# Cat2Type: Wikipedia Category Embeddings for Entity Typing in Knowledge Graphs

Russa Biswas, Radina Sofronova, Harald Sack, Mehwish Alam
FIZ-Karlsruhe - Leibniz Institute for Information Infrastructure
Karlsruhe, Germany
firstname.lastname@fiz-karlsruhe.de

## ABSTRACT

The entity type information in Knowledge Graphs (KGs) such as DBpedia, Freebase, etc. is often incomplete due to automated generation. Entity Typing is the task of assigning or inferring the semantic type of an entity in a KG. This paper introduces an approach named *Cat2Type* which exploits the Wikipedia Categories to predict the missing entity types in a KG. This work extracts information from Wikipedia Category names and the Wikipedia Category graph which are the sources of rich semantic information about the entities. In Cat2Type, the characteristic features of the entities encapsulated in Wikipedia Category names are exploited using Neural Language Models. On the other hand, a Wikipedia Category graph is constructed to capture the connection between the categories. The Node level representations are learned by optimizing the neighbourhood information on the Wikipedia category graph. These representations are then used for entity type prediction via classification. The performance of Cat2Type is assessed on two real-world benchmark datasets DBpedia630k and FIGER. The experiments depict that Cat2Type obtained a significant improvement over state-of-the-art approaches.

## CCS CONCEPTS

• **Computing methodologies** → **Semantic networks**; **Information extraction**; **Neural networks**.

## KEYWORDS

Entity Type Prediction, Language Models, Node Embeddings, Wikipedia Categories

**ACM Reference Format:**
Russa Biswas, Radina Sofronova, Harald Sack, Mehwish Alam. 2021. Cat2Type: Wikipedia Category Embeddings for Entity Typing in Knowledge Graphs. In *Proceedings of the 11th Knowledge Capture Conference (K-CAP '21), December 2–3, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3460210.3493575

## 1 INTRODUCTION

Recent research has witnessed the automatic construction of Knowledge Graphs (KGs) from heterogeneous data sources such as text, images, etc. More specifically, cross-domain open KGs such as Freebase, DBpedia, etc. are either extracted automatically from structured data, generated using heuristics, or are human-curated[3, 7, 30]. These factors often lead to incomplete entity type information in KGs. For example, DBpedia version-2016-10 consists of 48 subclasses of *dbo:Person* and only 36.6% of the total number of entities belonging to *dbo:Person* are assigned to its subclasses. Also, 307,164 entities in the same DBpedia version are assigned to *owl:Thing*. Entity Typing is the task of assigning or inferring the semantic type of an entity in a KG and is an important step of KG construction and completion.

Previously, many entity typing approaches in KGs have been proposed which are based on heuristics [23] and or formulate entity typing as the classification problem [16, 17, 21, 33]. The state-of-the-art (SOTA) models predict entity types using different features of a KG such as the anchor text mentions in the textual entity descriptions, relations between the entities, entity names, and Wikipedia categories. However, this work focuses on entity type prediction model, named *Cat2Type*, by uncovering the underlying semantics of the Wikipedia categories.

Wikipedia categories provide a taxonomic organization of the knowledge underlying DBpedia, YAGO, etc. Due to their rich taxonomic structure as well as the textual information in their names, Wikipedia categories have been successfully used in tasks such as entity disambiguation [8], Named Entity Recognition in low-resource language [9], etc. Figure 1 shows an excerpt extracted from DBpedia including entities, their types, and Wikipedia Categories. In this example, *dbc:Musicians_from_ New_Jersey*[1] has type as well as the place of birth of the entities contained in that category, i.e., *dbr:Alan_Silvestri*. On the other hand, the entity *dbr:Robert_Zemeckis* has the *rdf:type dbo:Person* which is a coarse-grained type. The fine-grained type of the entity *film_director* is present in its Wikipedia Category *dbc:Science_fiction_film_directors*.

Various studies have been proposed which use the semantics underlying the Wikipedia categories for completing RDF data sources such as DBpedia. Existing approaches use graph-based algorithms [2] for deriving "is a" taxonomy from Wikipedia categories or rule-based algorithms [15, 25, 26] or association rule based approaches [1]. In contrast to existing approaches, the proposed model Cat2Type exploits different characteristic features of the Wikipedia categories, namely **(i)** the textual content in the Wikipedia category names by leveraging different Neural Language Model (NLMs) and **(ii)** the structural information exploiting the connectivity of the Wikipedia categories based on shared entities between them using network embedding model.


*K-CAP '21, December 2–3, 2021, Virtual Event, USA*

ACM ISBN 978-1-4503-8457-5/21/12…$15.00
https://doi.org/10.1145/3460210.3493575

[1] prefix dbc: <http://dbpedia.org/resource/Category>

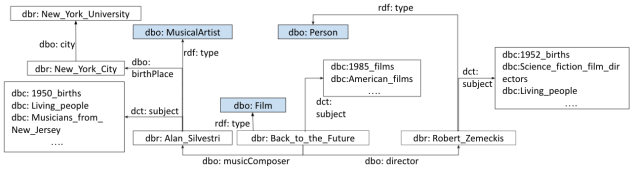



**Figure 1: Excerpt from DBpedia**

NLMs such as Word2vec [22], BERT [11], etc. have gained huge momentum in different NLP-based applications. These pretrained NLMs generate generalized representations of textual information without being trained on task-specific corpus and these representations can be used in various downstream tasks while retaining the performance [11, 27]. Therefore, this motivates in exploiting the NLMs on the Wikipedia category names for predicting types in the KGs. Furthermore, the reusability of the models also reduces the computational complexity of the proposed Cat2Type model. Moreover, this study includes the construction of the Wikipedia Category network in a novel way for capturing contextual information about the categories. Vector representations from these categories are then generated using random walk based node embedding algorithm. Finally, classification is performed using a fully connected neural network for entity type prediction on the top of the feature vectors generated from the aforementioned representations. Several experiments were performed to show the feasibility of the proposed approach on two benchmark datasets DBpedia630k and FIGER [33]. Cat2Type outperforms the state-of-the-art (SOTA) model HMGCN with an average improvement of 19.3% and 16.7% on $Ma - F_1$ and $Mi - F_1$ respectively on DBpedia630k dataset and 5.4% for $Mi - F_1$ on FIGER. Further experiments show that the proposed model also performs considerably well for unseen data. The main contributions of the paper are:

