Representing Prior Knowledge Using Randomly, Weighted Feature Networks for Visual Relationship Detection

Jinyung Hong[1], **Theodore P. Pavlic**[1,2,3,4]

[1] School of Computing and Augmented Intelligence, Arizona State University, Tempe, AZ 85281, USA
[2] School of Sustainability, Arizona State University, Tempe, AZ 85281, USA
[3] School of Complex Adaptive Systems, Arizona State University, Tempe, AZ 85281, USA
[4] School of Life Sciences, Arizona State University, Tempe, AZ 85281, USA
jhong53@asu.edu, tpavlic@asu.edu

Abstract

The single-hidden-layer Randomly Weighted Feature Network (RWFN) introduced by Hong and Pavlic (2021) was developed as an alternative to neural tensor network approaches for relational learning tasks. Its relatively small footprint combined with the use of two randomized input projections – an insect-brain–inspired input representation and random Fourier features – allow it to achieve rich expressiveness for relational learning with relatively low training cost. In particular, when Hong and Pavlic compared RWFN to Logic Tensor Networks (LTNs) for Semantic Image Interpretation (SII) tasks to extract structured semantic descriptions from images, they showed that the RWFN integration of the two hidden, randomized representations better captures relationships among inputs with a faster training process even though it uses far fewer learnable parameters. In this paper, we use RWFNs to perform Visual Relationship Detection (VRD) tasks, which are more challenging SII tasks. A zero-shot learning approach is used with RWFN that can exploit similarities with other seen relationships and background knowledge – expressed with logical constraints between subjects, relations, and objects – to achieve the ability to predict triples that do not appear in the training set. The experiments on the Visual Relationship Dataset to compare the performance between RWFNs and LTNs, one of the leading Statistical Relational Learning frameworks, show that RWFNs outperform LTNs for the predicate-detection task while using fewer number of adaptable parameters ($1:56$ ratio). Furthermore, background knowledge represented by RWFNs can be used to alleviate the incompleteness of training sets even though the space complexity of RWFNs is much smaller than LTNs ($1:27$ ratio).

Introduction

Semantic Image Interpretation (SII) (Neumann and Möller 2008) is a task of automatically extracting high-level information about the content of a visual scene. This information consists of the objects in the scene, their attributes, and the relations among them. Formally, the result of applying SII on an image is called a *scene graph* (Krishna et al. 2017): the labeled nodes are regarded as objects in the scene and their attributes, the labeled edges indicate the relations between the corresponding nodes. The resulting scene graphs can be applied to many application domains, such as visual question answering, image captioning, image querying, and robot interaction.

Visual Relationship Detection (VRD) tasks are a special case of scene-graph construction, and a visual relationship can be readily represented as a triple of the form $\langle s, p, o\rangle$ where s and o indicate the subject and the object, respectively, that are the semantic classes or labels of two bounding boxes in the image. The predicate p is the label representing the relationship between the two bounding boxes. Therefore, constructing a scene graph in a visual scene is the task of correctly labeling subjects and objects nodes in the graph and labeling edges connected from subject to object.

In VRD tasks, visual relationships are mainly detected in a supervised fashion, but there are some challenges. For one, due to the enormous human effort of the annotators, a comprehensive and detailed annotation is not achievable. Furthermore, even when a training set covers a variety of different relationships, a new type of relationship may appear only in the test set. We use *zero-shot learning* (Lampert, Nickisch, and Harmeling 2013) to address these VRD issues. The zero-shot approach can be achieved by exploiting the similarity with the triples in the training set or a high-level description of the relationship. This is closer to human learning for supervised learning. Indeed, humans can both generalize from seen or similar examples and use their background knowledge to identify never seen relationships (Lampert, Nickisch, and Harmeling 2013).

A zero-shot training approach using Logic Tensor Networks (LTN) (Donadello and Serafini 2019) has been proposed as one of the state-of-the-art methods for detecting invisible visual relationships. The LTN (Serafini and Garcez 2016) is a Statistical Relational Learning framework that learns from relational data in the presence of additional logical constraints. So, Donadello and Serafini (2019) leveraged LTNs to exploit the similarities with already seen triples. They showed that the results on the Visual Relationship Dataset jointly using logical knowledge and data outperform the state-of-the-art approaches based on data or linguistic knowledge and that logical knowledge can compensate for the incompleteness of the datasets due to the high annotation effort.

