Preface (CSSA)

Since the beginning of the 2000s, there has been an increasing number of studies and standards proposed for generating large scale symbolic representations of knowledge (known as Knowledge Graphs (KGs)) out of heterogeneous resources such as text, images, etc. Moreover, there have been many advances in symbolic reasoning, as well as their applications to various fields. Recently, sub-symbolic methods have gained momentum. These methods aim at generating distributed representations from several resources such as text or symbolic representations (Graph Neural Networks, KG embeddings, etc.). These sub-symbolic methods for symbolic representations mainly focus on the task of KG completion. However, they have also recently been used for various tasks, e.g., in Natural Language Processing (NLP). The future perspective for these methods would be a combination of these approaches, leading to a form of neurosymbolic reasoning. Advances in the real world applications related to these methods will also serve as a stepping stone in the proving their practicality.

Overview (KGRL)

Knowledge Graphs are becoming the standard for storing, retrieving, and querying structured data. In academia and industry, they are increasingly used to provide background knowledge. Over the last years, several research contributions were made which show that machine learning, especially representation learning, can be successfully applied to knowledge graphs enabling inductive inference about facts with unknown truth values.
Brief Introduction

Several of these approaches encode the graph structure that can be used for tasks such as link prediction, node classification, entity resolution, recommendation, dialogue systems, and many more. Although proposed graph representations can capture the complex relational patterns over multiple hops, they are still insufficient to solve more complex tasks such as relational reasoning .For this kind of tasks, we envision a need for representations with more expressive power, which could include representation in non-Euclidean space. This starts by capturing e.g., type constrained, transitive or hierarchical relations in an embedding up to learning expressive knowledge representations languages like first-order logic rules.

Furthermore, most approaches for learning representations for knowledge graphs focus on transductive settings, i.e., all entities and relations need to be

seen during training, not allowing predictions for unseen elements. For evolving graphs, approaches are required that generalize to unseen entities and relations. One avenue of research to address inductiveness is to employ multimodal approaches that compensate for missing modalities, and recently meta-learning approaches have successfully been applied

Lately, the generalization of deep neural network models to non-Euclidean domains such as graphs and manifolds is explored They study the fundamental aspects that influence the underlying geometry of structured data for building graph representations Recent advances in graph representation learning led to novel approaches such as convolutional neural networks for graphs attention-based graph networketc. Most graphs here are either undirected or directed with both discrete and continuous node and edge attributes representing types of spatial or spectral data.

In this workshop, we want to see novel representation learning methods, approaches that can be applied to inductive learning and to (logical) reasoning and works that shed insights into the expressive power, interpretability, and generalization of graph representation learning methods.
Also, we want to bring together researchers from different disciplines but united by their adoption of earlier mentioned techniques from machine learning.

Ontology-based n-ball Concept Embeddings Informing Few-shot Image Classification

Mirantha Jayathilaka[0000−0002−2462−4833], Tingting Mu[0000−0001−6315−3432], and Uli Sattler[0000−0003−4103−3389]

Department of Computer Science, The University of Manchester, UK
{mirantha.jayathilaka,tingting.mu,uli.sattler}@manchester.ac.uk

Abstract. We propose a novel framework named ViOCE that integrates ontology-based background knowledge in the form of n-ball concept embeddings into a neural network based vision architecture. The approach consists of two components - converting symbolic knowledge of an ontology into continuous space by learning n-ball embeddings that capture properties of subsumption and disjointness, and guiding the training and inference of a vision model using the learnt embeddings. We evaluate ViOCE using the task of few-shot image classification, where it demonstrates superior performance on two standard benchmarks.

Keywords: Background Knowledge · Ontology · Machine Learning · Few-shot Learning.

1 Introduction

Ontologies can capture consistent, generalised and structured knowledge that can be used with reasoning tools [23] that ensure knowledge consistency together with the ability to infer new knowledge [1]. Sometimes knowledge graphs are also called as ontologies [30], but we identify clear differences. Knowledge graphs tend to be more loosely defined, whereas ontologies have a well-defined semantics that distinguish concepts from the given knowledge specification and other relationships (e.g., hasPart) between concepts bound by logical axioms. Sometimes a knowledge graph can be seen as a specific instantiation of a whole or part of an ontology representing only object-level information [15], whereas ontologies include both concept-level information and objects or terms. Moreover with powerful reasoning tools, ontologies facilitate the discovery of implicit knowledge from explicitly define knowledge. This study sheds light on the use of ontologies in a machine learning context. We use the Web Ontology Language (OWL) [18] in constructing our ontologies in this study. In order to assess the impact of knowledge integration to a visual recognition task, we chose few-shot image classification [2] to be the main task in this study. Few-shot learning in

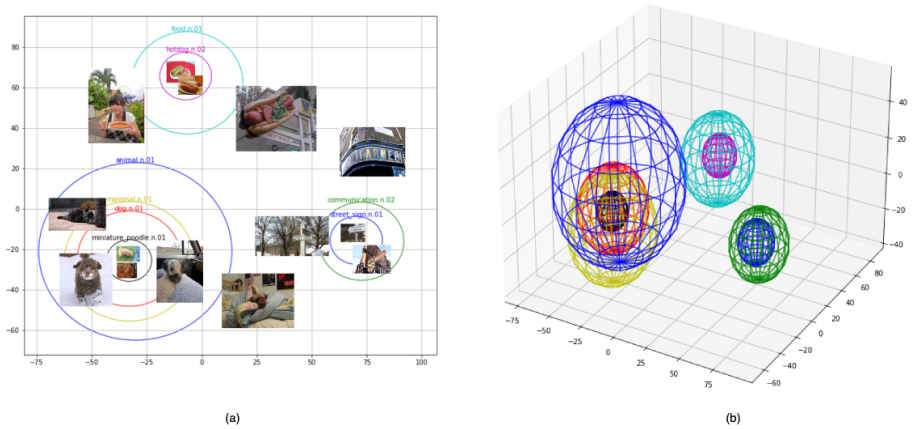

Fig. 1. The proposed approach classifies images by projecting them towards concept n-balls defined in high-dimensional space according to ontology-based background knowledge. (a) shows a snapshot of a few predictions for three miniImageNet[27] classes 'miniature poodle', 'hotdog' and 'street sign' made by a model trained using the ViOCE framework during few-shot image classification. The dimensionality of the n-balls is reduced to 2 for visualisation purposes. A correct prediction is an image projected to be inside of the n-ball of its ground truth label. Additionally, the surrounding n-balls to the ground truth n-balls, defined according to the background knowledge, gives us an unique opportunity to measure the 'certainty' of the model in classifying each image. For example, not all 'miniature poodle' images lie inside the ground truth n-ball but an image lying inside 'dog' can be identified as semantically meaningful. (b) is a visualisation of the same set of n-balls in (a) reduced to a 3-dimensional space in order to provide a clearer idea on the nature of n-ball shape and placing.

an image classification context focuses on effectively learning the visual features of a class with very few examples.

The proposed ViOCE framework, we adopt a technique to embed ontology-based knowledge as n-balls inspired by the work done by Kulmanov et al. [14]. This embedding can represent specialisations (e.g., Dog SubclassOf Animal) using the property of one n-ball enclosing another and partonomies (e.g., Dog hasPart Tail) using translations of n-ball positions. In this study, we directly utilise two loss design components of [14] to capture subsumption and disjointness axioms, while extending their approach with more regularisation components in order to embed large hierarchies in a favourable manner for a downstream vision task. Additionally, we propose the use of the inferred class hierarchy of the input ontology and introduce a technique to evaluate the quality of the learnt embeddings during the embedding learning process. The learnt n-ball embeddings can be seen as definitions of space for each concept in consideration that preserves the inferred class hierarchy entailed by the ontology. Next, we introduce a method to use a vision model [5; 12] to map input images to the space defined by the concept embeddings, informing the vision task with the knowl-

edge captured from the ontology. Figure 1 shows a snapshot of a few predictions for some miniImageNet[27] classes 'miniature poodle', 'hotdog' and 'street sign' made by a model trained using the ViOCE framework. We find that our approach facilitates better transparency on the behaviour of both knowledge embeddings and visual feature learning.

Overall, we extend [14] to capture knowledge from an ontology in the form of n-ball embeddings and show that they are favourable for the downstream vision task of few-shot image classification. This is also coupled with a technique to measure the quality of the learnt embeddings with respect to the knowledge entailed by the ontology. Next, we propose a technique to utilise the n-balls to guide a vision model during its training and inference stages performing few-shot image classification.

2 Related Work

An area that inspires the investigation of background knowledge integration in vision is the existing work done in knowledge-based vision systems [13; 25]. In [13], an interesting categorisation of knowledge that can be used as background knowledge is proposed, namely, permanent theoretical knowledge, circumstantial knowledge, subjective experimental knowledge and data knowledge. Although how these categories are formed is debatable, the importance of looking into different forms of knowledge that can be used as background knowledge is identified. The choice of knowledge form can be very much based on the considered vision application, as pointed out in [25], where the authors curate a number of vision tasks along with the forms of knowledge used to inform the learning process. Out of these, the use of scene graphs, probabilistic ontologies and first-order logic rules grab the attention as promising paths to explore. Investigations into the use of background knowledge in the form of first-Order Logic (FOL) is prominently seen in several studies [10]. As shown in [10], adaptation of logical knowledge as constraints during the learning process has generated promising results, that reinforces the attempts to use ontologies as background knowledge. The area of neuro-symbolic approaches also provides insights into the use of logical knowledge during the training of artificial neural networks [22].

In terms of combining other sources of knowledge [16] with computer vision, this study is motivated by work such as [5; 12] and [29], where image features are mapped to a vector space defined by language embeddings. This is identified as informing the image model with more knowledge that do not exist merely in the image features. In the case of [5], the knowledge from an unstructured text corpus is captured in the form of word embeddings to be integrated to the vision architecture. These approaches were mostly evaluated on zero-shot image classification, making use of the distance between points in the vector space defined. These findings motivate the proposed approach in this study, since they allow to extend standard vision models to incorporate language information. In terms of evaluation however, it can be argued that few-shot image classification [20] is a better candidate to measure how additional knowledge could help grasp

new concepts faster. In terms of few-shot learning [4; 11], our study is motivated by metric learning methods [28; 20] due of their ability to extend standard vision architectures [6]. These approaches exploits image feature similarities [24] when learning and predicting a vision task.

3 *n*-Balls and EL Embeddings

The mathematical concept of ball refers to the volume space bounded by a sphere and is also called a solid sphere. An n-ball usually refers to a ball in an n-dimensional Euclidean space. The EL embeddings study [14] attempts to encode logical axioms by positioning n-balls. We explain how it works for encoding subsumption and disjointness as they are the most relevant to our work. Each concept P is embedded as an n-ball with its centre denoted by $\boldsymbol{c}_P \in \mathbb{R}^n$ and the radius by $r_P \in \mathbb{R}$. The basic idea is to move one ball inside the other for subsumption and to push two balls to stay away for disjointness. The following loss is minimized to encode $\mathcal{O} \models P \sqsubseteq Q$:

$$\begin{aligned} &l_{P \sqsubseteq Q}(\boldsymbol{c}_P, \boldsymbol{c}_Q, r_P, r_Q) \\ &\quad = \max(0, \|\boldsymbol{c}_P - \boldsymbol{c}_Q\|_2 + r_P - r_Q - \gamma) \\ &\qquad + \big|\|\boldsymbol{c}_P\|_2 - 1\big| + \big|\|\boldsymbol{c}_Q\|_2 - 1\big|, \end{aligned} \tag{1}$$

where $\|\cdot\|_2$ denotes the l_2 norm and $\gamma \in \mathbb{R}$ is a user-set hyperparameter. It enforces the inequality $\|\boldsymbol{c}_P - \boldsymbol{c}_Q\|_2 \leq r_Q - r_P + \gamma$, meanwhile regulates the ball centers to be close to a unit sphere. Through controlling the sign of γ, the user can adjust whether to push the P ball completely inside the Q ball. In a similar fashion, the loss for encoding $\mathcal{O} \models P \sqcap Q \sqsubseteq \bot$ is given as

$$\begin{aligned} &l_{P \sqcap Q \sqsubseteq \bot}(\boldsymbol{c}_P, \boldsymbol{c}_Q, r_P, r_Q) \\ &\quad = \max(0, -\|\boldsymbol{c}_P - \boldsymbol{c}_Q\|_2 + r_P + r_Q + \gamma) \\ &\qquad + \big|\|\boldsymbol{c}_P\|_2 - 1\big| + \big|\|\boldsymbol{c}_Q\|_2 - 1\big|. \end{aligned} \tag{2}$$

It enforces the inequality $\|\boldsymbol{c}_P - \boldsymbol{c}_Q\|_2 \geq r_Q + r_P + \gamma$. According to the setting of γ, the user can decide how far the two balls are pushed away.

4 Proposed Method: ViOCE

We study how to effectively integrate ontology-based background knowledge to improve few-shot image classification. More specifically, this paper is focused on using additional hierarchical knowledge about the different classes to help image classification, achieving reduced data dependency of vision model architectures that are based on deep neural networks.

Adopting few-shot image classification as our benchmark [9], we train a neural vision model using a set of background images $BI = \{(\boldsymbol{I}_i, y_i)\}_{i=1}^{m}$ (base set) from $\mathcal{K}$ classes with $y_i \in C_B = \{c_1, c_2, \ldots c_{\mathcal{K}}\}$ and a set of few-shot images

$FI = \{(\boldsymbol{I}_i, y_i)\}_{i=1}^{s}$ (novel set) from w classes with $y_i \in C_F = \{\tilde{c}_1, \tilde{c}_2, \ldots, \tilde{c}_w\}$, where $C_B \cap C_F = \emptyset$, and $\boldsymbol{I}_i$ denotes the raw image vectors containing pixel values. The few-shot success is usually assessed by how accurate a model can select a correct class from the candidate class set C_F for a new image from the few-shot classes. This is often referred to as the w-way s-shot few-shot image classification. We construct an ontology $\mathcal{O}$ by using the class label information C_B and C_F, and also WordNet. It provides information on relationships that can exist among the class labels, containing knowledge regarding to "SubClassOf" and "DisjointClasses". These define the subsumption and disjointness axioms in the ontology.

We propose ViOCE as a framework to improve few-shot image classification by integrating information provided by $\mathcal{O}$, BI and FI. It is composed of two main components: (1) to embed classes in C_B and C_F as n-balls based on the constructed $\mathcal{O}$, (2) to embed images in the same Euclidean space as the n-balls with a suitable arrangement, and to infer the class for a query image based on its image embedding and the n-ball embeddings of the candidate classes. Figure 2 shows the general framework flow with an overview of all processes and data inputs.