- A framework which leverages pretrained NLMs for learning the representations of entities for entity typing in KGs, and study its performance with different NLMs. The results provide strong evidence that entity representations based on pretrained language models exhibit strong generalization and are thus not limited to only NLP tasks.
- A novel category-category network has been constructed to leverage the underlying structure of the categories w.r.t. the shared entities between them. The results strengthen the fact that the category-category network is beneficial to predict types of unseen entities from a different KG.
- A generalized classification framework for both multi-label and multi-class classification is introduced, which can be easily deployed on any entity representations for entity type prediction for any KGs.

The rest of the paper is organized as: Section 2 gives an overview of the baseline approaches. Section 3 describes the proposed methodology, followed by experiments and results in Section 4. Finally, Section 5 provides the conclusion and an outlook of future work.

## 2 RELATED WORK

In this section, the SOTA models of entity typing are discussed categorized into two parts. The first part discusses the models exploiting the semantics underlying Wikipedia categories for entity typing whereas the second part focuses on the rest of the approaches which are heuristics-based or machine and deep learning-based.

In [15], the authors exploit the taxonomy, instances as well as lexicalization of Wikipedia categories for generating axioms for KG Completion (KGC). In [2], the authors propose a framework for deriving "is a" taxonomy from Wikipedia categories based on Directed Acyclic Graphs (DAG) using graph mining algorithms. Also, several rule-based approaches [10, 13, 25, 26] have been used in the past to predict entity types from the Wikipedia Categories.

Two models namely SDType [23], and [4, 5] do not consider the class hierarchy in KGs. SDType is a statistical heuristic model that exploits links between instances using weighted voting. In [5], different word embedding models, trained on triples, are used together with a classification model to predict the entity types which fail to capture the contextual information.

In CUTE [32], a hierarchical classification model which helps in cross-lingual Entity Typing by exploiting category, property, and property-value pairs. Another model [21] performs type prediction using the Scalable Local Classifier per Node (SLCN) algorithm using a set of incoming and outgoing relations. However, the entities with few relations are likely to be misclassified. MuLR [34] learns multi-level representations of entities via character, word, and entity embeddings followed by the hierarchical multi-label classification. FIGMENT [33] uses a global model and a context model. The global model predicts entity types based on the entity mentions from the corpus and the entity names. The context model calculates a score for each context of an entity and assigns it to a type. Therefore, it requires a large annotated corpora which is a drawback of the model. In APE [16], a partially labeled attribute entity-entity network is constructed containing structural, attribute and type information for entities followed by a deep neural network to learn the entity embeddings. MRGCN [31] is a multi-modal message-passing network that learns end-to-end from the structure of KGs as well as from multimodal node features. In HMGCN [17], the authors propose a GCN-based model to predict the entity types considering the relations, textual entity descriptions, and the Wikipedia categories.

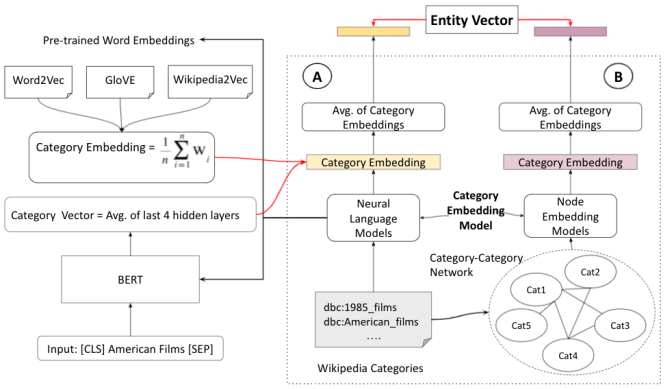
On the contrary to the aforementioned models, CatType encodes the Wikipedia Category information to exploit the respective semantic information using NLMs and node embeddings to predict the entity types.

## 3 CAT2TYPE FRAMEWORK

This section discusses the overall architecture of Cat2Type in detail which exploits the textual information in the Wikipedia Category Names as well as the structural information of Wikipedia Categories as illustrated by components Ⓐ and Ⓑ respectively in Figure 2.

**Preliminaries** A KG $G$ comprises of a set of types or classes denoted by $C$ and a set of entities $E$. `rdf:type` is an instance of `rdf:Property` that is used to state that a resource is an instance of a class. A triple of the form: $< e_i, \texttt{rdf:type}, C_k >$, states that $C_k \in C$, is an instance of `rdfs:Class` and $e_i \in E$ is an entity in $G$ and is an instance of $C_k$. Categories are used in Wikipedia to link

**Figure 2: Overall Architecture of the Cat2Type model**




articles under a common topic. There are 1,475,015 categories out of which 1,299,665 categories under the *Main topic classifications* which are considered for the entities similar to [15].

## Textual Information in Wikipedia Category Names

Wikipedia category names are comprised of rich semantic information about the characteristic features of the entities which have been used for KG completion tasks [15, 19]. In contrast to the previous work (cf. Section 2), in Cat2Type the textual information available in the Wikipedia category names are leveraged to predict the entity type information. Cat2Type focuses on uncovering the rich semantic information encoded in the Wikipedia category names (see Section 1 for the running example) using different NLM.

