irreflexive. For relations of Wikidata-ZS, the symmetric characteristic is extracted by looking up the Wikidata constraint Q21510862 (symmetric constraint) stated in P2302 (property constraint). While for the asymmetric characteristic, we extract relations which have identical domain and range, and then manually check whether a relation is asymmetric or not.

We use OWL built-in vocabularies *owl:SymmetricProperty* and *owl:AsymmetricProperty* to represent relation symmetric and asymmetric characteristics in our ontological schemas. When the ontologies are represented as schema graphs, relation characteristics are transformed into RDF triples like (r, *rdf:type*, *owl:SymmetricProperty*) which means r is a symmetric relation.

Reflexive & Irreflexive Relations. A relation r is regarded as reflexive if $(?x, r, ?x)$ holds, and as irreflexive if $(?x, r, ?x)$ does not hold, where $?x$ is an entity variable. As symmetric and asymmetric characteristics, reflexive and irreflexive characteristics could be utilized by potential ZSL methods for addressing unseen relations in KGC. If an unseen relation r is reflexive, we can directly infer a testing triple (x, r, x) for any entity x; if r is irreflexive, we can exclude entity x as the object of the tuple (x, r) in prediction.

We use the predicate "anti-reflexive" defined in NELL's original ontology file to add reflexive and irreflexive characteristics for relations in NELL-ZS, i.e., relations annotated with false are reflexive while those annotated with true are irreflexive. For relations in Wikidata-ZS, since Wikidata has no definitions towards these two characteristics, we extract relations that have identical domain and range, and manually check relations' reflexivity and irreflexivity. The representation of the reflexive and irreflexive characteristics in the schema graph is the same as the symmetric and asymmetric characteristics, but uses the OWL built-in vocabularies *owl:ReflexiveProperty* and *owl:IrreflexiveProperty*, respectively.

Functional & Inverse Functional Relations. Given a functional relation r, if $(?x, r, ?y)$ and $(?x, r, ?z)$ holds, then $?y$ and $?z$ must represent the same entity. Namely every entity can be related to at most one entity via a functional relation. A relation can furthermore be inverse functional, meaning that its inverse relation is functional. We consider both functional and inverse functional relations in our ontologies. They can constrain the prediction space of testing triples. For example, for a testing tuple (x, r) with functional relation r, its ground truth object entity is unique.

To extract these two characteristics, we look up the whole triples of the original NELL[10] and Wikidata (via its SPARQL Endpoint), and extract relations whose object entity is unique for the same subject for the functional characteristic, and relations whose subject entity is unique for the same object for the inverse functional characteristic. They are represented in the same way as the symmetric characteristic, but use built-in vocabularies *owl:FunctionalProperty* and *owl:InverseFunctionalProperty*, respectively.

4.2.4 Schema Overview. The statistics of the resulting ontological schemas of NELL-ZS and Wikidata-ZS are shown in Table 5. Each ontological schema is saved in two formats. The first is the original ontology file ended with .owl. It can be directly loaded and easily viewed by ontology editors such as Protege. The second is an RDF triple file to save the schema graph which is transformed from the ontology according to W3C OWL to RDF graph mapping. It is convenient for graph embedding methods e.g., GNNs and KG embedding algorithms to process. Note other mappings from OWL ontology to RDF graph can be considered by the resource user.

5 RESOURCE APPLICATION

5.1 Evaluating ZSL Model Performance

We first show the usage of our resources in evaluating and comparing different ZS-IMGC and ZS-KGC methods under different side information (i.e., external knowledge) settings.

5.1.1 ZS-IMGC. With the complete KGs of the ZS-IMGC benchmarks, we made the following four external knowledge settings which have different semantics:

- **Basic KG**: Hierarchy and attribute triples (i.e., the class hierarchy, class attributes and the attribute hierarchy).
- **Basic KG + literals**: Basic KG plus textual information.
- **Basic KG + CN**: Basic KG plus relational facts from ConceptNet.
- **Basic KG + logics**: Basic KG plus disjointness axioms.