Recently, a more economical approach for representing background knowledge has been proposed that is in-

Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

spired by the neural architecture of the insect brain. Insect neuroscience has shown that insects express sophisticated and complex behaviors although they possess central nervous systems far smaller than the human brain (Avarguès-Weber, Deisig, and Giurfa 2011). Specifically, it has been shown that the honey bee brain includes high levels of cognitive sophistication to learn relational concepts such as "same," "different," "larger than," "better than," among others (Avarguès-Weber and Giurfa 2013). Based on those observations, Hong and Pavlic (2021) proposed a novel insect-brain–inspired neural network, Randomly Weighted Feature Network (RWFN), for relational embedding that incorporates randomly drawn, untrained weights in its encoder with a trained linear model as a decoder. Their approach mimics the randomized, mostly feedforward architecture of the insect brain that projects olfactory features from the antennal lobe (AL) randomly to the mushroom body (MB), the main center of higher-order learning, that are then decoded by downstream neuropils that translate information processing into action. In addition to the insect-inspired architecture, RWFN also leverages random Fourier feature (Rahimi and Recht 2007), a kernel approximation method that overcomes the issues of conventional kernel machines or kernel methods (Smola and Schölkopf 1998) to concisely and efficiently compute linear interactions among inputs. When RWFNs were applied to one of the SII tasks defined by Donadello, Serafini, and Garcez (2017) to demonstrate the performance of LTNs, the RWFNs were able to effectively learn the *part-of* relation among inputs better than the neural tensor network in LTNs. In addition, Hong and Pavlic (2021) used an ablation study to show that the two randomized representations in RWFNs can be compensated for each other to achieve balanced performance among different kinds of dataset.

In this paper, we extend RWFNs to address zero-shot learning for the detection of unseen visual relationships. We perform experiments on the Visual Relationship Dataset (VRD) (Lu et al. 2016), a complex dataset containing 100 unary relationships and 70 binary relationships, and compare the performance between RWFNs and LTNs.

Related Work

Methods for Visual Relationship Detection

In order to extract a scene graph from images, several works that utilize axioms using fuzzy logic (Hájek 2013) have been proposed because fuzzy logic can handle the intrinsic noise from the object detector. Specifically, there are algorithms for building SII graphs with a fuzzy logic ontology of spatial relations (Atif, Hudelot, and Bloch 2013; Hudelot, Atif, and Bloch 2008), an iterative message passing algorithm where the information about the objects maximizes the likelihood of the relationships (Xu et al. 2017), and the combination with Long Short-Term Memories (LSTMs) for encoding the context given by the detected bounding boxes (Zellers et al. 2018). However, some of them are limited to only the spatial relationships and based on the implausible assumptions in real-work applications, for example, predicates must be mutually exclusive.

There are several methods using a Conditional Random Field (CRF) that encodes a fully connected graph and that labels or discards then nodes and edges by minimizing an energy function (e.g., Kulkarni et al. 2013; Chen, Zhou, and Prasanna 2012; Chen et al. 2014). However, some of them do not consider logical knowledge or use hand-crafted logical constraints, which are difficult to extend to other types of constraints.

Deep learning methods are also exploited for the task. Specifically, the detection of visual relationships using a Deep Relational Network (Dai, Zhang, and Lin 2017), a deep reinforcement learning model for detecting relationships and attributes (Liang, Lee, and Xing 2017), a message passing algorithm for sharing subject–object–predicate information among neural networks (Li et al. 2017), and an end-to-end system that can exploit the interaction of visual and geometric features of the subject, object and predicate (Yin et al. 2018). However, the above systems cannot utilize the visual/geometric features of the subject/object and additional background knowledge together.

A joint embedding with visual knowledge can exploit background knowledge. Implication, mutual exclusivity, and *type-of* are exploited as logical constraints by Ramanathan et al. (2015), whereas a word embedding of the subject/object labels is used as background knowledge by Lu et al. (2016). Yu et al. (2017) use background knowledge in the form of a probability distribution of a relationship given the subject/object. However, the above methods do not exploit any type of logical constraints.

LTN can compensate for the shortcomings of all the studies mentioned above. LTNs allow multiple edges between nodes and can exploit the integration of the visual/geometric features of the subject/object with additional background knowledge encoded by logical constraints, which none of the above works can include. Therefore, we mainly compare the performance of RWFN to LTN, which we describe in detail next.