NLMs are neural network based models that learn the distributed representation of words into a continuous low-dimensional vector space. The semantically similar words appear closer to each other in the embedding space. These models can be further sub-divided into **(i)** non-contextual and **(ii)** contextual Embeddings [27]. The non-contextual word embeddings such as Word2Vec [22], GloVe [24], etc., are static in nature and are context-independent. These models are trained in an unsupervised manner on huge text corpora and the latent representations of the words capture their semantic meanings. However, they do not dynamically change according to the context the words appear in. However, contextual embeddings such as BERT [11], GPT [28], etc., encode the semantics of the words differently based on different contexts. Pre-training of the NLMs also helps in avoiding over-fitting of the model [27].

**Word2Vec & Glove** aims at learning the distributed representation of words in a large corpus to a low dimensional vector space. On the other hand, Glove exploits the global word-word co-occurrence statistics in the corpus with the underlying intuition that the ratios of word-word co-occurrence probabilities encode the meaning of the words.In Cat2Type, the Wikipedia category representations of the entities are generated from the Word2Vec model pre-trained on Google News dataset[2]. Given a category $C$ of the $i^{th}$ entity $C_{e_i}$ $(w_1, w_2, ..., w_n)$ represent the sequence of $n$ words in the category name, the category representation is given by

$$C_{e_i}^{W2V} = \frac{1}{n}\sum_{j=1}^{n} \mathbf{W}_j, \tag{1}$$

where $\mathbf{W}_j$ is the word embedding of the $j^{th}$ word in the category name extracted from the pre-trained Word2Vec model. For Glove, the Wikipedia category representations of the entities are generated from the GloVe model pre-trained on Wikipedia 2014 version and Gigaword 5[3]. Given a category $C_{e_i} = (w_1, w_2, ..., w_n)$, where $e_i$ is the $i^{th}$ entity and $w_1, w_2, .., w_n$ are the $n$ words in the category name, the category representation is given by

$$C_{e_i}^{GloVe} = \frac{1}{n}\sum_{j=1}^{n} \mathbf{W}_j, \tag{2}$$

where $\mathbf{W}_j$ is the word embedding of the $j^{th}$ word in the category name extracted from the pre-trained GloVe model.

**Wikipedia2Vec** [35] is a skip-gram based LM in which the entities and words from Wikipedia are jointly learned and optimized using three sub-models namely, Wikipedia link graph model, Word-based skip-gram model, and Anchor context model. Wikipedia link graph model is trained on an undirected entity-entity graph, in which an edge between two entities exists, if the Wikipedia page of one entity contains a link to that of the other entity or if both pages link to each other. It learns the entity embeddings by predicting neighboring entities. The word-based skip-gram model learns word embeddings by predicting neighboring words to a given word on a Wikipedia page. Lastly, the Anchor context model learns embeddings by predicting neighboring words given each entity such that the entity and its similar words appear closer to each other in the vector space. This model explicitly uses the co-occurrences of entities across Wikipedia putting similar entities closer. Also, Wikipedia categories comprise of words from Natural Language Text, therefore the embedding of words and entities together in the same vector space helps in generating better category representations. Cat2Type leverages the word vectors generated by the Wikipedia2Vec model to generate category representation of entities. Wikipedia2Vec model pre-trained on English Wikipedia 2018

[2] https://code.google.com/archive/p/word2vec/

[3] http://nlp.stanford.edu/data/glove.6B.zip

version[4] is used. Given a category $C_{e_i} = (w_1, w_2, ..., w_n)$, where $e_i$ is the $i^{th}$ entity and $w_1, w_2, .., w_n$ are the $n$ words in the Category name, the category representation is given by

$$C_{e_i}^{Wiki2Vec} = \frac{1}{n}\sum_{j=1}^{n} W_j, \tag{3}$$

where $W_j$ is the word embedding of the $j^{th}$ word in the category name extracted from the pre-trained Wikipedia2Vec model.

**BERT** [11] is a multi-layer bidirectional Transformer encoder for word representations which takes a sequence of words as an input and generates their vector representations. The word representations generated by BERT are sensitive to their respective context in which they appear in the natural language text. The pre-trained BERT model optimized on a huge amount of text allows the model to learn transferable and task-agnostic properties of language [12].

In this work, the entire category name has been used as an input which allows the BERT model to capture the semantics in the category names based on the sequence of words contained in those names. Given a category $C_{e_i} = (w_1, w_2, ..., w_n)$, where $e_i$ is the $i^{th}$ entity and $w_1, w_2, .., w_n$ are the $n$ words in the category name, special tokens $[CLS]$ and $[SEP]$ are added by the BERT encoder at the beginning and at the end of the category Name respectively. $[CLS]$ is a special classification token that marks the beginning of the input sequence and $SEP$ is used to mark the end of the sequence. In BERT, the decision is that the hidden state of the first token is taken to represent the whole sentence. The input sequence to the BERT model is given by, $([CLS], w_1, w_2, ..., w_n, [SEP])$. The output of the model is a sequence of contextualized embeddings of the tokens in the input sequence together with the added special tokens and is given by, $g(\hat{C_{e_i}}) = (h_{[CLS]}, h_1, h_2, ..., h_n, h_{[SEP]})$, where $h_i$ is the hidden representation of the $i^{th}$ token of the input sequence. For instance, the input sequence to the BERT model for the category *dbc: Science_fiction_film_directors* is *[CLS] Science fiction film directors [SEP].* The internal representations of words in BERT are a function of the entire input sentence and hence are called contextualized word representations [12]. Therefore, unlike the static word embedding models, the words in the category *dbc: Science_fiction_film_directors* are represented w.r.t to its context in BERT. Cat2Type exploits the feature-based approach of the BERT model in which fixed features are extracted from the BERT model, similar to [11, 20]. In this work, the MEAN pooling is considered as the representation of the input sequence which is the average of the $k$ hidden layers. It is observed to have outperformed the other pooling methods over the hidden layers for several tasks [20]. Therefore, the final category representation generated from the category names is given by,

$$C_{e_i}^{(BERT)} = \frac{1}{k}\sum_{j=1}^{k}(h^j_{[CLS]}, h^j_1, ...h^j_n, h^j_{[SEP]}), \tag{4}$$

where $(h^j_{[CLS]}, h^j_1, ...h^j_n, h^j_{[SEP]})$ is the representation of the $j^{th}$ hidden state. Cat2Type considers the average of last 4 hidden layers.