To apply these external knowledge in ZSL models, we take advantage of some semantic embedding techniques to encode them and generate a vector representation for each class. Specifically, we adopt mature and widely-used TransE [1] to encode the structure knowledge contained in **Basic KG**, **Basic KG + CN** and **Basic KG + logics**. For **Basic KG + literals**, we adopt a text-aware graph embedding method used in [13] and [25] to simultaneously embed the text and structure knowledge.

Besides the above KG-based external knowledge settings, we also made the following simple but widely used settings. Relevant side information can be extracted from the original benchmarks, but has been included in our new resources:

- **Class Attributes (att)**: 85, 85 and 40 dimensional binary-valued (multi-hot) attribute vectors for classes of AwA, ImNet-A and ImNet-O, respectively.

[10] http://rtw.ml.cmu.edu/resources/results/08m/NELL.08m.1115.esv.csv.gz

Table 6: *Accuracy* (%) of DeViSE and OntoZSL on AwA.

External Knowledge	DeViSE *acc*	DeViSE acc_s	DeViSE acc_u	DeViSE *H*	OntoZSL *acc*	OntoZSL acc_s	OntoZSL acc_u	OntoZSL *H*
w2v (500)	24.22	78.42	1.05	2.08	45.21	57.83	34.53	43.24
w2v (300)	8.42	86.32	0.00	0.00	20.89	44.76	9.50	15.67
att	37.46	81.06	3.29	6.32	58.52	60.16	43.70	50.62
hie	43.50	65.25	5.60	10.32	38.89	51.08	31.38	38.88
Basic KG	43.24	86.44	6.40	11.91	**62.46**	**63.50**	**47.97**	**54.65**
Basic KG + literals	**46.12**	84.42	**8.76**	**15.88**	59.21	62.39	45.55	52.66
Basic KG + CN	44.39	**88.91**	0.68	1.35	48.56	62.82	37.50	46.96
Basic KG + logics	36.86	81.32	0.71	1.41	54.65	65.37	40.76	50.21

Table 7: *Accuracy* (%) of DeViSE and OntoZSL on ImNet-A.

External Knowledge	DeViSE *acc*	DeViSE acc_s	DeViSE acc_u	DeViSE *H*	OntoZSL *acc*	OntoZSL acc_s	OntoZSL acc_u	OntoZSL *H*
w2v (500)	13.52	59.71	0.63	1.25	20.87	40.14	13.90	20.65
w2v (300)	26.95	**84.36**	0.16	0.32	27.76	40.50	20.40	27.13
att	**35.39**	77.07	4.21	7.98	37.87	40.71	24.00	30.20
hie	30.94	62.07	1.67	3.25	33.32	**40.93**	23.06	29.50
Basic KG	34.36	26.79	28.75	**27.74**	37.92	40.43	25.00	30.90
Basic KG + literals	33.62	23.36	**29.33**	26.01	**38.58**	35.64	**27.64**	**31.13**
Basic KG + CN	33.95	68.86	8.38	14.94	34.43	38.14	25.35	30.46

- **Class Word Embeddings (w2v)**: Two word vectors considered for each class via its name. One is by [2] with a dimension of 500; the other is by a Glove model with a dimension of 300.
- **Class Hierarchy (hie)**: A 100 dimensional vector for each class, encoded by a graph auto-encoder [21] over the class hierarchy.

We select DeViSE [10] and OntoZSL [13] as two ZSL methods for comparison, since DeViSE is a classic and very representative method with good performance achieved on many tasks, while OntoZSL is a state-of-the-art method that was originally developed to utilize KG and ontology external knowledge. Briefly, DeViSE learns a mapping function from the image feature to the class embedding, and in testing, it searches for the nearest class in the class embedding space as the label of an image. OntoZSL leverages a Generative Adversarial Network (GAN) [16] to synthesize samples (image features) for unseen classes conditioned on their class embeddings. Note both methods are compatible to different external knowledge that have been embedded.

The results on AwA and ImNet-A are shown in Table 6 and Table 7, respectively, where we calculate the accuracy of each class and report the averaged accuracy of the seen and/or unseen classes, following the standard in ZSL community. Note we considered two ZSL settings: standard ZSL, where only unseen classes and their samples are tested with the averaged accuracy denoted as *acc*; generalized ZSL, where both seen and unseen classes, and their testing samples are tested, with the accuracy averaged over the seen and unseen classes denoted as acc_s and acc_u, respectively, and a harmonic mean $H = (2 \times acc_s \times acc_u)/(acc_s + acc_u)$ calculated as an overall metric.