Logic Tensor Networks

Here, we describe basic concepts underlying Logic Tensor Networks (LTNs) (Serafini and Garcez 2016). Although their structure is fundamentally different, RWFNs and LTNs use the same approach for mapping logical symbols to numerical values and learning reasoning relations among real-valued vectors using the logical formulas. However, LTNs combine reasoning with first-order logic with (in contrast to WRFN) learning based on Neural Tensor Netowrks (NTNs) (Socher et al. 2013). Consequently, the LTN framework can be implemented in TensorFlow (Badreddine et al. 2021).

We first define a First-Order-Logic (FOL) language $\mathcal{L}$ and its signature as containing three disjoint sets: i) $\mathcal{C}$ (constants), ii) $\mathcal{F}$ (functions), and iii) $\mathcal{P}$ (predicate). However, we will not specify the function symbols $\mathcal{F}$ in detail because they are not used in the tasks. For specifying an *arity* for a predicate symbol s, we use the notation as $\alpha(s)$. The logical formulas in $\mathcal{L}$ are used to describe relational knowledge. The objects in FOL are mapped to an interpretation domain $\subseteq \mathbb{R}^n$ so that every object is associated with an

n-dimensional vector of real numbers and is reasoned over FOL. Intuitively, this n-tuple symbolizes n numerical features of an object, and predicates are represented as fuzzy relationships on real vectors. With this numerical background, we can now establish the numerical *grounding* of FOL with the following semantics. The term *grounding* is used as a synonym of logical interpretation in a real world and needs to capture the latent correlation between the features of objects and their categorical or relational properties.

Let $n \in \mathbb{N}$. An n-*grounding*, or simply *grounding*, $\mathcal{G}$ for a FOL $\mathcal{L}$ is a function defined on the signature of $\mathcal{L}$ satisfying the following conditions:

- $\mathcal{G}(c) \in \mathbb{R}^n$ for every constant symbol $c \in \mathcal{C}$
- $\mathcal{G}(P) \in \mathbb{R}^{n \cdot \alpha(f)} \to [0,1]$ for predicate sym. $P \in \mathcal{P}$

Given a grounding $\mathcal{G}$, we can define the semantics of closed terms and atomic formulas as follows:

$$\mathcal{G}(P(t_1, \ldots, t_m)) \triangleq \mathcal{G}(P)(\mathcal{G}(t_1), \ldots, \mathcal{G}(t_m))$$

The semantics for connectives, such as $\mathcal{G}(\neg\phi), \mathcal{G}(\phi \wedge \psi), \mathcal{G}(\phi \vee \psi)$, and $\mathcal{G}(\phi \to \psi)$, can be computed by following the fuzzy logic such as the Lukasiewicz t-norm (Bergmann 2008).

The grounding of an m-ary predicate P, namely $\mathcal{G}(P)$, is defined as a generalization of the NTN (Socher et al. 2013), as a function from $\mathbb{R}^{mn}$ to $[0,1]$, as follows:

$$\mathcal{G}_{\text{LTN}}(P)(\mathbf{v}) = \sigma(u_P^\top \mathtt{f}(\mathbf{v}^\top W_P^{[1:k]} \mathbf{v} + V_P \mathbf{v} + b_P)) \quad (1)$$

where $\mathbf{v} = \langle \mathbf{v}_1^\top, \ldots, \mathbf{v}_m^\top \rangle^\top$ is the mn-ary vector obtained by concatenating each $\mathbf{v}_i$. σ is the sigmoidal logistic function, and $\mathtt{f}$ is the hyperbolic tangent ($\tanh$). The parameters for P are: $W_P^{[1:k]}$, a 3-D tensor in $\mathbb{R}^{k \times mn \times mn}, V_P \in \mathbb{R}^{k \times mn}, b_P \in \mathbb{R}^k$ and $u_P \in \mathbb{R}^k$. Because the RWFN model is a novel way of grounding a predicate as $\mathcal{G}_{\text{RWFN}}(P)$, we can directly compare the performance of RWFNs for the visual relationship detection tasks with LTNs.