Other pre-trained transformer based contextual text embedding models [18] as well as static embedding models, can easily be used to generate the representation of the categories. The main advantages of using the pre-trained language models for entity typing are: **(i)** they are computationally inexpensive as the model is precomputed on huge training data. However, if required the training of the NLMs can be done once and reused for entity typing and other downstream tasks. **(ii)** Task-specific classification architecture can be easily deployed on top of this representation. **(iii)** The types of entities can be predicted for the entities only with the Wikipedia Category information available for long-tailed or less popular entities in a KG, i.e., the entities with less or no properties in a KG. **(iv)** Category Representations can be generated for the new Wikipedia categories without any training of the model.

[4] https://wikipedia2vec.github.io/wikipedia2vec

## Structural Features of Wikipedia Categories

Besides the Category Name, the connection between the Wikipedia categories w.r.t an entity in a KG provides meaningful semantic information for the task of entity typing. This section provides a description of the proposed category-category network followed by embedding models which learn the representations of categories by preserving the neighbourhood information of the nodes.

**Category - Category Network Construction.** To capture the semantic relatedness between the categories, a latent representation of the categories is learned. For this reason, an undirected homogeneous category-category network is constructed in this work, as shown in component (**B**) in Figure 2. Since, Wikipedia categories club together similar entities, therefore, categories sharing the same entities are similar to each other. The category-category network is constructed to exploit this connection between the categories based on common entities between them. It is given by $G_{cat} = (V, R)$, where $V$ is the set of nodes and $R$ is the set of edges . The nodes in the network are the Wikipedia categories and there exists an edge $R_k$ between two nodes $V_i$ and $V_j$, if these two categories share common entities. The categories which do not share any common entities with other categories are not considered in the graph. The weights of the edges between different nodes are crucial due to their significant impact on the embedding model. Therefore, the number of common entities between two categories is the weight of the edge between them.

The advantage of constructing the category-category network instead of using the available category taxonomy is that it captures the connection between two categories with a large number of shared entities between them but belonging to different branches in the taxonomy. However, if only taxonomy is considered, there exists no connection between these two categories. Therefore, random walk-based network embedding algorithms such as node2vec fails to capture the similarity between these two categories when trained on the category taxonomy tree, projecting these two categories far apart in their vector space. On the other hand, the semantic relatedness between the categories with shared entities is exploited. It helps in capturing the diversity of connectivity patterns observed in the category-category network. Furthermore, the problems with numeric values with NLMs [29] can be avoided with the category-category network as the Wikipedia Category names might contain numerical values.

**Node2Vec** [14] learns the latent representation of the nodes in a graph by preserving neighborhood information. Biased random

walks based on an efficient network-aware search strategy are generated from target nodes. In the category-category network $G_{cat}$, the selection of the next hop in the first order random walk is done based on the transition probability calculated by normalizing the edge weights and is given by,

$$P(u|v) = \frac{w_{uv}}{\sum_{u' \in N_v} w(u'v)} = \frac{w_{uv}}{d(v)}, \tag{5}$$

where $u, v \in V$ are the nodes, $N_v$ are the neighbouring nodes of $v$, $d(v)$ is the degree of node $v$, and $w(u, v)$ is the weight of the edge between the nodes $u$ and $v$. The second order transition applies a bias factor $\alpha$ to reweigh the edge weights depending on the previous state and the transition probability is given by,

$$P(u|v, t) = \frac{\alpha_{pq}(t, u)w(u, v)}{\sum_{u' \in N_v \alpha_{pq}}(t, u')w(u', v)}, \tag{6}$$

where $t, u, v \in V$, and $N_v$ are the neighbouring nodes of $v$. The random walk has already traversed the edge $(t, v)$ and the transition probability is calculated for the next node from starting from $v$. The bias factor $\alpha$ is given by,

$$\alpha_{pq}(t, u) = \begin{cases} \frac{1}{p} & d_{tu} = 0 \\ 1 & d_{tu} = 1 \\ \frac{1}{q} & d_{tu} = 2 \end{cases}, \tag{7}$$

where $d_{tu}$ determines the shortest distance between the nodes $t$ and $u$, parameter $p$ controls the likelihood of immediately revisiting a node in the walk, and $q$ allows the search to differentiate between inward and outward nodes. These random walks are treated as sentences in the skip-gram model to generate the node embeddings in the node2vec model. The main idea is to maximize the probability of predicting the correct context node given center node.

The second order transition in the random walks that stores the interconnections between the neighbors of every node [14]. Furthermore, the neighborhoods $N_v$ considered in the random walks are not only restricted to immediate neighbors but also extended to vastly different structures within the network depending on the sampling strategy. Therefore, the connectivity between the categories in the category-category network is exploited to its fullest in learning the latent representation of the categories.

## Entity Type Prediction

In this work, entity typing is considered a classification problem that takes the entity representation as an input to the classifier and predicts the corresponding type of the entity.