4.1 Concept n-Ball Embeddings

We build upon the EL embedding technique [14] to learn a set of n-balls for all concepts $\widetilde{\mathcal{O}}$ in the ontology $\mathcal{O}$, which is referred to as a *concept embedding*. We extract subsumption and disjointness axioms to define the class hierarchy of the ontology $\mathcal{O}$. It has been noticed that the entailed transitive relations such as if *Poodle SubclassOf Dog* and *Dog SubclassOf Animal*, then *Poodle SubclassOf Animal* are usually not well reflected by the learned n-balls. To overcome this, we use the inferred class hierarchy (ICH). Assuming all the concepts are satisfiable with $\mathcal{O}$, the ICH is computed according to Equation 3. ICH contains all possible subsumption relations according to the definition of $\mathcal{O}$.

$$\text{ICH}(\mathcal{O}) = \{P \sqsubseteq Q | P \neq Q, P, Q \in \widetilde{\mathcal{O}}, \mathcal{O} \models P \sqsubseteq Q\}. \tag{3}$$

If simply to follow Eqs. (1) and (2), the radius of the learned n-ball for a leaf concept, which corresponds to an image class in C_B or C_F, can end up being very small, in order to fit into the balls of its ancestor concepts. Since in the image embedding learning, we will map each image as a data point inside the n-ball corresponding to its ground truth class, an overly small radius can affect the learning accuracy. To tackle this, we introduce a regularisation term in Eq. (5) to prevent radius shrinkage. Also, the embedding quality can deteriorate as the class hierarchy of the input ontology becomes larger. To improve the embedding quality, we introduce an extra hyperparameter in Eq. (4) to explore potentially more expressive design spaces, which is supported by an additional parameter

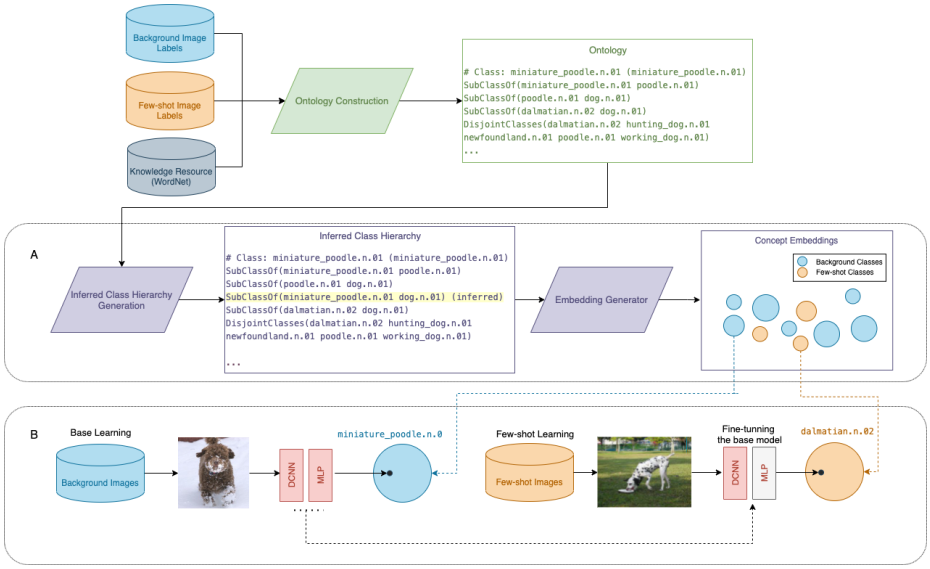

Fig. 2. The overview of the proposed ViOCE framework. If a suitable ontology for the task does not exist, the approach starts from constructing an ontology for the image labels capturing relationships between them based on an external knowledge resource (in our case WordNet). Subsequently the approach follows the two main components of the framework - A) Concept embedding learning process that starts with computing the inferred class hierarchy (ICH) of the input ontology and then generates n-ball embeddings for all the concepts found in the ontology. B) Visual model (DCNN+MLP) training where, first the background images are used to train a base model which gets fine-tuned (only MLP) using the few-shot images to produce the final model. During both base learning and few-shot learning processes, the concept embeddings guide the learning process by setting the objective of the model to project the image feature points inside the correct n-ball representing the ground truth label of an input image.

tuning process. Finally, we minimise the following loss function:

$$
\begin{aligned}
& l_c\left(\{\boldsymbol{c}_P\}_{P\in\widetilde{\mathcal{O}}}, \{r_P\}_{P\in\widetilde{\mathcal{O}}}\right) && (4)\\
&= \sum_{\mathrm{ICH}(\mathcal{O})\models P\sqsubseteq Q} \max(0, \|\boldsymbol{c}_P - \boldsymbol{c}_Q\|_2 + r_P - r_Q - \gamma)\\
&\quad + \sum_{\mathcal{O}\models C\sqcap D\sqsubseteq\bot} \max(0, -\|\boldsymbol{c}_P - \boldsymbol{c}_Q\|_2 + r_P + r_Q + \gamma)\\
&\quad + \sum_{P\in\widetilde{\mathcal{O}}} \max(0, \psi\sqrt{N_h - L(P)} - r_P) && (5)\\
&\quad + \sum_{P\in\widetilde{\mathcal{O}}} N(P)\big|\|\boldsymbol{c}_P\|_2 - \phi\big| && (6)
\end{aligned}
$$

Here, N_h denotes the total level number contained by the class hierarchy, and $L(P)$ denotes the level of the concept P in the hierarchy, e.g.,, the top-most concept has level 1. $N(P)$ denotes the number of times the concept P appears in the extracted axioms. Both $\psi, \phi > 0$ are hyperparameters. Eq. (5) restrict the radius of the concept P 's n-ball to be no less than $\psi\sqrt{N_h - L(P)}$. The top-level concepts are allowed to have larger n-balls than the bottom ones.

4.2 Hyperparameter Tuning of n-ball Embeddings

Three parameter tuning scores are proposed by examining whether $\|\boldsymbol{c}_P - \boldsymbol{c}_Q\| \leq r_Q - r_P$ holds for a ground truth subsumption $ICH(\mathcal{O}) \models P \sqsubseteq Q$. All the ground truth subsumptions are considered as positive instances. If the inequality holds, it is considered as a positive prediction. The classical F_1 score, which is the harmonic mean of the precision and recall, is used to assess the prediction accuracy of these subsumptions. We calculate two versions of F_1 score, one is referred to as $F_1^{(\text{all})}$ based on all the subsumptions extracted from $ICH(\mathcal{O})$. The other only considers the subsumptions involving the leaf concepts, which correspond to all the classes in C_B and C_F, as well as their direct parent classes. This score is referred to as $F_1^{(\text{leaf})}$. The third parameter tuning score S_D examines the disjointness between the leaf concepts. Enumerating all the pairs of leaf concepts, S_D is equal to the number of pairs for which the condition $\|\boldsymbol{c}_P - \boldsymbol{c}_Q\| \geq r_P + r_Q$ holds. A higher S_D indicates less overlapping between the n-balls of the leaf concepts. We need S_D to be greater that a threshold value of $\mathcal{T}$.

A good concept embedding result should have high $F_1^{(\text{all})}$, $F_1^{(\text{leaf})}$ and S_D scores. We compute these scores as a mandatory step at the end of each embedding learning process. The hyperparameters governing the scores are γ, ϕ and ψ. We use grid search to find the best combination of these parameters that would result in the best $F_1^{(\text{all})}$, $F_1^{(\text{leaf})}$ and S_D scores.

4.3 Image Embedding Learning

Our vision model is composed of a base DCNN architecture coupled with a multi-layer perceptron (MLP). The DCNN computes the visual features for an image by taking its raw pixel representation vector as the input: $\boldsymbol{f}_i = \phi_{\text{D}}(\boldsymbol{I}_i, \boldsymbol{\theta}_{\text{D}})$ where $\boldsymbol{f}_i \in \mathbb{R}^d$. The MLP is responsible for mapping the visual features $\boldsymbol{f}_i$ to the n-dimensional Euclidean space where the n-ball concept embeddings sit: $\boldsymbol{h}_i = \phi_{\text{M}}(\boldsymbol{f}_i, \boldsymbol{\theta}_{\text{M}})$ where $\boldsymbol{h}_i \in \mathbb{R}^n$. We use $\boldsymbol{\theta}_{\text{D}}$ and $\boldsymbol{\theta}_{\text{M}}$ to denote the neural network parameters to be trained for the DCNN and MLP, respectively. The idea is to identify visual features of an image (using a DCNN) so that they can be mapped (by an MLP) as a data point inside the n-ball of its ground truth class. For example, an image containing the visual features of a "poodle" should be mapped inside the n-ball of the "poodle" concept learnt from the ontology.

To achieve this, the following a pairwise ranking loss is used to optimise the network parameters:

$$\mathrm{l}_I(\boldsymbol{\theta}_{\text{D}}, \boldsymbol{\theta}_{\text{M}}) = \sum_{i=1}^{m} \left[\max\left(0, \|\boldsymbol{c}_P - \boldsymbol{h}_i\|_2 - \mu r_P\right) + \sum_{Q \in C_i^{(-)}} \max(0, \nu r_Q - \|\boldsymbol{c}_Q - \boldsymbol{h}_i\|_2) \right], \quad (7)$$

where $\mu, \nu > 0$ are hyperparameters. The set $C_i^{(-)}$ contains the negative classes defined for each image $\boldsymbol{I}_i$ of the positive class with its embedding computed by $\boldsymbol{h}_i = \phi_{\mathrm{M}}(\phi_{\mathrm{D}}(\boldsymbol{I}_i, \boldsymbol{\theta}_{\mathrm{D}}), \boldsymbol{\theta}_{\mathrm{M}})$. When setting $\mu = \nu = 1$, the loss enforces $\|\boldsymbol{c}_P - \boldsymbol{h}_i\|_2 \leq r_P$, pushing the embedded image point to stay inside the n-ball of the correct concept class P, while $\|\boldsymbol{c}_Q - \boldsymbol{h}_i\|_2 \geq r_Q$, to stay outside the n-ball of the incorrect concept class Q. The hyperparameters μ and ν are placed to control the intensity of this effect, e.g.,, $\mu < 1$ requiring to lie closer to the center which makes the task harder.

A specification crucial to learning performance is the selection of negatives concepts in $C_i^{(-)}$. Following the notion of "hard negatives" in [19], we select "hard negatives" for each positive concepts based on similarity. For example, the "poodle" concept is more similar to "golden retriever" in contrast to the "street sign", therefore it is more challenging to distinguish between "poodle" and "golden retriever". So we choose as the hard negatives the more similar concepts to a positive concept. Specifically, we evaluate similarities between concepts by Euclidean distances between the centre vectors of their corresponding n-balls, and perform k-means clustering based on these. After clustering the centre vectors of the leaf concepts (image classes), for each image class, all the other image classes from the same cluster as it are treated as the "hard negatives" and are included to $C_i^{(-)}$. In practice, we first train the DCNN and MLP from scratch by minimising Eq. (7) using the background images BI. This is called base learning (BL). Then, we fine tune the MLP by using the few-shot images FI by minimising the same loss, but keep the weights of DCNN fixed. This called the few-shot learning (FSL).

We test the vision model using the testing images of FI (FI_{te}) after the fine-tuning of MLP in the FSL stage. During inference, a prediction is made by finding the n-ball which an image feature projection lies in. Let $\mathcal{U} = \|\boldsymbol{c}_P - \boldsymbol{h}\| - r_P$, where $\boldsymbol{h}$ is an output feature for a query image from the vision model and $\boldsymbol{c}_P$ and r_P are the centre and radius of a selected n-ball of P respectively. If $\mathcal{U} \leq 0$, we find that $\boldsymbol{h}$ lies inside the n-ball of P. Hence the classification of $\boldsymbol{h}$ will be class P. In case some $\boldsymbol{h}$ does not lie inside any of the n-balls of the w classes in the few-shot task, we choose the closest lying n-ball centre $\boldsymbol{c}_i$ out of the classes to $\boldsymbol{h}$, where $\mathrm{argmin}_{\boldsymbol{c}_i (i=1,2,..,w)} \left(\|\boldsymbol{c}_i - \boldsymbol{h}\|\right)$, as the prediction. The proportion of the correct predictions out of all images in FI_{te} is recorded as the accuracy of the vision model in this study.

5 Experiment Setting

MiniImageNet dataset consists of 60,000 images of 100 classes from ImageNet where each class carries 600 example images [27]. Following the same splitting as in [8], 80 and 20 classes were allocated for training and testing respectively. TieredImageNet dataset is larger in size than miniImageNet, containing 608 classes from ImageNet [21]. Its classes are acquired based on 34 higher-level categories. We use a training set consisting of 26 higher-level categories with 448 classes, and testing set of 8 higher-level categories with 160 classes.

We construct two new ontologies based on the image labels of the datasets for each few-shot image classification benchmark. All selected datasets are subsets of ImageNet [3], where WordNet [17] synsets are used to annotate all images. This offered the opportunity to use the information from WordNet to formulate more knowledge about the image labels. We chose the hypernym tree of WordNet to be the source of the class hierarchy in this study, where given a label, the corresponding synset name together with all other synsets above it until the root (*entity.n.01*) was extracted. All these concepts were included in the ontology[1]. The dimensionality of the concept embeddings was chosen to be 300. During all experiments, ResNet50 [7] architecture was chosen to be the base network and the MLP was composed of 5 layers with sizes of 2048, 1024, 512, 512 and 300.

6 Results

6.1 Few-shot image classification results

ViOCE is evaluated by comparing with the performance of several existing approaches according to [26] under the same configuration. We conduct experiments for $w = \{5, 20\}$ and $s = \{1, 5\}$. Table 1 reports the 5-way 1-shot and 5-shot performance comparisons. It can be seen that ViOCE surpasses the the performance of all other approaches in every 5-way tasks with both datasets, while achieving >90% accuracy in miniImageNet 5-shot task.

Table 1. 5-way 1-shot and 5-shot accuracy comparison with existing approaches using miniImageNet and tieredImageNet benchmarks. All accuracies are reported with 95% confidence intervals.

Model	miniImageNet 5-way		tieredImageNet 5-way	
	1-shot (%)	**5-shot (%)**	**1-shot (%)**	**5-shot (%)**
MAML (Finn et al.)	48.70 ± 1.84	63.11 ± 0.92	51.67 ± 1.81	70.30 ± 1.75
Matching Networks (Vinyals et al.)	43.56 ± 0.84	55.31 ± 0.73	-	-
IMP (Allen et al.)	49.20 ± 0.70	64.7 ± 0.70	-	-
Prototypical Networks (Snell et al.)	49.42 ± 0.78	68.20 ± 0.66	53.31 ± 0.89	72.69 ± 0.74
Relational Networks (Sung et al.)	50.44 ± 0.82	65.32 ± 0.70	54.48 ± 0.93	71.32 ± 0.78
AdaResNet (Munkhdalai et al.)	56.88 ± 0.62	71.94 ± 0.57	-	-
TADAM (Oreshkin et al.)	58.50 ± 0.30	76.70 ± 0.30	-	-
Shot-Free (Ravichandran et al.)	59.04 ± n/a	77.64 ± n/a	63.52 ± n/a	82.59 ± n/a
MetaOptNet (Lee et al.)	62.64 ± 0.61	78.63 ± 0.46	65.99 ± 0.72	81.56 ± 0.53
Fine-tuning (Dhillon et al.)	57.73 ± 0.62	78.17 ± 0.49	66.58 ± 0.70	85.55 ± 0.48
LEO-trainval (Rusu et al.)	61.76 ± 0.08	77.59 ± 0.12	66.33 ± 0.05	81.44 ± 0.09
Embedding-distill (Tian et al.)	64.82 ± 0.60	82.14 ± 0.43	71.52 ± 0.69	86.03 ± 0.49
ViOCE	**65.71 ± 0.13**	**93.65 ± 0.07**	**73.4 ± 0.13**	**88.95 ± 0.09**

[1] The used ontologies can be accessed via `https://github.com/miranthajayatilake/ViOCE-Ontologies`

The study further extends the evaluation with the miniImageNet dataset to the task of 20-way 1-shot and 5-shot classification. In this case, considering all the 20 few-shot classes offers a bigger challenge to the model, having to distinguish between more classes with a few examples. Table 2 presents the result comparison on this task. ViOCE surpasses the performance of existing approaches in both 1-shot and 5-shot tasks with comfortable margins.