The optimization of the truth values of the formulas in a LTN's knowledge base is directly involved with learning the groundings, i.e. *grounded theory*. A *partial grounding* $\hat{\mathcal{G}}$ is a grounding that can be defined on a subset of the signature of $\mathcal{L}$. A grounding $\mathcal{G}$ is said to be a completion of $\hat{\mathcal{G}}$ if $\mathcal{G}$ is a grounding for $\mathcal{L}$ and coincides with $\hat{\mathcal{G}}$ on the symbols where $\hat{\mathcal{G}}$ is defined. Let *GT* be a grounded theory which is a pair $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$ with a set $\mathcal{K}$ of closed formulas and a partial grounding $\hat{\mathcal{G}}$. A grounding $\mathcal{G}$ satisfies a *GT* $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$ if $\mathcal{G}$ completes $\hat{\mathcal{G}}$ and $\mathcal{G}(\phi) = 1$ for all $\phi \in \mathcal{K}$. A *GT* $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$ is satisfiable if there exists a grounding $\mathcal{G}$ that satisfies $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$. That is, deciding the satisfiability of $\langle \mathcal{K}, \hat{\mathcal{G}} \rangle$ amounts to searching for a grounding $\mathcal{G}$ such that all the formulas of $\mathcal{K}$ are mapped to 1. If a *GT* is not satisfiable, the best possible satisfaction that we can reach with a grounding is of our interest.

Therefore, the best-satisfiability problem is an optimization problem on the set of LTN parameters $\Theta \triangleq \{W_P, V_P, b_P, u_P | P \in \mathcal{P}\}$ from Eq. (1) to be learned. We use notation $\mathcal{G}(\cdot|\Theta)$ to indicate the grounding with grounding-function parameters equal to Θ. Then, we define the best-satisfiability problem as finding the best parameter set:

$$\Theta^* \triangleq \arg\max_{\Theta} \mathcal{G}(\wedge_{\phi \in \mathcal{K}} \phi | \Theta) - \lambda \|\Theta\|_2^2, \quad (2)$$

where $\lambda \|\Theta\|_2^2$ is a regularization term.

Randomly Weighted Feature Networks

The Randomly Weighted Feature Network (RWFN) (Hong and Pavlic 2021) is a single-hidden-layer neural network that incorporates randomly drawn, untrained weights in an encoder that uses an adapted linear model as a decoder. A key characteristic of RWFN is the generation of unique hidden representations through the integration of two randomized methods for projecting from the input space to a higher-order space. Consequently, RWFNs can efficiently learn the degree of relationship among inputs by training only a linear decoder model. In addition, because all the weights of an encoder in the model are randomly drawn, the encoder can be shared with other classifiers that allow training only a decoder, which drastically reduces the space complexity of the model.

Unique Hidden Representation in RWFN

Insect-brain–inspired Representation. In the insect brain, processed olfactory, visual, and mechanosensory stimuli are conveyed to the *Mushroom Body (MB)* (Mobbs 1982), which can be viewed as the critical region responsible for multimodal associative learning (Menzel 2001). In the fruit-fly brain, each of thousands of Kenyon Cells (KCs) in the MB takes input from a random set of ∼7 inputs from Input Neurons (INs) (Caron et al. 2013; Inada, Tsuchimoto, and Kazama 2017). Thus, a simplified neural circuit modeling the MB is a neural network with three layers consisting of: i) INs that provide extracted features from olfactory, visual, and mechanosensory inputs, ii) KCs generating the sparse-encoding of sensory stimuli, and iii) mushroom body Extrinsic Neurons (ENs) for activating several different behavioral responses (Cope et al. 2018). Hong and Pavlic (2021) focused on the olfactory pathway between features extracted and coded by the AL and KCs in the MB (Cope et al. 2018; Peng and Chittka 2017); their RWFN architecture mimics the input transformation of odorant representation in the AL to the higher-order representation across the KCs in the MB.

Random Fourier Features. Kernel machines have attracted significant interest due to their capability of approximating functions with excellent performance for detecting decision boundaries given enough training data. These methods leverage transformations, via a lifting function ϕ, that allows better discrimination among different inputs. Given dataset vector inputs $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, the kernel function $k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$ represents the similarity (i.e., inner product) between $\mathbf{x}$ and $\mathbf{y}$ in the ϕ-transformed space. However, because of the potential complexity of the transformation ϕ, learning the kernel function k may suffer significant inefficiency of computational and storage costs.