**Entity Representation** Previously explained approach generates the category representations from the category names and nodes in the category-category network. The entity representation from the category names feature is the average of the category representations generated by the respective language models LMs corresponding to that entity. Formally, it is given by,

$$E_i^{Model_{LM}} = \frac{1}{m} \sum_{j=1}^{m} C_j^{LM}, \tag{8}$$

where $E_i^{Model_{LM}}$ is the entity representation of the $i^{th}$ entity generated by $Model_{LM}$ and $C_j^{LM}$ represents the Category embedding of the $j^{th}$ category generated by the LMs in Equations 1, 2, 3 and, 4, and $m$ is the total number of categories associated with the entity.

Similarly, the average of all category representations generated by the node embedding models is taken as the latent representation of the entities consisting of multiple categories and is given by,

$$E_i^{Model_{NE}} = \frac{1}{m} \sum_{j=1}^{m} C_j^{NE}. \tag{9}$$

Here, $E_i^{Model_{NE}}$ is the entity representation of the $i^{th}$ entity generated from the node embedding model Node2Vec, $C_j^{NE}$ represents the Category embedding of the $j^{th}$ category of the corresponding entity, and the total number of categories associated with the entity is $m$. For each entity, separate entity representations are generated from each of the embedding models used in this work.

**Classifiers** In this work, both multi-class and multi-label classification have been used. For multi-class classification, a Fully Connected Neural Network (FCNN) consisting of two dense layers with ReLU as an activation function is deployed on the top of the entity representation. A softmax classifier with cross-entropy loss function is used in the last layer to calculate the probability of the entities belonging to different classes. Formally it is given by,

$$f(s)_i = \frac{e^{s_i}}{\sum_j^{C_T} e^{s_j}}, \tag{10}$$

$$CE_{loss} = -\sum_{i}^{C_T} t_i log(f(s)_i), \tag{11}$$

where $s_j$ are the scores inferred for each class in $C_T$ given in Equation 10. $t_i$ and $s_i$ in Equation 11 are the ground truth and the score for each class in C, respectively.

On the other hand, for multi-label classification, a similar FCNN with RELU as an activation function is used for the two dense layers. In multi-label classification, an element can belong to more than one class. Hence, an entity belonging to one class has no impact on the decision of its belonging to another class. Therefore, a sigmoid function with binary cross-entropy loss is used in the last layer which sets up a binary classification problem for each class in $C_T$. Therefore, the binary cross-entropy loss is,

$$CE_{loss} = -t_i log(f(s_i)) - (1 - t_i) log(1 - f(s_i)), \tag{12}$$

where $s_i$ and $t_i$ are the score and ground truth for $i^{th}$ class in $C_T$.

# 4 EXPERIMENTS AND RESULTS

This section gives details about the benchmark datasets, experimental setup, and analysis of the results obtained.

## Datasets

The two benchmark datasets, i.e., FIGER [33] and DBpedia630k [36] are used to evaluate the proposed model. FIGER dataset consists of 201,933 entities with 102 classes from Freebase and DBpedia630k which was originally constructed for text classification consists of 630,000 entities and 14 non-overlapping classes. Both FIGER and DBpedia630k datasets have been expanded in [17]. However, the extended datasets are not available publicly. Therefore, in this paper, these datasets are rebuilt by following the description provided in [17]. Furthermore, it is expanded with Wikipedia categories. One

**Table 1: Statistics of the datasets**

| Parameters | DB-1 | DB-2 | DB-3 | FIGER |
|---|---|---|---|---|
| #Entities | 210,000 | 210,000 | 210,000 | 201,933 |
| #Categories | 232,112 | 231,979 | 231,580 | 322,654 |
| #Entities train | 105,000 | 105,000 | 105,000 | 101,266 |
| #Entities test | 63,000 | 63,000 | 63,000 | 60,447 |
| #Entities validation | 42,000 | 42,000 | 42,000 | 40,220 |

of the contributions of this paper is to rebuild and extend both the datasets and make them publicly available for reusability purposes. In FIGER, out of 201,933 entities 199,111 have a corresponding DBpedia entity via the `owl:sameAs` relation. The entities of DBpedia630k are splitted equally into three parts DB-1, DB-2, and DB-3, each containing 210,000 entities as in [17]. Furthermore, each split is divided into a training set with 50%, test set with 30%, and validation set with 20% of the total entities in each split. It is to be noted that there are no shared entities between the train, test, and validation sets for all the DBpedia splits as well as for FIGER dataset. The statistics of the extended versions of both the datasets is provided in Table 1. The code and datasets are available online[5].

## Results

To evaluate the performance of *Cat2Type*, similar to the baseline models [16, 33], Micro-averaged $F_1(Mi - F_1)$ and Macro-averaged $F_1(Ma - F_1)$ metrics are used. Different variants of *Cat2Type* have been evaluated which serve as an ablation study.

**Experimental Setup** Following the explanation of the language models described in Section 3, the dimensions of the word vectors generated from the pre-trained Word2Vec and GloVe models are 300. Wikipedia2vec model used in Cat2Type is pre-trained on Wikipedia 2018 version with window size 10, epochs 10, negative sampling 15, and dimension 300 is used. BERT base figuration *bert-base-uncased*[6] which comprises of 12-layers, 768 hidden layers, 12 attention heads, and 110M parameters is used. Since the average of the hidden layers in BERT is considered as the output of the model in Cat2Type, therefore the dimension of the output vector is also 768. The node2vec model is trained on the category-category network with window size 10, length of the biased walk 10, number of walks per node 100, and 100 epochs. The dimension of the output entity vectors is 300. The FCNN classifier used in Cat2Type has batch size $\{32, 64\}$ and 100 epochs. The experiments with Cat2Type are performed on an Ubuntu 16.04.5 LTS system with 503GiB RAM with TITAN X (Pascal) GPU.