The results of AwA (cf. Table 6) show that the KG-based external knowledge settings always achieve better performance than those simple and currently widely used settings – att, w2v and hie. The results of ImNet-A (cf. Table 7) show that although the KG-based setting do not always achieve the best performance w.r.t.

Table 8: OntoZSL Results (%) on NELL-ZS and Wikidata-ZS.

External Knowledge	NELL-ZS *MRR*	NELL-ZS *hit@* 10	NELL-ZS *hit@* 5	NELL-ZS *hit@* 1	Wikidata-ZS *MRR*	Wikidata-ZS *hit@* 10	Wikidata-ZS *hit@* 5	Wikidata-ZS *hit@* 1
Text	21.5	34.5	28.3	14.5	18.5	27.3	22.3	13.5
RDFS graph	22.3	35.1	29.1	15.3	18.5	27.5	22.3	13.4
RDFS + literals	**22.7**	**35.6**	**29.4**	**15.6**	**18.8**	**28.1**	**22.6**	**13.5**

all the metrics, the results are still comparable to the simple settings. All these illustrate the potential of KG-based external knowledge in ZS-IMGC. Besides, with TransE which was developed for KGs with relational facts alone and a simple workflow that has semantic embedding and ZSL model detached, although more semantics are introduced in **Basic KG + CN** and **Basic KG + logics**, their results are not better than **Basic KG**. This motivates the community to develop more effective ZS-IMGC techniques to utilize all these promising semantics for better performance, and our resource provides a chance for such investigation.

5.1.2 ZS-KGC. For NELL-ZS and Wikidata-ZS, we made three external knowledge settings via our new KG resources, and one setting that has already been used with the original benchmark:

- **Text**: Relations' textual descriptions which were originally proposed and used by Qin et al. [31].
- **RDFS graph**: subproperty, domain, range and subclass triples (i.e., relation semantics defined by RDFS);
- **RDFS + literals**: RDFS graph plus textual meta data of concepts and relations;
- **RDFS + OWL**: RDFS graph plus relation semantics in OWL.

As in ZS-IMGC, we embed the external knowledge and apply the resultant relation vectors in ZS-KGC methods. For **RDFS graph** and **RDFS + literals**, we adopt TransE and the text-aware graph embedding method used in [13] and [25], respectively, for embedding. To embed the textual descriptions, we follow [31] and perform a weighted summation of the embeddings of words in descriptions, where word embeddings[11] of dimension 300 pre-trained on Google News is used. We first compare these three external knowledge settings using the state-of-the-art method OntoZSL which utilizes GAN to generate features for unseen relations conditioned on relation embeddings.

The results on NELL-ZS and Wikidata-ZS are shown in Table 8. Note the specific KGC task is to predict the object entity given a subject entity and a relation, i.e., ranking the candidates according to their likelihood to be the object. Two commonly used metrics are adopted: mean reciprocal ranking (*MRR*) which is the average of the reciprocal predicted ranks of all the ground truths, and *hit@k* which is the ratio of testing samples whose ground truths are ranked in the top-k positions (k is set to 1, 5, 10) [39]. From Table 8, we can find that **RDFS graph** and **RDFS + literals** both lead to better performance than **Text**, and **RDFS + literals** performs the best.

The external knowledge defined by OWL is quite promising for augmenting ZSL, but there are current no systemic or robust methods. We try to validate the effectiveness of OWL semantics by testing an ensemble method, which combines symbolic reasoning and embedding-based prediction, under the setting of **RDFS +**