To mitigate these kernel-machine costs, random Fourier features (Rahimi and Recht 2007) provide a data transfor-

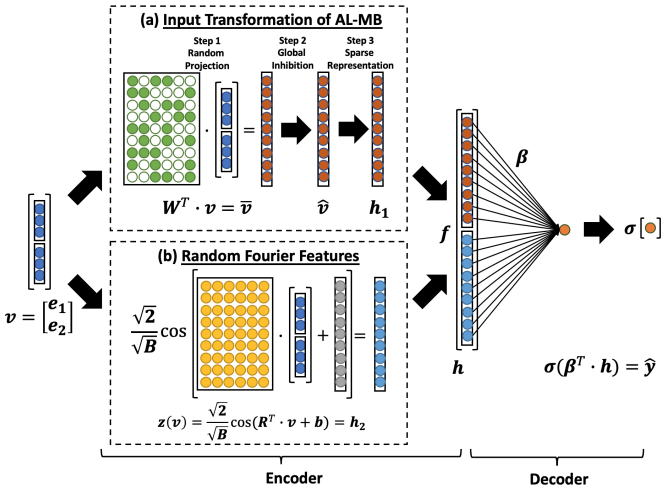

Figure 1: Visualization of the structure of the Randomly Weighted Feature Network. In the depicted case, the input vector $\mathbf{v}$ constitutes of two entities, $e_1, e_2 \in \mathbb{R}^3$ and it shows to learn a binary relation between them (e_1, R, e_2), such as (Cat, hasPart, Tail) (Hong and Pavlic 2021)

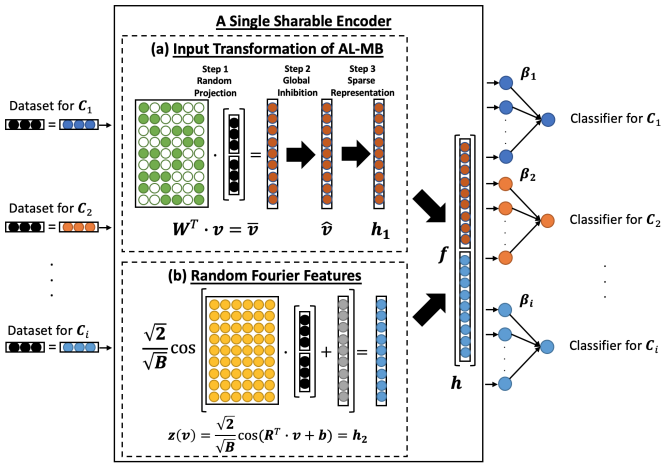

Figure 2: Visualization of the structure of the Randomly Weighted Feature Network with weight sharing. In the case of learning each classifier from the class $\mathcal{C}_1$ to the class $\mathcal{C}_i$, RWFNs allow us to use the same encoder to extract features from each data from the class $\mathcal{C}_1$ to the class $\mathcal{C}_i$ (Hong and Pavlic 2021)

mation that permits a far less expensive approximation of the kernel function. For each vector input $\mathbf{x} \in \mathbb{R}^d$, the method applies a randomized feature function $\mathbf{z} : \mathbb{R}^d \to \mathbb{R}^D$ (generally, $D \gg d$ with sample size $N \gg D$) that maps $\mathbf{x}$ to evaluations of D random Fourier bases from the Fourier transform of kernel k. In this transformed space, linear operations can be used to approximate kernel evaluations, as in:

$$k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle \approx \mathbf{z}(\mathbf{x})^\top \mathbf{z}(\mathbf{y}) \tag{3}$$

Therefore, by transforming the input with $\mathbf{z}$, fast linear learning can be used to approximate the evaluations of nonlinear kernel machines. Hong and Pavlic (2021) used random Fourier features as latent representations that reduce the complexity of learning relations among real-valued entities.

Model Architecture

Let the input vector $\mathbf{v} = [\mathbf{v}_1^\top, \ldots, \mathbf{v}_m^\top]^\top$ be the mn-ary vector where m is arity and n is the input dimension, and consider a hidden node $j \in \{1, ..., B\}$. To mimic the random AL–MB connections in the insect brain (Hong and Pavlic 2021), hidden node j is associated with a random subset $\mathcal{S}_j \subset \{1, ..., mn\}$ of $N_{in} = 7$ of the mn inputs (i.e., $|\mathcal{S}_j| = N_{in} = 7$). Furthermore, hidden node j takes as input a binary-weighted combination of input nodes so that $w_{j,i} = 1$ if $i \in S_j$ and $w_{j,i} = 0$ otherwise. So, the input nodes are effectively gated by the weights on each hidden node, and the weight matrix $\mathbf{W} \in \mathbb{R}^{mn \times B}$ in this computation is random, binary, and sparse. For the jth hidden node, $\bar{v}_j$ represents the corresponding binary-weighted sum.