**Impact of NLMs on DBpedia splits** The Cat2Type-BERT variant of the proposed model outperforms all the baseline models with an average improvement of 19.3% and 16.7% on $Ma - F_1$ and $Mi - F_1$ respectively for all the DBpedia splits as illustrated in Table 2. The contextual embedding model BERT encodes semantics of the words differently based on different contexts. Therefore, the input to the model i.e., the entire Wikipedia category name, as discussed in Section 3 allowed the encoder to better capture the underlying semantics needed to learn more informative category representations. For DB1, the variations of Cat2Type with the other pre-trained NLMs, namely Word2Vec, GloVe, and Wikipedia outperforms all the baseline models for DB1 split and achieves comparable results with the baseline models for the other two splits. The intuitive reasoning behind this difference in performance of the non-contextual NLMs and BERT are: **(i)** the embeddings generated from non-contextual NLMs are static. The vector representation of the work is always the same regardless of its context. Therefore, these NLMs fail to model polysemous words, i.e., the words with multiple meaning. **(ii)** These static NLMs suffer from out-of vocabulary problem [27]. The NLM based Cat2Type variations are similar to MuLR and FIGMENT which consider word embeddings and entity embeddings as features. However, unlike FIGMENT, no annotated corpus is required in Cat2Type. Furthermore, Cat2Type does not require any additional information about the entities in the KG apart from the Wikipedia category names.

**Impact of Node Embeddings on DBpedia splits** The variation with Node2Vec model namely, *Cat2Type-node2vec* in Table 2 trained on the category-category network considerably outperforms all the baseline models. It shows an average improvement of 16% on $Ma - F_1$ and 13% on $Mi - F_1$ for all the splits as depicted in Table 2. Furthermore, node2vec achieves the second best result after BERT variants amongst the Cat2Type variants. It can be inferred that in Node2Vec the biased random walks efficiently explore diverse neighborhoods of a given node.

**Impact of NLMs on FIGER** The FIGER dataset comprises of entities belonging to more than one class, hence multi-label classification is used. The Cat2Type-BERT variant outperforms the SOTA model HMGCN with an improvement of 5.4% for $Mi - F_1$ and achieves comparable result with HMGCN for $Ma - F_1$ as shown in Table 2. Also, the performance of other variants with the NLMs for FIGER are low as compared to Cat2Type-BERT. It is observed that the misclassification of most of the entities occurs for the Freebase class *Person* and its subclasses. Most of the entities of class *Person* in FIGER are assigned to generic Wikipedia categories, such as, *dbr:Debbie_Millman* is assigned to the only category *dbc:Living_people* which does not provide any information about the fine-grained type of the entity. Also, it contains 15 subclasses of the class *Person* along with the class *Person* itself as fine-grained types of the corresponding entities. Therefore, with such generic categories, the entities are assigned to the coarse-grained type *Person* and is considered as misclassification. However, similar to the DBpedia splits, the contextual embedding model works better on FIGER as well. Therefore, it can be inferred that each of the feature vectors have a significant impact on the type prediction of the entities. However, the class distribution in FIGER is unbalanced with the largest class *City* containing 18,686 entities and the smallest class *engine_device* containing 14 entities and only 5 classes out of 102 classes in FIGER contain more than 11,000 entities and 70 classes have less than 1000 entities. This contributes to the comparatively lesser accuracy in the results with the FIGER dataset for all the Cat2Type variants compared to the DBpedia splits.

**Impact of Node Embeddings on FIGER** However, it is observed that node2vec approach with the category-category network works better than the static NLM approaches because node2vec captures the underlying semantics of the categories shared between the

[5] https://github.com/russabiswas/Cat2Type/ [6] https://huggingface.co/models

**Table 2: Results on DBpedia splits and FIGER**

| Models | DB1 | | DB2 | | DB3 | | FIGER | |
|---|---|---|---|---|---|---|---|---|
| | $Ma-F_1$ | $Mi-F_1$ | $Ma-F_1$ | $Mi-F_1$ | $Ma-F_1$ | $Mi-F_1$ | $Ma-F_1$ | $Mi-F_1$ |
| CUTE [32] | 0.679 | 0.702 | 0.681 | 0.713 | 0.685 | 0.717 | 0.743 | 0.782 |
| MuLR [34] | 0.748 | 0.771 | 0.757 | 0.784 | 0.752 | 0.775 | 0.776 | 0.812 |
| FIGMENT [33] | 0.740 | 0.766 | 0.738 | 0.765 | 0.745 | 0.769 | 0.785 | 0.819 |
| APE [16] | 0.758 | 0.784 | 0.761 | 0.785 | 0.760 | 0.782 | 0.722 | 0.756 |
| HMGCN [17] | 0.785 | 0.812 | 0.794 | 0.820 | 0.791 | 0.817 | **0.789** | 0.827 |
| Cat2Type-node2vec | 0.950 | 0.948 | 0.948 | 0.946 | 0.948 | 0.946 | 0.683 | 0.84 |
| Cat2Type-word2vec | 0.876 | 0.876 | 0.723 | 0.738 | 0.723 | 0.742 | 0.502 | 0.726 |
| Cat2Type-GloVE | 0.883 | 0.884 | 0.728 | 0.742 | 0.731 | 0.746 | 0.501 | 0.726 |
| Cat2Type-Wikipedia2Vec | 0.897 | 0.897 | 0.733 | 0.749 | 0.739 | 0.754 | 0.522 | 0.737 |
| Cat2Type-BERT | **0.983** | **0.984** | **0.983** | **0.983** | **0.985** | **0.985** | 0.764 | **0.881** |