[11] https://github.com/mmihaltz/word2vec-GoogleNews-vectors

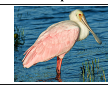
	Unseen Classes **horse**	**Contributing Seen Classes** **zebra**	**Unseen Class** **roseate spoonbill**	**Contributing Seen Classes** **spoonbill**
Image				
General Explanations	The prediction for samples of **horse** is supported by **zebra**.		The prediction for samples of **roseate spoonbill** is supported by **spoonbill**.	
Knowledge from Class Hierarchy	**Triples:** (horse, *rdfs:subClassOf*, equine), (zebra, *rdfs:subClassOf*, equine) **Generated Explanations:** Horse and zebra both belong to equine.		**Triples:** (roseate_spoonbill, *rdfs:subClassOf*, spoonbill) **Generated Explanations:** Roseate spoonbill is a member of spoonbill	
Knowledge from Class Attributes	**Triples:** (horse, *AwA:hasBodyPart*, long leg), (zebra, *AwA:hasBodyPart*, long leg), (horse, *AwA:hasBodyPart*, tail), (zebra, *AwA:hasBodyPart*, tail), (horse, *AwA:hasTexture*, furry), (zebra, *AwA:hasTexture*, furry), (horse, *AwA:hasBehavior*, group), (zebra, *AwA:hasBehavior*, group), … **Generated Explanations:** They both have long leg and tail, are both furry, and both like to live in a group, …		**Triples:** (roseate_spoonbill, *ImNet-A:hasBodyPart*, long thin legs), (spoonbill, *ImNet-A:hasBodyPart*, long thin legs), (roseate_spoonbill, *ImNet-A:hasBill*, white long flat and wider bill), (spoonbill, *ImNet-A:hasBill*, long flat and wider bill), (roseate_spoonbill, *ImNet-A:hasHabitat*, water), (spoonbill, *ImNet-A:hasHabitat*, water), … **Generated Explanations:** They both have long thin legs, are similar in bill, and both live in water, ….	
Knowledge from ConceptNet	**Triples:** (cn:horse, *rdfs:subClassOf*, cn:herd_animal), (cn:zebra, *rdfs:subClassOf*, cn:herd_animal), (cn:horse, *cn:hasA*, cn:eyes), (cn:zebra, *cn:hasA*, cn:eyes) **Generated Explanations:** They are both herd animals, and both have eyes.		**Triples:** (cn:roseate_spoonbill, *rdfs:subClassOf*, cn:bird), (cn:spoonbill, *rdfs:subClassOf*, cn:bird) **Generated Explanations:** They are both a type of bird.	

Figure 3: Examples for explaining why features of seen classes zebra and spoonbill are transferred to unseen classes horse and roseate spoonbill, respectively. Note we have replaced human unreadable entity ids by entity names.

OWL. Briefly, for a testing tuple (subject and relation), if the object can be inferred through logical expressions (cf. Section 4.2.3), we adopt the inferred object; otherwise, we use the predicted object ranking by OntoZSL. Even with such a naive ensemble solution, we got some encouraging results. On Wikidata-ZS, 5 unseen relations have inverse seen relations, and *hit*@1 increases from 28.6% to 55.3% when logical inference with the inverse semantics is used. On NELL-ZS, 4 unseen relations are composed by 10 seen relations, and the logical inference with composition leads to 6.3% increment on *hit*@1. These results demonstrate the superiority of the OWL-based relation semantics, although the method of utilizing OWL axioms presented here is quite preliminary. With our resources, more robust methods can be investigated to better utilize such logical expressions and significantly augment ZSL performance.

5.2 Evaluating ZSL Model Explanation

Since argumentative explanation by additional domain or background knowledge is a promising solution and has been widely investigated [5–7], our ZSL resources with rich external knowledge can also be used to evaluate ZSL model interpretation and prediction justification. For demonstration, we use different external knowledge settings that can be made by our resources to support a knowledge augmented ZSL explanation method named X-ZSL which explains the transferability of sample features in ZSL in a human understandable manner [14]. Briefly, X-ZSL first uses an Attentive Graph Neural Network to automatically learn seen classes that are contributing to the feature learning of an unseen class, then explains the feature transfer between them by extracting class knowledge from external KGs, and finally uses some templates to generate human readable natural language explanations.

Fig. 3 presents the explanations by X-ZSL on AwA and ImNet-A, with different knowledge extracted from our KG resources. As we can see, for the feature transferability between seen and unseen classes, the knowledge from class hierarchy provides overall explanations, from the perspective of their relatedness in biology; the knowledge from class attributes provides detailed explanations, from the perspective of their relatedness in characteristics especially in visual characteristics; while the relational facts from ConceptNet provide an important supplement. In summary, different semantics in our resources all have positive contributions to explain a ZSL model or to justify a ZSL prediction, and thus different methods can be developed and compared with using our resources.