Table 2. 20-way 1-shot and 5-shot accuracy comparison with existing approaches using miniImageNet dataset.

Model	**miniImageNet 20-way**	
	1-shot (%)	**5-shot (%)**
MAML (Finn et al.)	16.49	19.29
Meta LSTM (Ravi et al.)	16.70	22.69
Matching Networks (Vinyals et al.)	17.31	26.06
Meta SGD (Li et al.)	17.56	28.92
Deep Comparison Network (Zhang et al.)	32.07	47.31
TIM-GD (Boudiaf et al.)	39.30	59.50
ViOCE	**48.02**	**84.13**

Another interesting observation during the BL stage of ViOCE was the behaviour of the training and testing accuracies of the vision model. With miniImageNet for example, the model was trained with 500 images per class across 80 classes, which is comparable to a standard image classification task. The training and testing accuracies were 85.32% and 95.36% respectively. The higher testing accuracy demonstrates the better generalisation ability of the learnt model. We argue that this effect is due to not forcing the image features to a fixed point as done in a standard training setting. The n-ball embeddings define a volume of space for each class providing more flexibility for the arrangement of image feature points.

7 Conclusion

We show that the introduction of ontology-based background knowledge to a visual model can improve its performance in the task of few-shot image classification. The proposed ViOCE framework is capable of utilising the n-ball concept embeddings in an effective way to inform the training and inference procedures of a vision model, and producing superior performance on two benchmarks. In future, we plan to extend this study to evaluate the semantically meaningful errors in classification and utilise multi-relational knowledge when learning concept embeddings.

Bibliography

[1] Alsubait, T., Parsia, B., Sattler, U.: Measuring similarity in ontologies: a new family of measures. In: International Conference on Knowledge Engineering and Knowledge Management. pp. 13–25. Springer (2014)

[2] Chen, W.Y., Liu, Y.C., Kira, Z., Wang, Y.C.F., Huang, J.B.: A closer look at few-shot classification. arXiv preprint arXiv:1904.04232 (2019)

[3] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: CVPR09 (2009)

[4] Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 1126–1135. JMLR. org (2017)

[5] Frome, A., Corrado, G., Shlens, J., Bengio, S., Dean, J., Ranzato, M., Mikolov, T.: Devise: A deep visual-semantic embedding model (2013)

[6] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

[7] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Jun 2016). https://doi.org/10.1109/cvpr.2016.90, `http://dx.doi.org/10.1109/cvpr.2016.90`

[8] He, X., Qiao, P., Dou, Y., Niu, X.: Spatial attention network for few-shot learning. In: International Conference on Artificial Neural Networks. pp. 567–578. Springer (2019)

[9] Hu, Y., Gripon, V., Pateux, S.: Leveraging the feature distribution in transfer-based few-shot learning. arXiv preprint arXiv:2006.03806 (2020)

[10] Hu, Z., Ma, X., Liu, Z., Hovy, E., Xing, E.: Harnessing deep neural networks with logic rules. arXiv preprint arXiv:1603.06318 (2016)

[11] Jayathilaka, M.: Enhancing generalization of first-order meta-learning (2019)

[12] Jayathilaka, M., Mu, T., Sattler, U.: Visual-semantic embedding model informed by structured knowledge. arXiv preprint arXiv:2009.10026 (2020)

[13] Ji, Q.: Combining knowledge with data for efficient and generalizable visual learning. Pattern Recognition Letters **124**, 31–38 (2019)

[14] Kulmanov, M., Liu-Wei, W., Yan, Y., Hoehndorf, R.: El embeddings: Geometric construction of models for the description logic el++. arXiv preprint arXiv:1902.10499 (2019)

[15] Marino, K., Salakhutdinov, R., Gupta, A.: The more you know: Using knowledge graphs for image classification. arXiv preprint arXiv:1612.04844 (2016)

[16] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)

[17] Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)

[18] Motik, B., Patel-Schneider, P.F., Parsia, B., Bock, C., Fokoue, A., Haase, P., Hoekstra, R., Horrocks, I., Ruttenberg, A., Sattler, U., et al.: Owl 2 web ontology language: Structural specification and functional-style syntax. W3C recommendation **27**(65), 159 (2009)
[19] Mu, T., Jiang, J., Wang, Y., Goulermas, J.Y.: Adaptive data embedding framework for multiclass classification. IEEE transactions on neural networks and learning systems **23**(8), 1291–1303 (2012)
[20] Qiao, S., Liu, C., Shen, W., Yuille, A.L.: Few-shot image recognition by predicting parameters from activations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7229–7238 (2018)
[21] Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J.B., Larochelle, H., Zemel, R.S.: Meta-learning for semi-supervised few-shot classification. arXiv preprint arXiv:1803.00676 (2018)
[22] Serafini, L., Garcez, A.d.: Logic tensor networks: Deep learning and logical reasoning from data and knowledge. arXiv preprint arXiv:1606.04422 (2016)
[23] Shearer, R., Motik, B., Horrocks, I.: Hermit: A highly-efficient owl reasoner. In: Owled. vol. 432, p. 91 (2008)
[24] Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. arXiv preprint arXiv:1703.05175 (2017)
[25] de Souza Alves, T., de Oliveira, C.S., Sanin, C., Szczerbicki, E.: From knowledge based vision systems to cognitive vision systems: a review. Procedia Computer Science **126**, 1855–1864 (2018)
[26] Tian, Y., Wang, Y., Krishnan, D., Tenenbaum, J.B., Isola, P.: Rethinking few-shot image classification: a good embedding is all you need? arXiv preprint arXiv:2003.11539 (2020)
[27] Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. arXiv preprint arXiv:1606.04080 (2016)
[28] Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: Advances in neural information processing systems. pp. 3630–3638 (2016)
[29] Wang, X., Ye, Y., Gupta, A.: Zero-shot recognition via semantic embeddings and knowledge graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6857–6866 (2018)
[30] Wickramarachchi, R., Henson, C., Sheth, A.: An evaluation of knowledge graph embeddings for autonomous driving data: Experience and practice. arXiv preprint arXiv:2003.00344 (2020)

ContextBERT: Contextual Graph Representation Learning in Text Disambiguation

Mozhgan Saeidi
orcidID0000-0002-1736-0737

mozhgan.saeidi@dal.ca

Abstract. Word representations derived by neural language models have been shown to effectively carry useful semantic information to improve the final results of various Natural Language Processing tasks. The information provided by these representations encodes the subtle distinction that might occur between different meanings of the same word. However, these representations do not include the input text's information, as the context, and a semantic knowledge base network. This integration of context and semantic network is helpful in NLP tasks, specifically in the lexical ambiguity problem. In this paper, we first analyzed the defects of current state-of-the-art representations learning approaches, and second, we present a word representation learning method, named ContextBERT, that is aware of the semantic knowledge base network and the context. ContextBERT is a novel approach to producing sense embeddings for the lexical meanings within a lexical knowledge base, using pre-trained BERT model The novel difference in our representation is the integration of the knowledge base information and the input text. Our representations enable a simple 1-Nearest-Neighbour algorithm to perform state-of-the-art models in the English Word Sense Disambiguation task.

Keywords: Sense Embedding · Representation Learning · Word Sense Disambiguation · Pre-trained Language Models · Semantic Networks.

1 Introduction

Text disambiguation is one of many problems in Natural Language Processing (NLP) tasks. In this task, we have an input text including a word with multiple possible meanings based on a semantic knowledge base network, and the question is which one of those multiple meanings is the best meaning match for the word in the text, based on its context [17,32]. The context here refers to the input document text. The text disambiguation task is mostly referred to as Word Sense Disambiguation (WSD) task in NLP. Knowledge bases are different in nature [2]; for example, WordNet is a lexical graph database of semantic relations (e.g., synonyms, hyponyms, and meronyms) between words. Synonyms are grouped into synsets with short definitions and usage examples. WordNet can thus be seen as a combination and extension of a dictionary and thesaurus [3]. Wikipedia is a hyperlink-based graph between encyclopedia entries[1].

The text ambiguity task is easy for humans by considering the context. The context enables us to identify the correct meaning of the ambiguous words. In computational methods, we try to enhance the algorithms to mimic this approach. These methods often represent their output by linking each word occurrence to an explicit representation of the chosen sense [37]. Two approaches to tackle this problem are the machine learning-based approach and the knowledge-based approach. In the machine learning-based approach, systems are trained to perform the task [32]. The knowledge-based approach requires external lexical resources such as Wikipedia, WordNet [13], a dictionary, or a thesaurus. The machine learning-based approaches mainly focus on achieving maximal precision or recall and have their drawbacks of run-time and space requirement at the time of classifier training [4]. So, knowledge-based disambiguation methods still have advantages to study. Among different knowledge-based methods, coherence-based has been more effective in explaining it. In the coherence-based approach, one important factor is the coherence of the whole text after disambiguation, while in other approaches, this factor might change to considering the coherence of each sentence or paragraph.

There are different factors that play important roles in solving the WSD problem, including word representation. Word representations have been shown to play an important role in different Natural Language Processing (NLP) tasks, especially in disambiguation tasks. There are many different approaches to generate word representation embeddings. Recently, embeddings based on pre-trained deep language models have attracted much interest. These models have proved to be superior to classical embeddings for several NLP tasks, including Word Sense Disambiguation (WSD). Some of most used models in this category are including ELMO [22], BERT [5], and XLNET [38]. these models encode several pieces of linguistic information in their word representations. These representations differ from static neural word embeddings [21] in that they are dependent on the surrounding context of the word [29].This difference makes these vector representations especially interesting for disambiguation, where effective contextual representations can be highly beneficial for resolving lexical ambiguity. In addition, these representations enabled sense-annotated corpora to be exploited more efficiently [10].

In this study, next section, we overview different current approaches for text embedding with focusing on the contextualized word representation. We analyzed the effectiveness of these methods on different types of words. We show the pros and cons of these state-of-the-art models in word representation learning on parts of speeches are. In our representation, we enhanced this detected defectiveness to improve representations. Our novel contribution provides a new representation of words using the context of the input text and the context of the knowledge base and uses the nearest neighbor heuristic algorithm to disambiguate ambiguous words. We finally compare the performance of our proposed approach with our representations with the most current methods in the disambiguation task.

2 Related Work

The Word Sense Disambiguation is one core problem in NLP, which addresses the ambiguity of words in a given context. In this task, we have access to two main sources of information to disambiguate the ambiguous words. One source is a semantic network,

and the other is sense-annotated corpora. Semantic networks encode a more general knowledge that is not tied to a specific task, and the information enclosed therein is usually employed for WSD by knowledge-based approaches. Instead, sense annotated corpora are tailored to the WSD task and are typically used as training sets for supervised systems. Therefore, we divide the WSD approaches into two categories of knowledge-based and supervised approaches [17].

2.1 Knowledge-Based Approaches

In the knowledge-based methods, the semantic network structure of the knowledge base is used, e.g., Wikipedia [7], WordNet [13], BabelNet [19], to find the correct meaning based on its context for each input word [16]. These approaches employ algorithms on graphs to address the word ambiguity in texts [1]. Disambiguation based on Wikipedia has been demonstrated to be comparable in terms of coverage to domain-specific ontology [36] since it has broad coverage, with documents about entities in a variety of domains [31]. The most widely used lexical knowledge base is WordNet, although it is restricted to the English lexicon, limiting its usefulness to other vocabularies. BabelNet solves this challenge by combining lexical and semantic information from various sources in numerous languages, allowing knowledge-based approaches to scale across all languages it supports. Despite their potential to scale across languages, knowledge-based techniques on English fall short of supervised systems in terms of accuracy.

2.2 Supervised Approaches

The supervised approaches surpass the knowledge-based ones in all English data sets. These approaches use neural architectures, or SVM models, while still suffering from the need of creating large manually-curated corpora, which reduces their usability to scale over unseen words [20]. Automatic data augmentation approaches [33] developed methods to cover more words, senses, and languages.

In recent years, the contextual representation learning approaches have improved the performance of WSD models, where they have been employed for the creation of sense embeddings. Most NLP tasks now use semantic representations derived from language models. There are static word embeddings and contextual embeddings. This section covers aspects of the word and contextual embeddings that are especially important to our work.

Static Word Embeddings Word embeddings are distributional semantic representations usually with one of two goals: predict context words given a target word (Skip-Gram), or the inverse (CBOW) [12]. In both, the target word is at the center, and the context is considered as a fixed-length window that slides over tokenized text. These models produce dense word representations. One limit for word embeddings, as mentioned before, is meaning conflict around word types. This limitation affects the capability of these word embeddings for the ones that are sensitive to their context [28].

Contextual Word Embeddings The problem mentioned as a limitation for the static word embeddings is solved in this type of embeddings. The critical difference is that the contextual embeddings are sensitive to the context. It allows the same word types to have different representations according to their context. The first work in contextual embeddings is ELMO [22], which is followed by BERT [5], as the state-of-the-art model. The critical feature of BERT, which makes it different, is the quality of its representations [30]. Its results are task-specific fine-tuning of pre-trained neural language models. The recent representations which we analyze their effectiveness are based on these two models [24,23].

In our representation, we use different resources to build the vectors. In this section, we provide information on these resources.

2.3 Wikipedia

is the largest electronic encyclopedia freely available on the Web. Wikipedia organized its information via articles called Wikipedia pages. Disambiguation based on Wikipedia has been demonstrated to be comparable in terms of coverage to domain-specific ontology [36] since it has broad coverage with documents about entities in a variety of domains [11]. Moreover, Wikipedia has unique advantages over the majority of other knowledge bases, which include [40]:

- The text in Wikipedia is primarily factual and available in a variety of languages.
- Articles in Wikipedia can be directly linked to the entities they describe in other knowledge bases.
- Mentions of entities in Wikipedia articles often provide a link to the relevant Wikipedia pages, thus providing labeled examples of entity mentions and associated anchor texts in various contexts, which could be used for supervised learning in WSD with Wikipedia as the knowledge base.

2.4 BabelNet

is a multilingual semantic network, which comprises information coming from heterogeneous resources, such as WordNet, and Wikipedia [19]. It is organized into synsets, i.e., sets of synonyms that express a single concept, which, in their turn, are connected to each other by different types of relationships. One of Babelnet's features which is useful for our representation is *hypernym-hyponym* relations. In this relation, each concept is connected to other concepts via hypernym relation (for generalization) and via hyponym relation (for specification). *Semantically-related* relation is the other feature that we use that expresses a general notation of relatedness between concepts. The last feature of Babelent used in this work is *mapping to Wikipedia*, which maps its concepts to Wikipedia pages.

2.5 WordNet

is the most widely used lexical knowledge repository for English. It can be seen as a graph, with nodes representing concepts (synsets) and edges representing semantic relationships between them. Each synset has a set of synonyms, such as the lemmas spring, fountain, and natural spring in the synset, A natural flow of groundwater.

2.6 SemCor

is the typical manually-curated corpus for WSD, with about 220K words tagged with 25K distinct WordNet meanings, resulting in annotated contexts for around 15% of WordNet synsets.

2.7 BERT

is a Transformer-based language model for learning contextual representations of words in a text. The contextualized representation of BERT is the key factor that has changed the performance in many NLP tasks, such as text ambiguity. In our representations, we use BERT-base-cased to generate the vectors of each sense [5].