**Table 3: Results on DBpedia splits on 7 classes**

| Models | DB1 | | DB2 | | DB3 | |
|---|---|---|---|---|---|---|
| | $Ma-F_1$ | $Mi-F_1$ | $Ma-F_1$ | $Mi-F_1$ | $Ma-F_1$ | $Mi-F_1$ |
| Cat2Type-node2vec | 0.972 | 0.973 | 0.971 | 0.972 | 0.969 | 0.971 |
| Cat2Type-word2vec | 0.917 | 0.931 | 0.797 | 0.812 | 0.797 | 0.813 |
| Cat2Type-GloVE | 0.938 | 0.942 | 0.803 | 0.816 | 0.801 | 0.814 |
| Cat2Type-Wikipedia2Vec | 0.955 | 0.953 | 0.813 | 0.822 | 0.812 | 0.822 |
| Cat2Type-BERT | **0.98** | **0.99** | **0.98** | **0.99** | **0.989** | **0.99** |

entities in the dataset as shown in Table 2. Also, the Cat2Type-node2vec approach for FIGER outperforms all the baseline models for the $Mi-F_1$ metric and performs second best.

**Impact on coarse-grained DBpedia Types** To analyze the impact of the model on coarse-grained entity types in a KG, the types in the DBpedia630k dataset are substituted with 7 coarse-grained types from DBpedia hierarchy. The types are *dbo: Organisation*, *dbo: Person*, *dbo: MeanOfTransportation*, *dbo: Place*, *dbo: Animal*, *dbo: Plant*, and *dbo: Work*. Here also the BERT variant yielding the best result and node2vec is the second best as depicted in Table 3. However, it is interesting to observe that the static NLMs show considerable improvement in their performances. This is due to the fact, the Wikipedia Category names often have meaning in a broader sense such as *dbc:Swiss_Jews*, which makes it difficult for the static embeddings to obtain fine-grained types of the entities. The performance of the node2vec model strengthens the fact that the underlying structural information in the $G_{cat}$ network provides rich semantic information for better entity representations.

Also to analyze the results on fine-grained types, the classes of DBpedia630k dataset are substituted with the subclasses resulting in 37 classes from the DBpedia hierarchy. The accuracy of the FCNN classifier on 37 classes, BERT yields the best results with 73.33%, 71.99%, and 91.59% whereas Node2Vec performs the second best with 69.06%, 67.06%, and 87.34% for DB1, DB2, and DB3 respectively.

**Impact of Node Embeddings on Unseen data** This work focuses on reusability of pre-trained models as it uses several pre-trained NLMs for predicting the types of the entities in a KG. However, the node2vec model has been trained on the category-category network. The reusability of the pre-trained node embeddings and the robustness of the classification model is analyzed on the *Movie dataset (mdgenre)* [6]. This dataset is a subset of Wikidata in the movie domain, consisting of the movies that are recorded as ever having won or been nominated for an award. The movies are the entities in the KG and their corresponding types are given by genres. The Wikipedia categories of the movies are extracted via Wikidata entities. The dataset comprises of 5996 entities (805 dev, 2347 test and 2844 train) and 12 classes. The pre-trained node2vec model trained on DBpedia is used to predict the types of the movies. The results depicted in Table 4 show that the Cat2Type-pretrained model outperforms the baseline models [6].

**Table 4: Results on Movie Dataset (Accuracy in %)**

| Models | mdgenre |
|---|---|
| RGCN | 63 |
| MRGCN | 62 |
| Cat2Type-node2vec | 66.47 |

**Table 5: Results on Unseen DBpedia entities (Accuracy in %)**

| Models | Unseen DBpedia entities |
|---|---|
| Cat2Type-Word2Vec | 98.64 |
| Cat2Type-GloVe | 98.26 |
| Cat2Type-Wikipedia2Vec | 98.76 |

The main advantages of learning the category representations using a category-category network are **(i)** for unseen entities, entity representations can be generated from their corresponding category embeddings to predict their types without training the whole model. **(ii)** KGs consist of many more number of entities than the Wikipedia categories. Therefore, the computational complexity of traversing a

complete graph with $\frac{n(n-1)}{2}$ edges where $n$ is the number of nodes is very high. Therefore, the category-category network avoids the computational complexity as well as implicitly encodes the entity information via the weights on its edges.

**Impact of Language Models on Unseen data** For further analysis, 307,164 unseen entities with `rdf:type` information as `owl:Thing` is extracted from DBpedia 2016-10 version, out of which 222,385 entities have `rdf:type` information in the current DBpedia version [7]. Amongst these 222,835 entities, 118,608 entities have a type belonging to the classes of DBpedia630k. The end to end Cat2Type model trained on DB1 with the static NLMs namely Word2Vec, GloVe, and Wikipedia2Vec are tested on 118,608 entities and the predicted types are compared against the types of the entities extracted from the current DBpedia version. The results depicted in Table 5 show that Cat2Type is robust and the type information of the entities can be inferred only from the Category Names of the entities. A major portion of the unseen entities from DBpedia comprises of entities from the class *dbo: Album*, whereas the other entities belong to the classes *dbo: Artist*, *dbo: Film*, *dbo: Athelete*, *dbo: Company*, and *dbo: Building*. Further analysis show that the test set without the entities from the class *dbo: Album*, also yields an average accuracy of 87%. Therefore, pre-trained classifier of the Cat2Type model can be efficiently used to predict types of new unseen entities in a KG.

## 5 CONCLUSION AND FUTURE WORK

This paper proposes a novel entity type prediction model, Cat2Type which exploits Wikipedia Categories from a KG and is evaluated on FIGER and DBpedia630k datasets, and their extensions are also published. A new category embedding method is introduced in which the category-category network is constructed based on the common entities between the categories. Also, non-contextual and contextual text embeddings are exploited for entity typing. Cat2Type model outperforms all the baselines for all the DBpedia splits and FIGER. Furthermore, the model performs well with unseen data. In the future, other contextual text embedding models will be exploited along with other features in the KG for entity typing.