Next, for each hidden node j, a post-processing step produces the jth intermediate output $\hat{v}_j$ by subtracting the average of the binary-weighted sums, as in:

$$\hat{v}_j = \bar{v}_j - \frac{1}{B} \sum_{i=1}^{B} \bar{v}_i \tag{4}$$

where B is the number of hidden units. Therefore, the sparse output of the j-th KC node can be defined as $h_j^{(1)} = g(\hat{v}_j)$ where g is the ReLU function (Glorot, Bordes, and Bengio 2011) that allows the model to produce sparse hidden output. By doing so, the output vector is defined as $\mathbf{h}_1 = [h_1^{(1)}, \ldots, h_B^{(1)}]^\top$.

On the other hand, to generate random Fourier features, the randomized feature function $\mathbf{z}(\cdot)$ (Rahimi and Recht 2007; Sutherland and Schneider 2015) was used so the inputs are projected as follows:

$$\mathbf{h}_2 = \mathbf{z}(\mathbf{v}) = (\sqrt{2}/\sqrt{B}) \cos(\mathbf{R}^\top \mathbf{v} + \mathbf{b}) \tag{5}$$

where $\mathbf{R} \sim \mathcal{N}^{mn \times B}(0, 1)$ and $\mathbf{b} \sim U^B(0, 2\pi)$, which is Gaussian kernel approximation. Consequently, the output vector $\mathbf{h}_2$ can be considered as another latent representation of relationship among input.

Finally, using the above two latent representations, the RWFNs can be defined as a function from $\mathbb{R}^{mn}$ to $[0, 1]$:

$$\mathcal{G}_{\text{RWFN}}(P)(\mathbf{v}) = \sigma\left(\boldsymbol{\beta}^\top \mathbf{h}\right) = \sigma\left(\boldsymbol{\beta}^\top \mathtt{f}\left(\begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \end{bmatrix}\right)\right) \tag{6}$$

where $\mathbf{h}$ is the final hidden representation obtained by applying (for numerical stabilization) the hyperbolic tangent ($\tanh$) function $\mathtt{f}$ to the concatenation of $\mathbf{h}_1$ and $\mathbf{h}_2$, and σ is the sigmoidal logistic function. Because RWFN requires to adapt only $\boldsymbol{\beta} \in \mathbb{R}^{2B}$, it possess a faster learning process with fewer parameters compared to LTNs. Fig. 1 shows a visualization of the model structure.

RWFNs with Weight Sharing

Because of the above characteristics of the model, the randomized encoder of an RWFN can serve as a shared resource for multiple relatively simple (i.e., linear) downstream decoders trained for different classifiers. Hong and

Pavlic (2021) called this property as *weight sharing*. Fig. 2 shows a visualization of the structure of RWFN applied with weight sharing to the learning of i different classifiers. Hong and Pavlic show that RWFN with weight sharing allows RWFN to drastically reduce space complexity because the most costly components of the RWFN (i.e., the input and hidden layers) can be reused across multiple classifiers instead of being implemented (and trained) in parallel.

RWFN for Visual Relationship Detection

Donadello and Serafini (2019) defined how to encode the problem of detecting visual relationship with LTNs. Because RWFNs can be readily applied using the same problem encodings of LTNs, we re-use the formalization of the problem from Donadello and Serafini for visual relationship detection in RWFNs. In this section, we briefly introduce that formalization.