2.8 SBERT

is a modification of the pre-trained BERT network that uses siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine-similarity. We use this sentence representation when generating the vector representations of sense sentences, both in the input text and in the knowledge base text.

2.9 Graph Convolutional Network

Graph Convolutional Networks (GCN) is a very powerful multilayer neural network architecture for machine learning on graphs [8]. GCN operates directly on a graph and induces embedding vectors of nodes based on the properties of their neighborhoods. In fact, they are so powerful that even a randomly initiated $2-$layer GCN can produce useful feature representations of nodes in networks[2]. Formally, consider a graph $G = (V, E)$, where $V(|V| = n)$ and E are sets of nodes and edges, respectively. Every node is assumed to be connected to itself, i.e., $(v, v) \in E$ for any v which the reason for this assumption is mentioned at the end of this paragraph. Let $X \in R^{n\times m}$ be a matrix containing all n nodes with their features, where m is the dimension of the feature vectors, each row $x_v \in R_m$ is the feature vector for v. We introduce an adjacency matrix A of G and its degree matrix D, where $D_{ii} = \sum_j A_{ij}$. Because of self-loops, the diagonal elements of A are all 1. We now have a graph, its adjacency matrix A, and a set of input feature X. After applying the propagation rule $f(X, A) = AX$ and $X = I$, the representation of each node (each row) is now a sum of its neighbor's features. In other words, the graph convolutional layer represents each node as an aggregate of its neighborhood. The reason for considering the self-loops in the graph is the aggregated representation of a node to include its own features.

For a one-layer GCN, the new k-dimensional node feature matrix $L^{(1)} \in R^{n\times k}$ is computed as:

$$L^{(1)} = \rho(\hat{A}XW_0) \tag{1}$$

[2] The notation we used for GCN in this paper are the same as notations in [39]

where $\hat{A}$ is $D^{-0.5}AD^{-0.5}$, the normalized symmetric adjacency matrix and $W_0 \in R^{m \times k}$ is the weight matrix. The ρ is the activation function (RELU); $\rho(x) = max(0, x)$. GCN can capture information only about immediate neighbors with one layer of convolution. When multiple GCN layers are stacked, information about larger neighborhoods are integrated;

$$L^{(j+1)} = \rho(\hat{A}L^{j}W_j) \tag{2}$$

which j is the layer number and $L^0 = X$. In other words, the size of the second dimension of the weight matrix determines the number of features at the next layer. The feature representations can be normalized by node degree by transforming the adjacency matrix A by multiplying it with the inverse degree matrix D. First we used the simple propagation rule $f(X, A) = D^{-1}AX$, while then improved it. The improved version is inspired by a recent work [8] that proposes a fast approximate spectral graph convolutions using a spectral propagation rule $f(X, A) = \sigma(D^{-0.5}\hat{A}D^{-0.5}XW)$. They showed this property is very useful, that connected nodes tend to be similar (e.g. have the same label).

3 Methodology

This section presents our novel embedding approach of creating sense representations of BabelNet senses. Our representation learning is created by combining semantic and textual information from the first paragraph of each sense's Wikipedia page and the input document paragraph, which includes the ambiguous word. our approach uses the representation power of neural language models, i.e., BERT and SBERT. We divide our approach into the following steps:

3.1 Context Retrieval

In this step, we collect suitable contextual information from Wikipedia for each given concept in the semantic network. Similar to [34], we exploit the mapping between synsets and Wikipedia pages available in BabelNet, as well as its taxonomic structure, to collect textual information that is relevant to a target synset s. For each synset s, we collect all the connected concepts to s through hyponym and hypernym connections of the BabelNet knowledge base. We show this set of related synsets to s by R_s which is:

$$R_s = \{s^{'} | (s, s^{'}) \in E\}$$

Similar to [34], we use E as the set includes all hyponyms and hypernyms connections. In this work, for each page p_s, we consider the first opening paragraph of the page and compute its lexical vector by summing the SBERT vector representation of the sentences in this first paragraph. These lexical representations are later used for the similarity score finding between p_s and $p_{s'}$, for each $s^{'} \in R_s$ by using the weighted overlap measure from [25], which is defined as follows:

$$WO(p_1, p_2) = (\sum_{w \in O} \frac{1}{r_w^{p_1} + r_w^{p_2}})(\sum_{i=1}^{|O|} \frac{1}{2i})^{-1}$$

where O is the set of overlapping dimensions of p_1 and p_2 and $r_w^{p_i}$ is the rank of the word w in the lexical vector of p_i. We preferred the weighted overlap over the more common cosine similarity as it has proven to perform better when comparing sparse vector representations [25]. Similar to [34], Once we have scored all the $(p_s, p_{s'})$ pairs, we create partitions of R_s, each comprising all the senses s' connected to s with the same relation r, where r can be one among: hypernymy, and hyponymy. We then retain from each partition only the top-k scored senses according to $WO(p_s, p_{s'_i})$, which we set $k = 15$ in our experiments.

3.2 Word Embedding

In the second step, we use BERT for the representation of the given concepts from the input text. For each ambiguous word–which we call this word by mention– of the input, we extract the BERT representation of the mention. Using the BabelNet relations of hyponymy and hypernymy, we extract all synsets of mention from BabelNet (set E). For each one of these senses, use the link structure of BabelNet and Wikipedia; we collect all the Wikipedia pages for each sense. We use BERT representation for the second time to generate vector representation for senses. In the settings, each word is represented as a 300-dimensional vector, as the BERT dimension.

3.3 Sense Embedding

In this step, we build the final representation of each concept. From the previous step, we took the representation of mention, $R(m)$, and the representation of each one of its senses. We show the representations of each k sense of m by $R(s_i)$ which i varies from 1 to k, based on their similarity scores. Our unique representations combine the mention representation with sense representation, averaging the vector representations of $R(m)$ and $R(s_i)$. If mention m has k senses, our model generates k different representations of $R(m, s_1)$, $R(m, s_2)$, ..., $R(m, s_k)$. Since the dimension representation of $R(m)$ and each $R(s_i)$ is 300, these averaged representation dimensions are 300. Next novelty in our representations is ranking the k senses of each mention based on their relevancy degree to the context. To this aim, we average the representations of the first step. In the first step, we took the representation of the input text paragraph, which contains the ambiguous mention, show it by $R(PD)$ which stands for representation of the **P**aragraph of the input **D**ocument. In the first step, we also took the representation of the first paragraph of the Wikipedia page, which represents it by $R(PW)$, which stands for representation of the first **P**aragraph of the **W**ikipedia page. Finally, we average these two representations as $R(PD, PW)$. In the $R(PD, PW)$, the context is constant for each sense since the input text as the context is constant for each possible sense of the ambiguous words. The dimension of this averaged representation is also equal to the word representation, so it makes it possible to calculate their cosine similarities. To rank the senses most related to the context, we use the cosine similarity as follows:

$$\text{Sim}(m, s_i) = Cosine(R(m, s_i), R(PD, PW)) \text{ , for i=1 , ... , k}$$

This ranking provides the most similar sense to the context for each mention. This novelty makes this representation more effective than the previous contextualized-based

embeddings, especially in the task of sense disambiguation. At the end of these three steps, each sense is associated with a vector that encodes both the contextual information and knowledge base semantic information from the extracted context of Wikipedia and its gloss.

We consider each mention of the document as one node of the graph, and a newly added node (redirect link) will connect with its nearest neighbor by using cosine similarity, which makes the edges of the graph. The cosine similarity between two nodes on the edges makes the weight matrix. The number of nodes in the text graph $|V|$ is the number of mentions. For each sense s, we use an integrated representation of its mention m with its own representation, i.e,. $R(m, s)$. We set the feature matrix X as extracted representation of BERT as input to GCN. The dimension of the feature matrix here is 300, as it is the averaged representation length of two BERT embeddings, one for the mention and the other for the sense.

As mentioned, formally, the weights of edge between node i and node j defines as:

$$W_{ij} = \text{cosine sim}(R(i), R(j)) = \frac{R(i).R(j)}{||R(i)||||R(j)||} \tag{3}$$

which $R(i)$ is our representation of node i.

After building the graph, we feed it into a simple $2-$layers GCN as [8], the second layer node (mention,sense) embeddings are fed into a softmax classifier:

$$Z = softmax(\hat{A}RELU(\hat{A}XW_0)W_1) \tag{4}$$

where

$$\hat{A} = D^{-0.5}AD^{-0.5}$$

and

$$softmax(x_i) = \frac{1}{Z}exp(x_i)$$

with $S = \sum_i exp(x_i)$. The loss function is the one defined in [39] as:

$$L = -\sum_{d \in Y}\sum_{f=1}^{F} Y_{df} ln Z_{df} \tag{5}$$

where Y_D is the set of mention indices that have labels and F is the dimension of the output feature. Y is the label indicator matrix. Similar to [39], the weight parameters W_0 and W_1 can be trained via gradient descent. The $\hat{A}XW_0$ contains the first layer (mention,sense) and embeddings, and $\hat{A}RELU(\hat{A}XW_0)W_1$ contains the second layer (mention,sense) and embeddings. This two-layer GCN performs message passing between nodes to two steps away, maximum. Therefore, the two-layer GCN allows the exchange of information between pairs of nodes. This GCN model on our experimental datasets shows better performance than a one-layer model and models with more than two layers. This shows the validity of our model, based on similar results in other recent works [8,9].

4 Experimental Setup

We present the settings of our evaluation of our representation in the English WSD task. This setup includes the benchmark, our representation setup for disambiguation task and state-of-the-art WSD models as our comparison systems.

Evaluation Benchmark We use the English WSD test set framework which is constructed by five standard evaluation benchmark datasets[3]. It is included of Senseval-2 [6], Senseval-3 [35], SemEval-07 [26], SemEval-13 [18], SemEval-15 [15] along with ALL, i.e., the concatenation of all the test sets [27].

Experiment Setup In our experiments, we use BERT pre-trained cased model. Similar to [34], among all the configurations reported by Devlin et al. (2019), we used the sum of the last four hidden layers as contextual embeddings of the words since they showed it has better performance. In order to be able to compare our system with supervised models, we build a supervised version of our representations. This version combines the gloss and contextual information with the sense-annotated contexts in SemCor [14], a corpus of 40K sentences where words have been manually annotated with a WordNet meaning. We leveraged SemCor for building a representation of each sense therein. To this end, we followed [22], given a mention-sense pair (m, s), we collected all the sentences $c_1, ..., c_n$ where m appears tagged with s. Then, we fed all the retrieved sentences into BERT and extracted the embeddings BERT(c_1, m), ... , BERT(c_n, m). The final embedding of s was built by the average of its context and sense gloss vectors and its representation coming from SemCor, i.e., the average of BERT(c_1, m), ... , BERT(c_n, m). We note that when a sense did not appear in SemCor, and we built its embedding by replacing the SemCor part of the vector with its sense gloss representation.

WSD Model For WSD modeling, we employed a 1-nearest neighbor approach– as previous methods in the literature– to test our representations on the WSD task. For each target word m in the test set, we computed its contextual embedding by means of BERT and compared it against the embeddings of our representation associated with the senses of m. Hence, we took as a prediction for the target word the sense corresponding to its nearest neighbor. We note that the embeddings produced by our representations are created by averaging two BERT representations, i.e., context and sense gloss (see Section 3.3), hence we repeated the BERT embedding of the target instance to match the number of dimensions.

Comparison Systems We compared our representation against the best recent performing systems evaluated on the English WSD task. LMMS is one of these systems which generates sense embedding with full coverage of Wordnet. It uses pre-trained ELMO and BERT models, as well as the relations in a lexical knowledge base to create contextual embeddings [10]. SensEmBERT is the next system that relies on different resources for building sense vectors. These resources include Wikipedia, BabelNet, NASARI lexical vectors, and BERT. It computes context-aware representations of BabelNet senses by combining the semantic and textual information derived from multilingual resources. This model uses the BabelNet mapping between WordNet senses and Wikipedia pages which drops the need for sense-annotated corpora [34]. The next

[3] http://lcl.uniroma1.it/wsdeval/

Table 1: F-Measure performance of WSD evaluation framework on the test sets of the unified dataset.

Model	Senseval-2	Senseval-3	Semeval-7	Semeval-13	Semeval-15	All
BERT	77.1±0.3	73.2±0.4	66.1±0.3	71.5±0.2	74.4±0.3	73.8±0.3
LMMS	76.1±0.6	75.5±0.2	68.2±0.4	75.2±0.3	77.1±0.4	75.3±0.2
SensEmBERT	72.4±0.1	69.8±0.2	60.1±0.4	78.8±0.1	75.1±0.2	72.6±0.3
ARES	78.2±0.3	77.2±0.1	71.1±0.2	77.2±0.2	83.1±0.2	77.8±0.1
our model	79.6±0.2	78.5±0.2	74.6±0.3	79.3±0.6	82.9±0.4	78.9±0.1

comparison system is ARES, a semi-supervised approach to produce sense embeddings for all the word senses in a language vocabulary. ARES compensates for the lack of manually annotated examples for a large portion of words' meanings. ARES is the most recent contextualized word embedding system, to our knowledge. In our comparisons, we also considered BERT as a comparison system since it is at the core of all the considered methods. BERT also has shown good performance in most NLP tasks by using pre-trained neural networks.

5 Results

The results of our evaluations on the WSD task are represented in this section. We show the effectiveness of our representation by comparing it with the existing state-of-the-art models on the standard WSD benchmarks. In Table 1 we report the results of our representation and compare it against the results obtained from other state-of-the-art approaches on all the nominal instances of the test sets in the framework of [27]. All performances are reported in terms of F1-measure, i.e., the harmonic mean of precision and recall. As we can see, our model achieves the best results on the datasets when compared to other precious contextualized approaches. It indicates that our representation is competitive with these previous models. These results show the novel idea in the nature of creating this new representation has improved the lexical ambiguity. It is a good indicator of the dependency of the WSD task to the representation that is aware of the context and the information extracted from the reference knowledge base.

Analysis by Part-of-Speech One other possible way to analyze the errors that arise in WSD with each embedding approach is to measure the frequency of mis-disambiguation in different parts of speech. The considered parts of speech are nouns, verbs, adjectives, and adverbs, as are the covered types in the datasets. The F-measure performance of the 1-NN WSD of each embedding on All dataset is shown in Table 3 which is categorized by parts of speech. As it shows, the type in which its disambiguation has been correct more than other types is adverbs. At the same time, verbs are the ones that are difficult to disambiguate because they have the lowest mis-disambiguation frequency across all language models. In each one of the models, disambiguating the nouns is more accurate than verbs, when the embedding model is BERT. The coverage of verb senses can

Table 2: The Number of instances and ambiguity level of the concatenation of all five WSD datasets [27].

	Nouns	Verbs	Adj.	Adv	All
#Entities	4300	1652	955	346	7253
Ambiguity	4.8	10.4	3.8	3.1	5.8

explain this disambiguation performance difference between verbs and the other three parts of speech in WordNet, significantly less than the coverage of noun senses. To be more specific with our quantitative POS analysis, we tried to find the type of words in all datasets with more errors when disambiguating with different representations. We evaluate the effectiveness of our representation on parts of speeches, in comparison with the recent methods. The parts of speech that we have in the dataset are nouns, verbs, adjectives, and adverbs. Table 2 shows the number of instances in each category. In our second evaluation, we examined the effect of our representation against previous ones on each word category. Table 3 represents the F-Measure performance of the 1-NN WSD of each one of the contextualized word embeddings which we considered on All datasets split by parts of speech.