## REFERENCES


[1] Mehwish Alam, Aleksey Buzmakov, Víctor Codocedo, and Amedeo Napoli. [n.d.]. Mining Definitions from RDF Annotations Using Formal Concept Analysis. In *Twenty-Fourth International Joint Conference on Artificial Intelligence 2015*.

[2] Mohamed Ben Aouicha, Mohamed Ali Hadj Taieb, and Malek Ezzeddine. 2016. Derivation of "is a" taxonomy from Wikipedia Category Graph. *Eng. Appl. Artif. Intell.* (2016).

[3] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*.

[4] Russa Biswas, Radina Sofronova, Mehwish Alam, Nicolas Heist, Heiko Paulheim, and Harald Sack. [n.d.]. Do Judge an Entity by Its Name! Entity Typing Using Language Models. In *ESWC 2021 P & D*.

[5] Russa Biswas, Radina Sofronova, Mehwish Alam, and Harald Sack. 2020. Entity Type Prediction in Knowledge Graphs using Embeddings. *arXiv* (2020).

[6] Peter Bloem, Xander Wilcke, Lucas van Berkel, and Victor de Boer. 2021. kg-bench: A Collection of Knowledge Graph Datasets for Evaluating Relational and Multimodal Machine Learning. In *European Semantic Web Conference*.

[7] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *ACM SIGMOD international conference on Management of data*.

[8] Silviu Cucerzan. 2007. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.


[7] https://downloads.dbpedia.org/repo/dbpedia/mappings/instance-types/2021.06.01/


[9] Arjun Das, Debasis Ganguly, and Utpal Garain. 2017. Named Entity Recognition with Word Embeddings and Wikipedia Categories for a Low-Resource Language. *ACM Trans. Asian Low Resour. Lang. Inf. Process.* (2017).

[10] Gerard De Melo and Gerhard Weikum. 2010. MENTA: Inducing multilingual taxonomies from Wikipedia. In *19th ACM international conference on Information and knowledge management*.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[12] Kawin Ethayarajh. 2019. How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings. In *Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

[13] Marco Fossati, Dimitris Kontokostas, and Jens Lehmann. 2015. Unsupervised learning of an extensive and usable taxonomy for DBpedia. In *11th International Conference on Semantic Systems*.

[14] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *22nd ACM SIGKDD international conference on Knowledge discovery and data mining*.

[15] Nicolas Heist and Heiko Paulheim. 2019. Uncovering the Semantics of Wikipedia Categories. In *18th International Semantic Web Conference*.

[16] Hailong Jin, Lei Hou, Juanzi Li, and Tiansi Dong. 2018. Attributed and Predictive Entity Embedding for Fine-Grained Entity Typing in Knowledge Bases. In *27th International Conference on Computational Linguistics*.

[17] Hailong Jin, Lei Hou, Juanzi Li, and Tiansi Dong. 2019. Fine-Grained Entity Typing via Hierarchical Multi Graph Convolutional Networks. In *Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

[18] Qi Liu, Matt J. Kusner, and Phil Blunsom. 2020. A Survey on Contextual Embeddings. *CoRR* (2020).

[19] Qiaoling Liu, Kaifeng Xu, Lei Zhang, Haofen Wang, Yong Yu, and Yue Pan. 2008. Catriple: Extracting triples from wikipedia categories. In *Asian Semantic Web Conference*.

[20] Xiaofei Ma, Zhiguo Wang, Patrick Ng, Ramesh Nallapati, and Bing Xiang. 2019. Universal text representation from bert: An empirical study. *arXiv preprint arXiv:1910.07973* (2019).

[21] A. Melo, H. Paulheim, and J. Völker. 2016. Type Prediction in RDF Knowledge Bases Using Hierarchical Multilabel Classification. In *WIMS*.

[22] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality.. In *NIPS*.

[23] H. Paulheim and C. Bizer. 2013. Type Inference on Noisy RDF Data. In *ISWC*.

[24] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Empirical methods in natural language processing*.

[25] Simone Paolo Ponzetto and Michael Strube. 2011. Taxonomy induction based on a collaboratively built knowledge repository. *Artificial Intelligence* (2011).

[26] Simone Paolo Ponzetto, Michael Strube, et al. 2007. Deriving a large scale taxonomy from Wikipedia. In *AAAI*.

[27] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences* (2020).

[28] Alec Radford and Karthik Narasimhan. 2018. Improving Language Understanding by Generative Pre-Training.

[29] Avijit Thawani, Jay Pujara, Pedro A Szekely, and Filip Ilievski. 2021. Representing Numbers in NLP: a Survey and a Vision. *arXiv preprint arXiv:2103.13136* (2021).

[30] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* (2014).

[31] WX Wilcke, P Bloem, V de Boer, RH van't Veer, and FAH van Harmelen. 2020. End-to-End Entity Classification on Multimodal Knowledge Graphs. *arXiv* (2020).

[32] Bo Xu, Yi Zhang, Jiaqing Liang, Yanghua Xiao, Seung-won Hwang, and Wei Wang. 2016. Cross-Lingual Type Inference. In *Database Systems for Advanced Applications - 21st International Conference, DASFAA*.

[33] Yadollah Yaghoobzadeh, Heike Adel, and Hinrich Schütze. 2018. Corpus-Level Fine-Grained Entity Typing. *J. Artif. Intell. Res.* (2018).

[34] Yadollah Yaghoobzadeh and Hinrich Schütze. 2017. Multi-level Representations for Fine-Grained Typing of Knowledge Base Entities. In *15th Conference of the European Chapter of the Association for Computational Linguistics*.

[35] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2018. Wikipedia2vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from wikipedia. *arXiv preprint arXiv:1812.06280* (2018).

[36] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*.