6 CONCLUSION AND FUTURE WORK

External knowledge, especially KG, plays a critical role in ZSL. To address the issues of semantic insufficiency in existing ZSL resources and lacking standard benchmarks to fairly compare KG-based methods, we created systemic resources for KG-based research on ZS-IMGC and ZS-KGC, including 5 standard ZSL datasets and their corresponding KGs. For KGs of ZS-IMGC, we integrate not only typical side information such as class hierarchy, attributes and text, but also common sense relational facts from ConceptNet and some logical expressions such as class disjointness. Regarding ZS-KGC, we build ontological schemas with semantics defined by RDFS and OWL (e.g., relation hierarchy, relation domain and range, concepts, relation characteristics, and relation and concept textual meta data) for a NELL KG and a Wikidata KG that are to be completed. The usage of these resources have been demonstrated by evaluating the performance and explanation of several ZSL methods.

In the future, we plan to leverage these resources to develop more effective semantic embedding methods to fully utilize all kinds of external knowledge and develop more robust ZSL methods. We also plan to make a systemic evaluation towards many ZSL methods, especially those utilizing KGs, and compare their component techniques including multi-relation graph embedding such as R-GCN [35] and HAKE [45], ontology embedding such as OWL2Vec* [4], multi-modal KG embedding which can take kinds of meta data such as numeric values, text and even images into consideration [15]. Meanwhile, we are continuously extending our resources to support the following aspects in order: a new task on visual question answering with unseen answers, ontological schemas for NELL-ZS and Wikidata-ZS with more complex logical relationships, and a new ZS-KGC task with unseen entities.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems (NIPS)*. 1–9.

[2] Soravit Changpinyo, Wei-Lun Chao, Boqing Gong, and Fei Sha. 2016. Synthesized classifiers for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5327–5336.

[3] Jiaoyan Chen, Yuxia Geng, Zhuo Chen, Ian Horrocks, Jeff Z Pan, and Huajun Chen. 2021. Knowledge-aware Zero-Shot Learning: Survey and Perspective. *arXiv preprint arXiv:2103.00070* (2021).

[4] Jiaoyan Chen, Pan Hu, Ernesto Jimenez-Ruiz, Ole Magnus Holter, Denvar Antonyrajah, and Ian Horrocks. 2021. OWL2Vec*: Embedding of OWL ontologies. *Machine Learning* (2021).

[5] Jiaoyan Chen, Freddy Lécué, Jeff Z. Pan, Ian Horrocks, and Huajun Chen. 2018. Knowledge-Based Transfer Learning Explanation. In *KR*. AAAI Press, 349–358.

[6] Roberto Confalonieri, Tillman Weyde, Tarek R Besold, and Fermín Moscoso del Prado Martín. 2021. Using ontologies to enhance human understandability of global post-hoc explanations of black-box models. *Artificial Intelligence* 296 (2021), 103471.

[7] Kristijonas Čyras, Antonio Rago, Emanuele Albini, Pietro Baroni, and Francesca Toni. 2021. Argumentative XAI: A Survey. *arXiv preprint arXiv:2105.11266* (2021).

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 248–255.

[9] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. 2009. Describing objects by their attributes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1778–1785.

[10] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. 2013. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*. 2121–2129.

[11] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the 22nd international conference on World Wide Web*. 413–422.

[12] Junyu Gao, Tianzhu Zhang, and Changsheng Xu. 2019. I know the relationships: Zero-shot action recognition via two-stream graph convolutional networks and knowledge graphs. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 8303–8311.

[13] Yuxia Geng, Jiaoyan Chen, Zhuo Chen, Jeff Z. Pan, Zhiquan Ye, Zonggang Yuan, Yantao Jia, and Huajun Chen. 2021. OntoZSL: Ontology-enhanced Zero-shot Learning. In *The Web Conference (WWW)*.

[14] Yuxia Geng, Jiaoyan Chen, Zhiquan Ye, Wei Zhang, and Huajun Chen. 2020. Explainable Zero-shot Learning via Attentive Graph Convolutional Network and Knowledge Graphs. *Semantic Web* (2020).