Table 3: F-Measure performance of the 1-NN WSD of each embedding on the standard WSD dataset split by parts of speech. The dataset in this experiment is a concatenation of all five datasets, which is split by Part-of-Speech tags.

Model	Nouns	Verbs	Adjectives	Adverbs
BERT	76.2±0.2	62.9±0.5	79.7±0.2	85.5±0.5
LMMS	78.2±0.6	64.1±0.3	81.3±0.1	82.9±0.3
SensEmBERT	77.8±0.3	63.4±0.5	80.1±0.4	86.4±0.2
ARES	78.7±0.1	67.3±0.2	82.6±0.3	87.1±0.4
our model	79.6±0.2	69.6±0.1	85.2±0.1	89.3±0.5

6 Conclusion

In this paper, we consider the problem of text ambiguity and one of its important factors, the word representation. We evaluate the pros and cons of current state-of-the-art approaches for word embedding, and applied them in parts of speeches on the standard datasets. By observing the opportunities to improve a word embedding model,

we present a novel approach for creating word embeddings. In our model, we consider the knowledge base and the context of the input document text, when generating the representation. We showed that this context-rich representation is beneficial for lexical ambiguity in English. The results of experiments in the WSD task show the efficiency of our representations compared to other state-of-the-art methods, despite relying only on English data. We further tested our embeddings on the split data into four parts of speeches. As the results of our second experiment show, the effectiveness of the contextualized embeddings in WSD on verbs is not as good as on nous. This defect is because of the lack of instances in the dataset in each word category. As future work, one point to improve our representations in the text ambiguity task is by training the model with data including more verbs than the current one.

References

1. Agirre, E., de Lacalle, O.L., Soroa, A.: Random walks for knowledge-based word sense disambiguation. Computational Linguistics **40**(1), 57–84 (Mar 2014), https://direct.mit.edu/coli/article/40/1/57/145
2. Aleksandrova, D., Drouin, P., Lareau, F. c c.o., Venant, A.: The multilingual automatic detection of 'e nonc é s bias 'e s in wikip é dia. ACL (2020), https://www.aclweb.org/anthology/R19-1006.pdf
3. Azad, H.K., Deepak, A.: A new approach for query expansion using wikipedia and wordnet. Information sciences **492**, 147–163 (2019), https://www.sciencedirect.com/science/article/pii/S0020025519303263
4. Calvo, H., Rocha-Ramírez, A.P., Moreno-Armendáriz, M.A., Duchanoy, C.A.: Toward universal word sense disambiguation using deep neural networks. IEEE Access **7**, 60264–60275 (2019), https://ieeexplore.ieee.org/abstract/document/8706934
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In North American Association for Computational Linguistics (NAACL) (2018), https://www.aclweb.org/anthology/N19-1423/
6. Edmonds, P., Cotton, S.: SENSEVAL-2: Overview. In: Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems. pp. 1–5. Association for Computational Linguistics, Toulouse, France (Jul 2001), https://www.aclweb.org/anthology/S01-1001.pdf
7. Fogarolli, A.: Word sense disambiguation based on wikipedia link structure. In: 2009 IEEE International Conference on Semantic Computing. pp. 77–82. IEEE (2009), https://ieeexplore.ieee.org/stamp/stamp.jsp
8. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
9. Li, Q., Han, Z., Wu, X.M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: AAAI. vol. 32, pp. 234–242. Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans (2018)
10. Loureiro, D., Jorge, A.: Language modelling makes sense: Propagating representations through wordnet for full-coverage word sense disambiguation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.Florence, Italy. p. 5682–5691 (2019), https://www.aclweb.org/anthology/P19-1569
11. Martinez-Rodriguez, J.L., Hogan, A., Lopez-Arevalo, I.: Information extraction meets the semantic web: a survey. Semantic Web **Preprint**, 1–81 (2020), http://repositorio.uchile.cl/bitstream/handle/2250/174484/Information-extraction-meets-the-Semantic-Web.pdf?sequence=1

12. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. ICLR **4**, 321–329 (2013), https://arxiv.org/pdf/1301.3781.pdf
13. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to wordnet: An on-line lexical database. International journal of lexicography **3**(4), 235–244 (1990), https://watermark.silverchair.com/235.pdf
14. Miller, G.A., Leacock, C., Tengi, R., Bunker, R.T.: A semantic concordance. In: Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993 (1993), https://www.aclweb.org/anthology/H93-1061/
15. Moro, A., Navigli, R.: SemEval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In: SEM. pp. 288–297. Association for Computational Linguistics, Denver, Colorado (Jun 2015), https://www.aclweb.org/anthology/S15-2049.pdf
16. Moro, A., Raganato, A., Navigli, R.: Entity linking meets word sense disambiguation: a unified approach. Transactions of the Association for Computational Linguistics **2**, 231–244 (2014), https://watermark.silverchair.com/tacl_a_00179.pdf
17. Navigli, R.: Word sense disambiguation: A survey. ACM computing surveys (CSUR) **41**(2), 1–69 (2009), https://dl.acm.org/doi/abs/10.1145/1459352.1459355
18. Navigli, R., Jurgens, D., Vannella, D.: SemEval-2013 task 12: Multilingual word sense disambiguation. In: SEM. Association for Computational Linguistics, Atlanta, Georgia, USA (Jun 2013), https://www.aclweb.org/anthology/S13-2040.pdf
19. Navigli, R., Ponzetto, S.P.: Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artificial intelligence **193**, 217–250 (2012), https://www.sciencedirect.com/science/article/pii/S0004370212000793
20. Pasini, T., Elia, F.M., Navigli, R.: Huge automatically extracted training sets for multilingual word sense disambiguation. arXiv preprint arXiv:1805.04685 (2018), https://arxiv.org/abs/1805.04685
21. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing. pp. 1532–1543. EMNLP, Qatar (2014), https://www.aclweb.org/anthology/D14-1162.pdf
22. Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. Association for Computational Linguistics pp. 2227–2237 (2018), https://www.aclweb.org/anthology/N18-1202
23. Peters, M.E., Logan IV, R.L., Schwartz, R., Joshi, V., Singh, S., Smith, N.A.: Knowledge enhanced contextual word representations. arXiv preprint arXiv:1909.04164 (2019), https://arxiv.org/pdf/1909.04164.pdf
24. Peters, M.E., Neumann, M., Zettlemoyer, L., Yih, W.t.: Dissecting contextual word embeddings: Architecture and representation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing p. 1499–1509 (2018), https://www.aclweb.org/anthology/D18-1179/
25. Pilehvar, M.T., Jurgens, D., Navigli, R.: Align, disambiguate and walk: A unified approach for measuring semantic similarity. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1341–1351 (2013), https://www.aclweb.org/anthology/P13-1132.pdf
26. Pradhan, S., Loper, E., Dligach, D., Palmer, M.: SemEval-2007 task-17: English lexical sample, SRL and all words. In: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007). pp. 87–92. Association for Computational Linguistics, Prague, Czech Republic (Jun 2007), https://www.aclweb.org/anthology/S07-1016
27. Raganato, A., Camacho-Collados, J., Navigli, R.: Word sense disambiguation: A unified evaluation framework and empirical comparison. In: Proceedings of the 15th Conference of

the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. pp. 99–110 (2017), https://www.aclweb.org/anthology/E17-1010/
28. Reisinger, J., Mooney, R.: Multi-prototype vector-space models of word meaning. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. pp. 109–117 (2010), https://www.aclweb.org/anthology/N10-1013.pdf
29. Saeidi, M., Kosmajac, D., Taylor, S.: Dnlp@ fintoc'20: Table of contents detection in financial documents. In: Proceedings of the 1st Joint Workshop on Financial Narrative Processing and MultiLing Financial Summarisation. pp. 169–173 (2020)
30. Saeidi, M., Milios, E., Zeh, N.: Contextualized knowledge base sense embeddings in word sense disambiguation. In: International Conference on Document Analysis and Recognition. pp. 174–186. Springer (2021)
31. Saeidi, M., Milios, E., Zeh, N.: Graph representation learning in document wikification. In: International Conference on Document Analysis and Recognition. pp. 509–524. Springer (2021)
32. Saeidi, M., Sousa, S.B.d.S., Milios, E., Zeh, N., Berton, L.: Categorizing online harassment on twitter. In: Joint European Conference on Machine Learning and KDD. pp. 283–297. Springer (2019), https://link.springer.com/chapter/10.1007/978-3-030-43887-6_22
33. Scarlini, B., Pasini, T., Navigli, R.: Just "onesec" for producing multilingual sense-annotated data. In: Proceedings of ACL. pp. 699–709 (2019), https://www.aclweb.org/anthology/P19-1069.pdf
34. Scarlini, B., Pasini, T., Navigli, R.: Sensembert: Context-enhanced sense embeddings for multilingual word sense disambiguation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 8758–8765 (2020), https://ojs.aaai.org//index.php/AAAI/article/view/6402
35. Snyder, B., Palmer, M.: The English all-words task. In: Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text. pp. 41–43. Association for Computational Linguistics, Barcelona, Spain (Jul 2004), https://www.aclweb.org/anthology/W04-0811
36. Weikum, G., Dong, L., Razniewski, S., Suchanek, F.: Machine knowledge: Creation and curation of comprehensive knowledge bases. arXiv preprint arXiv:2009.11564 (2020), https://arxiv.org/pdf/2009.11564.pdf
37. West, R., Paranjape, A., Leskovec, J.: Mining missing hyperlinks from human navigation traces: A case study of wikipedia. In: Proceedings of the 24th international conference on World Wide Web, pp. 1242–1252 (2015), https://dl.acm.org/doi/pdf/10.1145/2736277.2741666
38. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. Curran Associates, Inc. **32**, 221–229 (2019), https://proceedings.neurips.cc/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf
39. Yao, L., Mao, C., Luo, Y.: Graph convolutional networks for text classification. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 7370–7377. AAAI, Honolulu (2019)
40. Zhao, G., Wu, J., Wang, D., Li, T.: Entity disambiguation to wikipedia using collective ranking. Information Processing & Management **52**(6), 1247–1257 (2016), https://www.sciencedirect.com/science/article/pii/S0306457316301893

Contextual Language Models for Knowledge Graph Completion

Russa Biswas, Radina Sofronova, Mehwish Alam, and Harald Sack

FIZ Karlsruhe - Leibniz Institute for Information Infrastructure,
Karlsruhe Institute of Technology, AIFB
firstname.lastname@fiz-karlsruhe.de

Abstract. Knowledge Graphs (KGs) have become the backbone of various machine learning based applications over the past decade. However, the KGs are often incomplete and inconsistent. Several representation learning based approaches have been introduced to complete the missing information in KGs. Besides, Neural Language Models (NLMs) have gained huge momentum in NLP applications. However, exploiting the contextual NLMs to tackle the Knowledge Graph Completion (KGC) task is still an open research problem. In this paper, a GPT-2 based KGC model is proposed and is evaluated on two benchmark datasets. The initial results obtained from the fine-tuning of the GPT-2 model for triple classification strengthens the importance of usage of NLMs for KGC. Also, the impact of contextual language models for KGC has been discussed.

Keywords: GPT-2 · Knowledge Graph Embedding · Triple Classification.

1 Introduction

Knowledge Graphs (KGs) such as DBpedia, YAGO, Freebase, etc. have emerged as the backbone of various applications in Natural Language Processing (NLP) such as entity linking [9], question answering [2], etc. KGs are multi-relational directed graphs with nodes as real world entities and relationships between them are represented on the edges. The facts are represented as a triple $< h, r, t >$, where h and t are the head and tail entities respectively and r represents the relation between them. However, these KGs are often incomplete. Knowledge Graph Completion (KGC) is the task of predicting the missing links between entities, mining missing relations, and discovering new facts. Recent years have witnessed extensive research on KGC with a focus on representation learning. Most of these models use structural information i.e., the triple information such as TransE [3], ConvE [5] whereas a few others include textual entity descriptions such as TEKE [22], DKRL [25], etc. However, the models considering the textual information leverage only static word embedding approaches, such as word2vec, GloVe etc. to generate the latent representation of the textual entity descriptions. Consequently, the semantic information encoded in the contextual entity embeddings are not exploited for KGC.

On the other hand, pre-trained contextualized Neural Language Models (NLMs) such as BERT [11], GPT-2 [20], have gained huge momentum in applications of NLP. These models are trained on huge amount of free text resulting in encoding of the semantic information leading to better linguistic representation of the words. GPT-2 is one of the distinguished models which has achieved state-of-the-art results for various language understanding based tasks. It operates on a transformer decoder architecture with attention masks to predict next word of a sequence.

However, a combination of contextualized NLMs for the task of KGC is an open research problem. KG-BERT [29] is one of the pioneers in this research in which the BERT model is fine-tuned on KG data and has been used for link prediction and triple classification as sub-tasks of KGC. The results presented in [29] depict that the information contained in pre-trained NLMs play an important role in the predicting the missing links in a KG. Inspired by KG-BERT, a novel GPT-2 based KGC model is explored in this work for the triple classification sub-task. The triples in a KG are considered as sentences and the triple classification is considered as a sequence classification problem. Furthermore, an analysis of the contextualised NLMs for KGC is also provided.

The rest of the paper is organised as follows. To begin with, a review of the related work is provided in Section 2 followed by the preliminaries in Section 3. Section 4 accommodates the outline of the proposed approach followed by experimental results in Section 5. Finally, an outlook of future work is provided in Section 6.

2 Related Work

This section presents the state-of-the-art (SOTA) models for KG embeddings with a focus on the models considering the textual descriptions.

A large variety of KG embedding approaches has been explored for the task of link prediction, such as translational models like TransE [3] and its variants, semantic matching models like DistMult [28], neural network based models like ConvE [5], graph structure based like GAKE [6], and literal (e.g., text, image, number, etc.) based like DKRL [25], Jointly(ALSTM) [27], MKBE [17], etc.

In a translational model such as TransE [3], given a triple (e_h, r, e_t) in a KG G, the relation r is considered as a translation operation between the head and tail entities on a low dimensional vector space defined by $\mathbf{e_h} + \mathbf{r} \approx \mathbf{e_t}$, where $\mathbf{e_h}, \mathbf{r}, \mathbf{e_t}$ are the embeddings of the head, relation and the tail entity respectively.

Another set of algorithms improve KG embeddings by taking into account different kinds of literals such as numeric, text or image literals and a detailed analysis of the methods is provided in [7]. DKRL [25] extends TransE [3] by incorporating the textual entity descriptions in the model. The textual entity descriptions are encoded using a continuous bag-of-words approach as well as a deep convolutional neural network based approach. Jointly(ALSTM) is another entity description based embedding model which extends the DKRL model with a gate strategy and uses attentive LSTM to encode the textual entity descrip-

tions. KG-BERT [29] is a contextual NLM based model which is fine tuned on BERT and have been used in downstream tasks.

However, the contextual NLMs are not considered to encode the triples or the entity descriptions in all the models except KG-BERT. Therefore, this study proposes a novel model in which the KG is fine-tuned with GPT-2 for KGC.

3 Preliminaries

A detailed explanation of pre-trained NLMs and KGC is provided in this section.

3.1 Language Models

A LM learns the probability of word occurrences based on a text corpus which is used for various machine learning based NLP applications such as Machine Translation [12], Speech Recognition [30], etc. It is the task of assigning probability to each sequence of words or a probability for the likelihood of a given word based on a sequence of words. [8]. LMs can be broadly divided into

- **Statistical Language Models (SLMs)** are *n-gram* based approaches that assign probabilities to a sequence s of n words, and is given by

$$P(s) = P(w_1 w_2 ... w_n) = P(w_1)P(w_2|w_1)...P(w_n|w_1 w_2 ... w_{(n-1)}), \quad (1)$$

  where w_i denotes $i-$th word in the sequence s. The probability of a word sequence is the product of the conditional probability of the next word given the previous words or the context [10]. The SLMs fail to assign probabilities to the n-grams that do not appear in the training corpus which is tackled using the smoothing techniques. However, the curse of dimensionality refrains the SLMs models to be trained on huge corpora.
- **Neural Language Models (NLMs)**, on the other hand, are neural network based LMs 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. The contextual information is captured on all the different levels in the text corpus, such as, sentences, sub-word, character, as well as the entire corpus.

The NLMs such as Word2Vec [13], BERT [11], GPT [19] etc. are beneficial for several NLP downstream tasks, such as question answering [23], sentiment analysis [26], etc. As mentioned in [18], these models can be further sub-divided into **(i)** Non-contextual and **(ii)** Contextual Embeddings. The Non-contextual word embeddings such as Word2Vec, GloVe, etc., are static in nature and are context independent. Although, the latent representations of the words capture the semantic meanings but they do not dynamically change according to the context the words appear in. However, Contextual embeddings such as BERT, GPT, etc., encode semantics of the words differently based on different contexts. All the language models are trained on huge unlabelled text corpora resulting in

increased number of model parameters. Therefore, the pre-trained models help in learning universal language representations of the words. It promotes better initialization of the model to have a better generalization performance on the downstream tasks. Pre-training of the NLMs also helps in avoiding overfitting of the model for small corpora [18]. Also, it improves the reuseability of the model as it prevents the training of the model from scratch. However, fine tuning of pre-trained contextual NLMs is often required to adapt the model to the specific data for the down-stream task. It bridges the gap between the data on which a particular NLM is trained on and the target data distribution.

3.2 Knowledge Graph Completion

The goal of KGC is the task of predicting missing instances or links to deal with the incompleteness and sparsity in KGs. As explained in [4] KGC methods can be broadly divided into the following classes:

- **Rule Based Models** that use rules or statistical features such as NELL [15], KGRL [24], etc., to infer new knowledge in KGs.
- **Representation Learning Based Models** such as TransE [3], ConvE [5], etc., that learn the latent representation of the entities and relations into a low-dimensional continuous vector space, in which semantically similar entities are placed closer to each other. These representations are then used for the KGC tasks of link prediction and triple classification.

In link prediction task, the head or tail entity in a triple $< h, r, ? >$ or $<?, r, t >$ is predicted by defining a mapping function $\psi : E \times R \times E \rightarrow R$, where E and R are the set of entities and relations in the KG. A score is assigned to each triple, where the higher the score of the triple indicates the more likely to be true. The triple classification task involves the training of binary classifier whether a given triple is false (0) or true (1).

4 Language Models for Knowledge Graph Completion

This section comprises of an analysis of NLMs on KGs followed by a detailed description of the GPT-2 based KGC task. The basic idea of the approach lies in the fact that the contextual NLMs trained on huge corpora also capture relational information present in the training data [16]. Consequently, NLM models can be exploited further to predict the missing links in a KG. However, the impact of the pre-trained contextual NLMs for KGC is still an open research.

BERT for KGC One of the pioneers in this domain is the KG-BERT [29] model in which the pre-trained BERT model is fine-tuned on KGs for KGC. Each triple $< h, r, t >$ is considered as a sentence and is provided as an input sentence of the BERT model for fine-tuning. For the entities, KG-BERT has been trained with either the entity names or their textual entity descriptions.

The first token of every input sequence is always $[CLS]$, whereas the separator token $[SEP]$ separates the head entity, relation and the tail entity. Therefore, each input sequence for the BERT model is given by
([CLS] head entity/description [SEP] relation [SEP] tail entity/description [SEP]).
A sigmoid scoring function is introduced on the top of the final layer for the triple classification which is a 2-dimensional vector $\in [0, 1]$.

GPT-2 for KGC Inspired by KG-BERT, GPT-2 [20] is exploited in this work for KGC. GPT-2 is a large transformer-based language model trained on 8 million web pages with 1.5 billion parameters. The model predicts the next word based on all the previous words in the text corpus. An attention mechanism is used to selectively focus on the segments of the input text. The architecture comprises of a 12-layer decoder-only transformer, using 12 masked self-attention heads, with 64 dimensional states each. The Adam optimization is used and the learning rate was increased linearly from zero to a maximum of 2.5×10^{-4}. The model was able to outperform the previous NLMs on language tasks like question answering, reading comprehension, summarization, translation, etc. However, the basic difference between BERT and GPT-2 is that BERT uses transformer encoder blocks whereas GPT-2 uses transformer decoder blocks.

Similar to KG-BERT, GPT-2 is also fine tuned with KG triples where each triple is considered as an input sequence. In this model, two variants have been used to model the input sequence for the fine-tuning task. Given a triple *Albert Einstein, bornIn, Germany*, the input sequence is modelled as

- Albert Einstein bornIn Germany $[EOS]$,
- $[BOS]$ Albert Einstein $[EOS]$ bornIn $[EOS]$ Germany $[EOS]$,

where *[BOS]* and *[EOS]* are the beginning of sequence and end of sequence respectively. Both entity names and descriptions are considered for the head and tail entity. The input sequences are fed into the GPT-2 model architecture which is a transformer decoder based on the original implementation [20]. It consists of stacked decoder blocks of the transformer architecture and the context vector is initialised with zero for the first word embedding. The masked self-attention is used to extract information from the prior words in the sentence as well as the context word. The word vectors in the first layer of GPT-2 follows byte pair encoding i.e., tokens are parts of words. Furthermore, it compresses the tokenized words list into a set of vocabulary items by considering the most common word components. The GPT-2 sequence classification module is leveraged to determine the plausibility of the triples. Since, GPT-2 outputs one token at a time, the classifier is built on the last token. A 2-dimensional vector $\in [0, 1]$ sigmoid scoring function is introduced for triple classification.

5 Experiments

This section comprises of an analysis of the initial results obtained on deploying GPT-2 model on the triple classification task for KGC. The model has been evaluated on two benchmark datasets WN11 and FB13.

Table 1. Dataset Statistics

Dataset	#Ent.	#Rel.	#Train	#Val.	#Test
WN11	38,696	11	112,581	2,609	10,544
FB13	75,043	13	316,232	5,908	23,733

Table 2. Results of Language Models on Triple Classification (accuracy in %)

Model Types	Models	WN11	FB13
KG embeddings with Textual	TEKE	86.1	84.2
Contextual LMs	KG-BERT (labels)	93.5	79.2
	KG-BERT (description)	-	90.4
	Ours with GPT2 (labels)	83	73
	Ours with GPT2 (description)	85	89

Datasets The two benchmark datasets WN11 and FB13 are subsets of WordNet and Freebase KGs respectively and are introduced in [21]. WordNet [14] is a large lexical KG of English comprising of nouns, verbs, adjectives and adverbs. They are grouped into sets of cognitive synonyms known as synsets. Each synset expresses a distinct concept. They are interlinked by means of conceptual-semantic and lexical relations. Freebase [1] is a large collaborative KG consisting of structured data captured from various sources including individual, user-submitted wiki contributions. The statistics of the KGs used to fine-tuning with GPT-2 followed by triple classification is provided in Table 1.

Experimental Setup The pre-trained GPT-2 base model with 12 decoder layers, 768 hidden layers, 12 attention heads and 117M parameters is used for fine-tuning. The set of hyperparameters chosen are as follows: batch sizes = $\{256, 128, 32, 8, 1\}$, epochs = $\{5, 3\}$, and learning rate = $\{2e-5, 5e-5\}$. The experiments with GPT-2 have been performed on an Ubuntu 16.04.5 LTS system with 503GB RAM and Tesla V100S GPU.

Results The results depicted in Table 2 represent some initial results on the triple classification task using the pre-trained GPT-2 model on KGs. Since all the triples in the training set are true, a negative sampling method is used to generate synthetic negative triples for the training of the classifier. The negative triples are generated for this task, by replacing the head and the tail entities with arbitrary entities based on a local closed world assumption. In this work, filtered settings is used, i.e., if by chance true triples are generated using negative sampling methods, then they are removed. Therefore, the set of triples in the train, test, and validation sets are disjoint.

TEKE [22] and KG-BERT are considered as baseline models as they consider NLMs to model the KGs for KGC. TEKE exploits structural information of the KGs using an embedding layer, a BiLSTM layer followed by mutual attention

Table 3. Results with the pre-trained GPT2 model for Triple Classification with different parameter settings

Dataset	Feature	Model details	Precision	Recall	F_1-score
WN11	Labels	batch=128, epoch=10, lr=2e-5	0.76	0.76	0.76
		batch=32, epoch=3, lr=5e-5	0.74	0.74	0.74
		batch=1, epoch=3, lr=5e -5	0.83	0.83	0.83
	Description	batch=8, epoch=5, lr=2e - 5	0.79	0.79	0.79
		batch=1, epoch=3, lr=5e -5	0.85	0.85	0.85
FB13	Labels	batch=32, epoch=10, lr=2e -5	0.69	0.64	0.61
		batch=256, epoch=5, lr=2e -5	0.68	0.68	0.68
	Description	batch=1, epoch=3, lr=5e-5	0.90	0.89	0.89

layer. The results of the baselines are taken from the KG-BERT paper [29] except for KG-BERT (labels) variant for FB13. The experiment for this variant is performed with the same settings as mentioned in [29]. It is observed from the results that with GPT-2, the model achieves comparable results with the previous models. Also, the results are better for GPT-2 with descriptions variant, this is because the textual entity descriptions have more contextual information resulting in generation of better representation of triples. The same behaviour has been observed for KG-BERT. Since the NLMs are trained on large corpora, the model parameters contain huge amount of linguistic knowledge which helps in overcoming the data sparsity problem in KGs. Furthermore, the main advantage of contextual NLM based KGC methods that they do not consider the structural information of the entities in a KG. Hence it is independent of any underlying structure in a KG. Furthermore, these models are also applicable to the less popular entities in KGs with lesser number of triples compared to the others. The task of triple classification in KGC with GPT-2 is similar to the sequence classification task in text and the self attention mask helps in identifying the important words in the sequences. The variants with labels i.e., the entity names for both KG-BERT and the proposed GPT-2 based model work better for WN11 as compared to FB13. This is because WordNet is a linguistic KG and the NLMs are able to capture more information on the entity names as compared to FB13.

Table 3 depicts the precision, recall, and F_1 score of the model with different hyper-parameter settings. It is observed that the best results are obtained with batch=1, epoch=3, and lr=5e-5. The changing of epochs does not have much variation in the model whereas batch size has. The lower the batch size, the better the performance of the model.

6 Conclusion and Future Work

This work presents an analysis of the effect of exploiting NLMs for KGC. A novel GPT-2 based KGC model has also been proposed. The initial results from the triple classification sub-task shows that the semantic information stored in the NLMs can provide vital information for the KGC task. In future, further hyper-

parameter tuning will improve model performance and additional experiments on link prediction sub-tasks will be conducted.

References

1. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of data. pp. 1247–1250 (2008)
2. Bordes, A., Chopra, S., Weston, J.: Question answering with subgraph embeddings. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 615–620 (2014)
3. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. Advances in neural information processing systems **26** (2013)
4. Chen, Z., Wang, Y., Zhao, B., Cheng, J., Zhao, X., Duan, Z.: Knowledge graph completion: A review. IEEE Access **8**, 192435–192456 (2020)
5. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: Thirty-second AAAI conference on artificial intelligence (2018)
6. Feng, J., Huang, M., Yang, Y., Zhu, X.: GAKE: Graph aware knowledge embedding. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. pp. 641–651. The COLING 2016 Organizing Committee, Osaka, Japan (Dec 2016), `https://www.aclweb.org/anthology/C16-1062`
7. Gesese, G.A., Biswas, R., Alam, M., Sack, H.: A survey on knowledge graph embeddings with literals: Which model links better literal-ly? Semantic Web (Preprint), 1–31
8. Goldberg, Y.: Neural network methods for natural language processing. Synthesis lectures on human language technologies **10**(1), 1–309 (2017)
9. Hoffart, J., Yosef, M.A., et al., I.B.: Robust disambiguation of named entities in text. In: Proc. of the 2011 Conf. on Empirical Methods in Natural Language Processing, EMNLP 2011. pp. 782–792 (2011)
10. Jing, K., Xu, J.: A survey on neural network language models. arXiv preprint arXiv:1906.03591 (2019)
11. Kenton, J.D.M.W.C., Toutanova, L.K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT. pp. 4171–4186 (2019)
12. Koehn, P.: Statistical machine translation. Cambridge University Press (2009)
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
14. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)
15. Paulheim, H., Bizer, C.: Improving the quality of linked data using statistical distributions. International Journal on Semantic Web and Information Systems (IJSWIS) **10**(2), 63–86 (2014)
16. Petroni, F., Rocktäschel, T., Riedel, S., Lewis, P., Bakhtin, A., Wu, Y., Miller, A.: Language models as knowledge bases? In: Proceedings of the 2019 Conference

on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 2463–2473 (2019)
17. Pezeshkpour, P., Chen, L., Singh, S.: Embedding multimodal relational data for knowledge base completion. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 3208–3218 (2018)
18. Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., Huang, X.: Pre-trained models for natural language processing: A survey. Science China Technological Sciences pp. 1–26 (2020)
19. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018)
20. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
21. Socher, R., Chen, D., Manning, C.D., Ng, A.: Reasoning with neural tensor networks for knowledge base completion. In: Advances in neural information processing systems. pp. 926–934 (2013)
22. Wang, Z., Li, J., Liu, Z., Tang, J.: Text-enhanced representation learning for knowledge graph. In: Proceedings of International Joint Conference on Artificial Intelligent (IJCAI). pp. 4–17 (2016)
23. Wang, Z., Ng, P., Ma, X., Nallapati, R., Xiang, B.: Multi-passage bert: A globally normalized bert model for open-domain question answering. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 5878–5882 (2019)
24. Wei, Y., Luo, J., Xie, H.: Kgrl: an owl2 rl reasoning system for large scale knowledge graph. In: 2016 12th International Conference on Semantics, Knowledge and Grids (SKG). pp. 83–89. IEEE (2016)
25. Xie, R., Liu, Z., Jia, J., Luan, H., Sun, M.: Representation learning of knowledge graphs with entity descriptions. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 30 (2016)
26. Xu, H., Liu, B., Shu, L., Philip, S.Y.: Bert post-training for review reading comprehension and aspect-based sentiment analysis. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 2324–2335 (2019)
27. Xu, J., Qiu, X., Chen, K., Huang, X.: Knowledge graph representation with jointly structural and textual encoding. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. pp. 1318–1324 (2017)
28. Yang, B., Yih, W., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), `http://arxiv.org/abs/1412.6575`
29. Yao, L., Mao, C., Luo, Y.: Kg-bert: Bert for knowledge graph completion. arXiv preprint arXiv:1909.03193 (2019)
30. Yu, D., Deng, L.: Automatic Speech Recognition. Springer (2016)

On Refining BERT Contextualized Embeddings using Semantic Lexicons

Georgios Zervakis[1][0000−0002−3015−2238], Emmanuel Vincent[1][0000−0002−0183−7289], Miguel Couceiro[1][0000−0003−2316−7623], and Marc Schoenauer[2][0000−0003−1450−6830]

[1] Université de Lorraine,CNRS, INRIA, LORIA, F-54000 Nancy, France
[2] INRIA TAU, LRI, France
{georgios.zervakis,emmanuel.vincent,miguel.couceiro,marc.schoenauer}@inria.fr

Abstract. Word vector representations play a fundamental role in many NLP applications. Exploiting human-curated knowledge was proven to improve the quality of word embeddings and their performance on many downstream tasks. Retrofitting is a simple and popular technique for refining distributional word embeddings based on relations coming from a semantic lexicon. Inspired by this technique, we present two methods for incorporating knowledge into contextualized embeddings. We evaluate these methods with BERT embeddings on three biomedical datasets for relation extraction and one movie review dataset for sentiment analysis. We demonstrate that the retrofitted vectors do not substantially impact the performance for these tasks, and conduct a qualitative analysis to provide further insights on this negative result.