[15] Genet Asefa Gesese, Russa Biswas, Mehwish Alam, and Harald Sack. 2019. A survey on knowledge graph embeddings with literals: Which model links better literal-ly? *Semantic Web* Preprint (2019), 1–31.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[17] Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. 2017. Knowledge Transfer for Out-of-Knowledge-Base Entities : A Graph Neural Network Approach. In *IJCAI*. ijcai.org, 1802–1808.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[19] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, et al. 2020. Knowledge graphs. *arXiv preprint arXiv:2003.02320* (2020).

[20] Michael Kampffmeyer, Yinbo Chen, Xiaodan Liang, Hao Wang, Yujia Zhang, and Eric P Xing. 2019. Rethinking knowledge graph propagation for zero-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11487–11496.

[21] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308* (2016).

[22] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. 2013. Attribute-based classification for zero-shot visual object categorization. *IEEE transactions on pattern analysis and machine intelligence* 36, 3 (2013), 453–465.

[23] Chung-Wei Lee, Wei Fang, Chih-Kuan Yeh, and Yu-Chiang Frank Wang. 2018. Multi-label zero-shot learning with structured knowledge graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1576–1585.

[24] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[25] Hatem Mousselly-Sergieh, Teresa Botschen, Iryna Gurevych, and Stefan Roth. 2018. A multimodal translation-based approach for knowledge graph representation learning. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*. 225–234.

[26] Nihal V Nayak and Stephen H Bach. 2020. Zero-Shot Learning with Common Sense Knowledge Graphs. *arXiv preprint arXiv:2006.10713* (2020).

[27] Yavor Nenov, Robert Piro, Boris Motik, Ian Horrocks, Zhe Wu, and Jay Banerjee. 2015. RDFox: A highly-scalable RDF store. In *International Semantic Web Conference*. Springer, 3–20.

[28] Maria-Elena Nilsback and Andrew Zisserman. 2008. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*. IEEE, 722–729.

[29] Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg S Corrado, and Jeffrey Dean. 2013. Zero-shot learning by convex combination of semantic embeddings. *arXiv preprint arXiv:1312.5650* (2013).

[30] Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. 2009. Zero-shot learning with semantic output codes. In *NeurIPS*.

[31] Pengda Qin, Xin Wang, Wenhu Chen, Chunyun Zhang, Weiran Xu, and William Yang Wang. 2020. Generative adversarial zero-shot relational learning for knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 8673–8680.

[32] Scott Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. 2016. Learning deep representations of fine-grained visual descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 49–58.

[33] Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. 2021. Knowledge graph embedding for link prediction: A comparative analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 15, 2 (2021), 1–49.

[34] Abhinaba Roy, Deepanway Ghosal, Erik Cambria, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. 2020. Improving Zero Shot Learning Baselines with Commonsense Knowledge. *arXiv preprint arXiv:2012.06236* (2020).

[35] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*. Springer, 593–607.

[36] Haseeb Shah, Johannes Villmow, Adrian Ulges, Ulrich Schwanecke, and Faisal Shafait. 2019. An open-world extension to knowledge graph completion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3044–3051.

[37] Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31.

[38] Komal K Teru, Etienne Denis, and William L Hamilton. 2019. Inductive Relation Prediction by Subgraph Reasoning. *arXiv* (2019), arXiv–1911.

[39] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.

[40] Wei Wang, Vincent W Zheng, Han Yu, and Chunyan Miao. 2019. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (2019), 1–37.

[41] Xiaolong Wang, Yufei Ye, and Abhinav Gupta. 2018. Zero-shot Recognition via Semantic Embeddings and Knowledge Graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6857–6866.

[42] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. 2010. Caltech-UCSD birds 200. (2010).

[43] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. 2018. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence* 41, 9 (2018), 2251–2265.

[44] Jingqing Zhang, Piyawat Lertvittayakumjorn, and Yike Guo. 2019. Integrating Semantic Knowledge to Tackle Zero-shot Text Classification. In *NAACL-HLT (1)*. Association for Computational Linguistics, 1031–1040.

[45] Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 3065–3072.