Keywords: Contextualized embeddings · BERT · Knowledge integration · Retrofitting · Qualitative analysis

1 Introduction

The introduction of word embeddings was a breakthrough in NLP. Early approaches based on the *distributional hypothesis* — words that appear in the same context tend to be semantically similar — such as word2vec [11] provided a fixed embedding for each word. Recently, *contextualized embedding* systems like BERT [3] allow the generation of context-dependent word representations, which substantially improve the performance on many downstream NLP tasks.

Although such systems can be trained on data specific to the domain of interest, it is not yet clear how we can encode factual knowledge or impose constraints in the embeddings. Knowledge bases typically provide this type of

information, hence it is reasonable to exploit them in order to obtain more accurate and explainable embeddings.

Retrofitting [4] is a popular technique that modifies any set of pretrained distributional word embeddings to account for relational information encoded by a semantic lexicon. This is done as a post-processing step using an iterative update method called belief propagation [1] on a graph of relations obtained from the lexicon to update the word vectors. This method was proven to improve performance on various intrinsic and extrinsic evaluation tasks [2, 5, 9, 12, 13].

In this paper, we aim to extend retrofitting to operate with contextualized word embeddings. More specifically, we propose two different methods that, as in the original retrofitting approach, make use of similarity relations between words in order to move the respective embeddings closer to each other in the latent space. The first method combines the embedding of a given test sentence with the embeddings of sentences involving similar words in the training set, while the second method replaces a word in the test sentence by all possible similar words and combines the resulting embeddings. We evaluate the proposed methods with BERT embeddings on three biomedical datasets for a relation extraction task and one movie review dataset for sentiment analysis, and compare them with an oracle topline and two baselines (weighted majority vote and class posterior averaging). We show that both methods do not substantially impact the performance for this task, and conduct a qualitative analysis to provide further insights on this negative result.

The paper is organised as follows. We discuss related work in Section 2, and present the proposed methods in Section 3. We describe the experimental evaluation setup in Section 4, and we analyze the obtained results in Section 5. We provide conclusions and discuss future work in Section 6.

2 Related Work

There have been several attempts to improve the quality of word embeddings by incorporating knowledge into the process. Two main categories of methods can be distinguished, which we refer to as *joint* or *post-hoc*.

Joint methods integrate knowledge by retraining the embedding model from scratch using a modified training objective. For example, [10] proposed to replace the classical bag-of-words contexts in the word2vec Skip Gram model by dependency-based contexts, and showed that the resulting embeddings better reflect the syntactic similarities between words. In another approach, [19] modified a BiLSTM recurrent neural network to take into account information coming from the WordNet and NELL knowledge bases. To this end, they employed an attention mechanism that computes the relevance of candidate concepts from the knowledge base to the current input, and a second component that decides whether to exploit this information or not, and they reported improvements on both entity and event extraction tasks. In the same fashion, KnowBERT [15] incorporates WordNet and part of Wikipedia into BERT, showing the ability of the model to recall facts from the databases, improving downstream relation

extraction, entity typing and word sense disambiguation tasks at the same time. Nonetheless, joint methods come with the downside that they are model-specific, and often time-consuming since they require retraining the system afresh.

Post-hoc methods surpass these limitations, since knowledge is inserted in the word embeddings after training, regardless of the model used to obtain them. The most popular technique among these is retrofitting [4]. This is a graph-based approach that, given a semantic lexicon, i.e., a knowledge graph whose nodes represent words and edges represent relations between them, tries to reposition the word embeddings in such a way that they become closer (under some distance metric) to neighborhood embeddings in the graph. Initially, [4] considered a single type of relation between words, namely 'similarity'. Later approaches have extended retrofitting to account for 'dissimilarity' relations [9, 12, 13] and ordering (ranking) between the relations [6].

By default, all of the above retrofitting methods can only be applied to distributional word embeddings, i.e., a single representation vector per word. When we shift to contextualized embeddings, each word in the vocabulary can have a different representation in each sentence. An attempt to retrofit contextualized embeddings coming from ELMo is presented in the Paraphrase-aware Retrofitting (PAR) [16] method. More specifically, PAR learns an orthogonal transformation matrix that pulls closer the embeddings of words in paraphrased contexts, and separates those in unrelated contexts. However, this approach is limited to pairs of paraphrased contexts and cannot benefit from different sources of linguistic information. To our knowledge, there is no existing method for contextualized embeddings that takes full advantage of the benefits of retrofitting.

3 Proposed Contextualized Embedding Refinement Methods

As in the conventional retrofitting approaches discussed in Section 2, we assume a vocabulary of words $\mathcal{V} = \{w_1, \ldots, w_n\}$ and an ontology Ω of semantic relations between words in $\mathcal{V}$. We can then represent Ω in the form of an undirected graph $(\mathcal{V}, \mathcal{E})$, where nodes correspond to words in $\mathcal{V}$ and edges $(w_i, w_j) \in \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ to semantic relations between nodes. Now, suppose that we have a contextualized word representation model $\mathcal{M}$, along with a training corpus $\mathcal{D}_{\text{train}}$ on which it is fine-tuned and a test corpus $\mathcal{D}_{\text{test}}$ on which it is evaluated for a particular task.

3.1 Method A

The first proposed embedding refinement method, which we refer to as Method A, combines the contextualized embedding of a given word in the test set with the contextualized embeddings of all occurrences of all similar words in the training set. Let $\bar{q}_i \in \mathbb{R}^d$ be the contextualized embedding of word $w_i \in \mathcal{V}$ coming from $\mathcal{M}$ for a given test instance[3]. Let us further denote by $\mathcal{J}_i$ the set of words w_j

[3] For simplicity, $\bar{q}_i$ does not have a superscript for the test sentence as we only process one test sentence at a time.

which are adjacent to w_i according to Ω, and by $\mathcal{K}_j$ the set of training instances where w_j occurs. Then we define $\hat{q}_{jk} \in \mathbb{R}^d$ to be the contextualized embedding computed for all occurrences of w_j in $\mathcal{D}_{\text{train}}$, as index by $k \in \mathcal{K}_j$. The index sets $\mathcal{J}_i$ and $\mathcal{K}_j$ vary dynamically for every word.

The goal is to learn a new embedding q_i that it is close to $\bar{q}_i$ and to adjacent nodes in Ω under the $\mathcal{L}_2$ norm by minimizing

$$\mathcal{L}(q_i) = \|q_i - \bar{q}_i\|^2 + \sum_{j \in \mathcal{J}_i} \sum_{k \in \mathcal{K}_j} b_{ijk} \|q_i - \hat{q}_{jk}\|^2 \tag{1}$$

The weights b_{ijk} must naturally depend on the number of neighbours $|\mathcal{J}_i|$ of w_i, and on the number of occurrences $|\mathcal{K}_j|$ of each neighbor w_j in $\mathcal{D}_{\text{train}}$. In the following we define them as $b_{ijk} = c_{ij} \times d_{jk} = \frac{1}{|\mathcal{J}_i|^\alpha} \cdot \frac{1}{|\mathcal{K}_j|^\beta}$, $\alpha, \beta \in [0, \infty)$ where c_{ij} controls the contribution of each neighbour and d_{jk} controls the contribution of each of its occurrences. For example, $\alpha = \beta = 0$ results in equal weights $b_{ijk} = 1$ for all occurrences, while $\alpha = \beta = 1$ results in weights b_{ijk} that sum up to 1.

Equating to zero the derivative of $\mathcal{L}$ with respect to q_i and expressing the $\sum_k b_{ijk} \hat{q}_{jk}$ in terms of the mean $\mu_{\hat{q}_j}$ of all $\hat{q}_{jk}$ results in the following update rule:

$$q_i = \frac{\bar{q}_i + \sum_j \sum_k b_{ijk} \hat{q}_{jk}}{1 + \sum_j \sum_k b_{ijk}} = \frac{\bar{q}_i + |\mathcal{J}_i|^{-\alpha} \sum_j |\mathcal{K}_j|^{1-\beta} \mu_{\hat{q}_j}}{1 + |\mathcal{J}_i|^{-\alpha} \sum_j \mathcal{K}_j^{1-\beta}}. \tag{2}$$

The retrofitting operation therefore takes the form of a weighted average of the original embedding and the embeddings of all occurrences of all similar words in the training set.

3.2 Method B

The second proposed method, which we refer to as Method B, does not involve $\mathcal{D}_{\text{train}}$ at all. Instead, everything happens at test time. Again, we utilise $\mathcal{M}$ to obtain the embedding $\bar{q}_i$ of word w_i for a specific sentence in $\mathcal{D}_{\text{test}}$. In addition, we derive one embedding $\hat{q}_j$ for every word w_j which is adjacent to w_i according to Ω. To do so, we create a new sentence by replacing w_i with w_j in the test sentence, and repeat for every adjacent node of w_i in Ω. The objective is once more to learn a new vector q_i that is close to both $\bar{q}_i$ and all $\hat{q}_j$ under the $\mathcal{L}_2$ norm by minimizing

$$\mathcal{L}(q_i) = \|q_i - \bar{q}_i\|^2 + \sum_{j \in \mathcal{J}_i} b_{ij} \|q_i - \hat{q}_j\|^2 \tag{3}$$

Similarly to the above, we define the weights as $b_{ij} = \frac{1}{|\mathcal{J}_i|^\alpha}$, $\alpha \in [0, \infty)$.

Equating to zero the derivative of $\mathcal{L}$ with respect to q_i and expressing the $\sum_j b_{ij} \hat{q}_j$ in terms of the mean $\mu_{\hat{q}_j}$ of all $\hat{q}_j$ results in the following update rule:

$$q_i = \frac{\bar{q}_i + \sum_j b_{ij} \hat{q}_j}{1 + \sum_j b_{ij}} = \frac{\bar{q}_i + |\mathcal{J}_i|^{1-\alpha} \mu_{\hat{q}_j}}{1 + |\mathcal{J}_i|^{1-\alpha}}. \tag{4}$$

Again, the retrofitting operation takes the form of a weighted average of the original embedding and the embeddings of all neighbouring words.

The main difference between the two methods lies in the way we exploit the information coming from the knowledge graph. Method A typically results in a large number of neighbouring vectors $\hat{q}_{ik}$ that contain noise, since the context around the corresponding words differs from that of the test sentence in general. In contrast, Method B generates fewer neighbouring vectors $\hat{q}_j$ that share exactly the same context as the test sentence being processed.

4 Experimental Setup

In this section, we first provide information with respect to the data, the semantic lexicons and the contextual word embedding model we used to evaluate the proposed retrofitting methods. Then, we describe the experimental evaluation and we suggest three alternative strategies for comparison.

4.1 Data

We consider two tasks: relation extraction from biomedical data[4] and sentiment analysis of movie reviews. Two semantic verb lexicons are introduced in [2], referred to as **annotated** and **expanded clusters**. The former contains 192 verbs that appear frequently in a corpus of 2,230 biomedical journal articles, while the latter is an extended version of 1,149 verbs. Both lexicon come with three levels of granularity, i.e., verbs are grouped into 16, 34 and 50 classes[5], and are used for relation extraction.

ChemProt is a manually annotated corpus of relations between drugs/ chemical compounds and genes/proteins mentions found in PubMed abstracts. The relations are categorized into ten classes from which only five are used during evaluation. The task is to predict whether a pair of such entities is related or not, and if so, output the type of relation.

The **DDI** corpus aims in the development of systems that can automatically detect drug entities and drug-to-drug interactions in biomedical text. The corpus itself consists of texts from the DrugBank database and abstracts from the MedLine database. Annotations were provided by domain experts that classified drug-drug interactions into four DDI types.

i2b2 2010 corpus promotes the study of extraction/classification/relations of medical problems, tests, and treatments. The data consist of discharge summaries collected from Partners Healthcare, Beth Israel Deaconess Medical Center, and the University of Pittsburgh Medical Center, where relations of medical problems-treatments were grouped into eight classes.

[4] The biomedical datasets are included in the Biomedical Language Understanding Evaluation (BLUE) benchmark, as well as the preprocessing codes for creating the training, development and test sets.

[5] We refer to each different version of the verb lexicons simply by adding the number of the verb classes next to its name, e.g., annotated-34.

For the sentiment analysis task, we use the exact same semantic lexicons as in [4], namely, **FrameNet**, **PPDB** and two variants of **WordNet** which we refer to as $\textbf{WordNet}_{\text{syn}}$ and $\textbf{WordNet}_{\text{all}}$ (see more details in [4]). The size of these lexicons is relatively large, since they are general and contain knowledge about words which do not convey any sentiment, e.g., pronouns, prepositions, etc.. In order to focus on relevant words for the task, in conjunction with the semantic lexicons we utilize the **Bing Liu Sentiment Lexicon** [7], a domain-independent list of 6,786 adjectives that is manually created and that categorizes words as either positive or negative according to their sentiment.

SST-2 (Stanford Sentiment Treebank) [17] is a collection of 11,855 sentences from movie reviews including human annotations of their sentiment. The goal is to classify a given sentence as either positive or negative. Since the test labels are not publicly available, we split the training set such that 13% of the sentences are used for testing and the remaining are used for training. The resulting test set has 462 positive and 438 negative reviews, while the training set has 3,148 positive and 2,872 negative reviews. Finally, we use the development set provided by the authors.

4.2 BERT Architecture and Retrofitting

There are different locations within the architecture of BERT, where retrofitting transformations can be applied. In general, the model consists of 12 Transformer blocks [18] followed by a pooling layer, i.e., a fully connected layer with a dropout layer and a *tanh* activation. Each block contains a sequence of transformations that is divided into layers. The output layer of each block consists of a linear transformation, followed by dropout and layer normalisation. For both approaches we experimented with four retrofitting different settings: before **or** after layer normalisation at Transformer block 11 **or** 12.

The motivation behind these choices is related to the complex architecture of the model. We hypothesize that the impact of any change into the embeddings would be more noticeable as we get closer to the output space, rather than in earlier layers of the model. Thus, we started experimenting at the pooling layer, which is the closest to the output space, but the results were not promising. Consequently, we moved one step back at the output layer of the last Transformer block, and further back to the same place of the preceding Transformer block.

In the retrofitting equations (1) or (3), we initially considered as $\bar{q}_i$ the embedding corresponding to the word token in the test sentence, but preliminary experiments showed that this did not have an impact on the final performance. To verify this, we replaced the embeddings of these individual words with random numbers, or even zeroes. Both cases did not affect the performance, indicating that the output classifier is not very much dependent on single word embeddings. Instead, we focus on the `[CLS]` token embedding which is a weighted linear average of all word embeddings in the test sentence, it is closer to the output space, and has a bigger impact on the final result. All $\hat{q}_{ij}$ in (1) correspond to the activations of the word token in training sentences, whereas all $\hat{q}_j$ in (3) correspond to the activations of the `[CLS]` token in modified test sentences.

4.3 Technical Details

For the relation extraction task we chose BlueBERT [14] a specific variant of BERT that is further pre-trained on PubMed abstracts and clinical notes from MIMIC-III database, while for sentiment analysis we experimented with the classical BERT. In particular, for both tasks we selected the BERT-Base release of the model, which makes use of the exact same configurations, (e.g., vocabulary, length) as in the original BERT, and we further fine-tuned it on the downstream task for each dataset. We treat both tasks as a sentence classification problem. For relation extraction the named entities are anonymized with pre-defined tags (e.g., `@GENE`, `@CHEMICAL` for ChemProt) as in [8]. Then, we feed an input sentence into BERT which makes use of the `[CLS]` token of that sentence to perform the classification. In particular, the `[CLS]` representation is forwarded into the output layer of the last Transformer block, that produces an estimation for each class.

4.4 Grid Search Optimization

In order to find a good set of values for the retrofitting hyperparameters α, β, we performed a grid search using the development sets. For the first approach, we used both annotated and expanded clusters and we searched for $\alpha, \beta \in [0, 2]$ with a step of 0.2. We do not proceed on testing Method A for SST-2, as it turns out to be inferior to Method B. For the second approach, we use all four lexicons for sentiment analysis in conjunction with Bing Liu's sentiment lexicon (explained in Section in 4.1), while for relation extraction we only used the 34 and 50 classes of the annotated clusters[6]. Once again, we performed a grid search on the development sets where we searched for $\alpha \in [0, 2]$ with a step of 0.2.

4.5 Alternative Classification Strategies

In order to assess the ability of our method to leverage the information in the lexicons, we augmented all datasets by adding all modified sentences that occur by replacing the underlying word with a neighbouring one, and compared with the following alternative strategies:

Topline: Always selecting the true class of a test sentence as the final prediction, if it was predicted by at least one of the original or the modified sentences.

Weighted majority vote (WMJ): Picking the predicted class with the most occurrences as the final prediction out of the original and the modified test sentences. Here, we assigned a weight of 1 to the original and a weight of $\frac{1}{|S|^\delta}, \delta \in [0, 1]$ to each modified sentence, where $|S|$ is the total number of sentences for the current test input. We experimentally noticed that choices of δ outside $[0, 1]$ did not affect the final prediction.

Average probabilities (AVGP): Averaging the probabilities of the predicted classes for both the original and the modified test sentences, and taking the class with the maximum probability as the final prediction.

[6] This is due to the extensive amount of neighbouring verbs on the annotated-16 and the expanded clusters, which significantly increases the computational cost.

5 Results and Qualitative Study

In this section we present the results obtained from the grid search, and conduct additional experiments that give more insight on the reasons why the proposed methods yield a similar performance to the baseline model.

5.1 Grid Search Experimental Results

After finding the best performing set of hyperparameters amongst all combinations of lexicons, Transformer blocks, and positions that were tested on the development set, we evaluated the corresponding model on the test set. We report the performance for each dataset in terms of micro F_1-score for relation extraction, and accuracy for sentiment analysis[7]. The results for both retrofitting approaches are displayed in Table 1. At first sight, both approaches seem to have no significant impact compared to the baseline performance. More specifically, Method A results in a decrease of performance on all datasets, while Method B slightly improves it for ChemProt and SST-2. Furthermore, we notice that in many cases the alternative strategies we propose work better than our retrofitting approaches. This suggests that i) the use of the lexicons is meaningful, but ii) we have not yet found the correct way of exploiting this knowledge. It is also worth highlighting the abrupt decrease in test performance on the i2b2-2010 for the AVGP method. We assume this is due to the model outputting different probabilities for each of the modified sentences. To confirm this, we compared with the score obtained from WMV for every $\delta \in [0, 1]$ with a step of 0.1, and we observed that for low values of the weight the performance is significantly worse. This indicates that the original sentence is more important than the modified ones, implying in turn that we should assign a higher weight on it. However, in AVGP the averaging equally favours each class, and thus performs poorly.

5.2 Euclidean Distance Ranking of Retrofitted Vectors

In order to understand in greater depth how our proposed methods change the embeddings in space, let us focus on a single test case[8] where the proportion of disagreements between the baseline model and the test case model is statistically significant (based on McNemar's test). This points out that both models behave differently, but on average they result in similar performance. To further analyse how Method A affects the embeddings in the latent space, we randomly select $5,000$ (out of $18,014$) test sentences where we apply our method, and we compute the corresponding activation of the `[CLS]` token before and after retrofitting. Next, we compute the Euclidian distance between every retrofitted vector and every `[CLS]` vector before retrofitting. This results in a 5000×5000 matrix, where

[7] This is the standard choice of metrics for these tasks and datasets [14, 17].

[8] This corresponds to Method A on ChemProt, using the expanded-16 clusters, and retrofitting after layer normalisation at Transformer block 12, with $\alpha = 0.4$ and $\beta = 1.4$ (second row of Table 1).

Table 1: Performance results across all datasets and proposed strategies as well as some retrofitting approaches for static word embeddings. Baseline corresponds to BERT base model finetuned on each dataset for the specific task. Method A, B denote the proposed retrofitting approaches. Topline, AVGP and WMV were discussed in Section 4.5, where for the last we select the weight (δ) based on the best performance on the validation set.

Corpus	**Model**	**Lexicon**	**Dev** miF_1/Acc	**Test** miF_1/Acc
ChemProt	Baseline	–	74.47	72.61
	Method A	expanded-16	74.86	72.56
	Method B	annotated-50	74.59	72.63
	Topline	annotated-50	75.54	73.67
	AVGP	annotated-50	72.92	72.07
	WMV ($\delta = 1.0$)	annotated-50	74.47	72.61
	Chiu et al. [2]	expanded-34	–	71.00
DDI	Baseline	–	71.34	80.11
	Method A	expanded-34	79.35	78.78
	Method B	annotated-34	72.33	79.43
	Topline	annotated-34	73.04	80.97
	AVGP	annotated-34	71.97	79.40
	WMV ($\delta = 0.1$)	annotated-34	72.02	79.60
i2b2-2010	Baseline	–	71.34	72.69
	Method A	expanded-16	72.92	72.52
	Method B	annotated-34	71.83	72.63
	Topline	annotated-34	73.71	74.18
	AVGP	annotated-34	60.79	58.50
	WMV ($\delta = 1.0$)	annotated-34	71.34	72.69
SST-2	Baseline	–	91.86	92.00
	Method B	$\text{WordNet}_{\text{syn}}$	92.09	92.11
	Topline	$\text{WordNet}_{\text{syn}}$	94.95	94.55
	AVGP	$\text{WordNet}_{\text{syn}}$	90.37	90.11
	WMV ($\delta = 1.0$)	$\text{WordNet}_{\text{syn}}$	91.86	92.00
	Faruqui et al. [4]	$\text{WordNet}_{\text{syn}}$	–	82.40

each row contains the distances of one retrofitted vector to all original vectors (before retrofitting). We then rank from $0 - 5000$ each retrofitted embedding by sorting each row in the matrix in ascending order. By doing so, we can check how far our method is moving the embeddings in the latent space. The distribution of the resulting rankings across all vectors is summarized in the histogram in Figure 1. From this plot, we can observe that a large proportion of vectors has a relatively low ranking (around $[0, 80]$), but there is also a considerable amount of vectors with high ranking (around $[950, 1000]$), suggesting that potentially the vectors do not move as far as they should, or sometimes they move too far. This is an indication that there is a lot of variation in the neighbouring embeddings, and therefore not all words in the lexicons are relevant for the task at hand. The following experiment will check if restricting the lexicons to the domain has any impact when retrofitting.

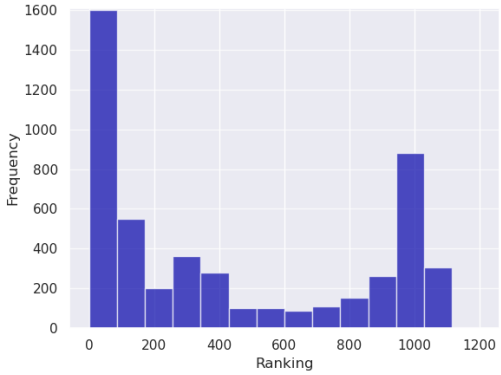

Fig. 1: Histogram of the ranking across `[CLS]` token retrofitted vectors for all 5000 ChemProt test sentences where Method A is applied.

5.3 Neighbouring Word Filtering

Bing Liu's list of adjectives allow us to focus on appropriate words in the semantic lexicons for the task of sentiment analysis. The next question we want to answer is which neighbouring words are relevant for the underlying word, and which are not. It is evident that not all neighbouring words for a given word in the lexicons are actual synonyms in the context of movie reviews. Replacing single words in the input sentence in Method B, forces the same context between the original and the modified sentence. Consequently, we restrict the lexicons to the domain by selecting neighbours that are "good" replacements instead of using the whole list. This is done by inspecting the predictions of BERT for every original and modified sentence on the augmented development set for a given lexicon (see Section 4.5). Then, we can distinguish between the following cases: **(A)** the original sentence was wrongly classified but the modified sentence was correctly classified (good case), **(B)** the original and the modified sentence were correctly/wrongly classified (neutral case), and **(C)** the original sentence was correctly classified but the modified sentence was wrongly classified (bad case).

Next, we compute the counts that correspond to good, neutral and bad cases for every pair of original-neighbouring word. These will show on average if a neighbour is a good replacement or not for a given word. Then, using the McNemar's statistical test, we create three reduced versions, one for each semantic lexicon, by selecting a neighbour for a given word with a 10%, 50% and 90% confidence level[9]. The higher the confidence level the more certain we are about replacing a word by another one, but the smaller the lexicon becomes (and vice versa). Finally, we repeat the grid search optimisation (see Section 4.4) and present in Table 2 the results for the best settings.

[9] We use the confidence level percentage as a subscript to denote the reduced lexicon, e.g., $\text{FrameNet}_{90\%}$.

Table 2: Results for the best performing lexicons derived from our neighbouring word selection for Method B and the proposed alternative strategies. Baseline corresponds to BERT base model, fine-tuned on SST-2 for sentiment analysis.

Lexicon	**Model**	**Dev** *Acc*	**Test** *Acc*
–	Baseline	91.86	92.00
$\text{FrameNet}_{10\%}$	Method B	92.09	92.00
	Topline	92.09	92.11
	AVGP	92.09	92.00
	WMV ($\delta = 0$)	92.09	92.00
$\text{WordNet}_{syn_{10\%}}$	Method B	92.09	92.00
	Topline	92.66	92.00
	AVGP	92.09	91.89
	WMV ($\delta = 0$)	92.09	92.00

Overall, there is some gain in performance compared to the baseline on the development set which is expected. For example, Method B reaches Topline performance for $\text{FrameNet}_{10\%}$, which suggests that retrofitting in the sense of averaging embeddings can be meaningful. Moreover, we can see that the Topline performance is almost identical to that of the baseline model on the test data. This is due to the limited size of the reduced lexicons[10]. Ideally, if the dataset were bigger, we would have selected lexicons with higher confidence level that would also be large enough to improve over the baseline, i.e., the Topline score would significantly outperform the baseline.

6 Conclusion and Future Work

In this paper, we proposed two approaches that extend the original retrofitting technique to operate with contextualized embedding systems. More precisely, we incorporated external knowledge coming from semantic lexicons into BERT contextualized representations. After conducting a large-scale series of experiments on three biomedical datasets for relation extraction, and one movie review dataset for sentiment analysis, we observe that both approaches do not substantially affect the performance on these downstream tasks. Our test results show that the lexicons can be a useful source of information to further improve the results. However, the current experimental setting did not make it viable. This is demonstrated in our qualitative study, where we show that when we improve the quality of the semantic lexicons by selecting only relevant neighbours for a given word, the resulting lexicons are not sufficiently large to be able to generalize at test time. In the future, we plan to experiment with more fine-grained tasks where we are certain about the knowledge source, and where we would not need to heavily depend on word statistics to apply the proposed method.

[10] For example FrameNet originally consists of 1700 words and 90140 relations, while its largest reduced version, $\text{FrameNet}_{10\%}$, has only 1 word and 5 relations.

References

1. Bengio, Y., *et al.*: Label propagation and quadratic criterion. In: Semi-Supervised Learning. pp. 193–216. MIT Press (2006)
2. Chiu, B.*et al.*.: Enhancing biomedical word embeddings by retrofitting to verb clusters. In: BioNLP. pp. 125–134 (2019)
3. Devlin, J., *et al.*: BERT: Pre-training of deep bidirectional transformers for language understanding. In: HLT-NAACL. pp. 4171–4186 (2019)
4. Faruqui, M., *et al.*: Retrofitting word vectors to semantic lexicons. In: NAACL HLT. pp. 1606–1615 (2015)
5. Ferret, O.: Turning distributional thesauri into word vectors for synonym extraction and expansion. In: IJCNLP. pp. 273–283 (2017)
6. Ferret, O.: Turning distributional thesauri into word vectors for synonym extraction and expansion (2017)
7. Hu, M., Liu, B.: Mining and summarizing customer reviews. pp. 168–177 (2004)
8. Lee, J., *et al.*: Biobert: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics **36**(4), 1234–1240 (2020)
9. Lengerich, B., *et al.*: Retrofitting distributional embeddings to knowledge graphs with functional relations. In: COLING. pp. 2423–2436 (2018)
10. Levy, O., *et al.*: Dependency-based word embeddings. In: ACL. pp. 302–308 (2014)
11. Mikolov, T., *et al.*: Distributed representations of words and phrases and their compositionality. In: NIPS. pp. 3111–3119 (2013)
12. Mrkšić, N., *et al.*: Counter-fitting word vectors to linguistic constraints. arXiv preprint arXiv:1603.00892 (2016)
13. Mrkšić, N., *et al.*: Semantic specialization of distributional word vector spaces using monolingual and cross-lingual constraints. TACL **5**, 309–324 (2017)
14. Peng, Y., *et al.* biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. In: BioNLP. pp. 58–65 (2019)
15. Peters, M.E.*et al.*.: Knowledge enhanced contextual word representations. arXiv preprint arXiv:1909.04164 (2019)
16. Shi, W., *et al.* contextualized word embeddings with paraphrases. In: EMNLP-IJCNLP. pp. 1198–1203 (2019)
17. Socher, R., *et al.*C.D., compositionality over a sentiment treebank. In: EMNLP. pp. 1631–1642 (2013)
18. Vaswani, A., *et al.*: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017)
19. Yang, B., *et al.*: Leveraging knowledge bases in LSTMs for improving machine reading. In: ACL. pp. 1436–1446 (2017)

Acknowledgements

This research was partially supported by the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement No. 952215 TAILOR and by the Inria Project Lab "Hybrid Approaches for Interpretable AI" (HyA-IAI). Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).