The Knowledge base $\mathcal{K}_{\text{SII}}$. Given a dataset of images, let $B(p)$ be the corresponding set of bounding boxes of an image p in the dataset. Each bounding box in $B(p)$ has its annotations about a set of labels that describe the contained physical object, and pairs of bounding boxes have their annotations about the semantic relations between the contained physical objects. Let $\Sigma_{\text{SII}} = \langle \mathcal{P}, \mathcal{C} \rangle$ be the signature where $\mathcal{P} = \mathcal{P}_1 \cup \mathcal{P}_2$ is the set of predicates. $\mathcal{P}_1$ is a set of unary predicates indicating the *object types* or *semantic classes* for the label of bounding boxes. $\mathcal{P}_2$ includes binary predicates for the label of pairs of bounding boxes. Let $\mathcal{C} \triangleq \bigcup_p B(p)$ be the set of constants of all bounding boxes in the given dataset. A grounded theory can be defined as $\mathcal{T}_{\text{SII}} \triangleq \langle \mathcal{K}_{\text{SII}}, \hat{G}_{\text{SII}} \rangle$.

The knowledge base $\mathcal{K}_{\text{SII}}$ is for encoding the bounding box annotations in the dataset and some background knowledge about the domain so that it contains positive and negative examples (used for learning the grounding of the predicates in $\mathcal{P}$) and the background knowledge. The positive examples for a semantic class C are the atomic formulas $C(b)$ for every bounding box b labelled with class C. Regarding the negative examples, for a semantic class C, we consider the atomic formulas $\neg C(b)$ for every bounding box b not labelled with C. Regarding the background knowledge, Donadello and Serafini (2019) manually defined the logical constraints, such as *negative domain and range constraints*, by referring to on-line linguistic resources such as FrameNet (Baker, Fillmore, and Lowe 1998) and VerbNet (Schuler 2005) that provide the range and domain of binary relations through the so-called frames data structure.

Definition of New Features. On top of the features for grounding constants from Donadello, Serafini, and Garcez (2017), an extra set of features are added for representing constants in the knowledge base, called *joint features*. Joint features include quantities such as: the inclusion ration of two bounding boxes, the area of intersection of two bounding boxes, the Euclidean distance between the centroids of two bounding boxes, and others that allow for better capturing the geometric interactions between two bounding boxes.

New Optimization for Knowledge Base. Equation (2) defines how to learn the LTN parameters by maximizing the grounding of the conjunctions of the formulas in the knowledge base. However, there is a need to mitigate the issues that many t-norms have, such as leading the knowledge base satisfiability to zero, getting stuck in local optima, and underflow problems (Donadello and Serafini 2019). Consequently, a mean operator was added to Eq. (2) to return a global satisfiability of the knowledge base $\mathcal{K}_{\text{SII}}$ as follows:

$$\Theta^* = \arg\max_{\Theta} \text{mean}_p(\hat{\mathcal{G}}_{\text{SII}}(\phi|\Theta)|\phi \in \mathcal{K}_{\text{SII}}) - \lambda\|\Theta\|_2^2 \quad (7)$$

with $p \in \mathbb{Z}$.

Post Processing. After a grounded theory $\mathcal{T}_{\text{SII}}$ is learned, the set of groundings $\hat{\mathcal{G}}_{\text{SII}}(r(b,b'))_{r\in\mathcal{P}_2}$ with a new pair of bounding boxes $\langle b, b' \rangle$ are computed. Then, every resulting grounding $\hat{\mathcal{G}}_{\text{SII}}(r(b,b'))$ is multiplied with the frequency of the predicate in the training set. Furthermore, equivalences between the binary predicates are exploited in order to normalize the groundings.

Experiments

In our experiments[1], we use the Visual Relationship Dataset (VRD) (Lu et al. 2016), including 4000 images for the training set and 1000 for the testing set annotated with visual relationships. Bounding boxes are annotated with labels containing 100 unary predicates. These labels are for indicating animals, vehicles, clothes, and generic objects. Pairs of bounding boxes are annotated with labels including 70 binary predicates. These labels represent actions, prepositions, spatial relations, comparatives, or preposition phrases. The dataset contains 37993 instances of visual relationships and 6672 types of relationships. 1877 relationships occur only in the test set and are used to evaluate the zero-shot learning scenario.

Methods

VRD Tasks. The performances of both RWFNs and LTNs are tested on the following VRD standard tasks.

- *Phrase detection.* This task predicts a correct triple $\langle s, p, o \rangle$ and its location in a single bounding box that contains both the subject and the object. If the labels are the same as the ground truth triple and the predicted bounding box has at least 50% overlap with a corresponding bounding box in the ground truth, the triple is a true positive. The ground truth bounding box means the union of the ground truth bounding boxes of the subject and the object.
- *Relationship detection.* This task predicts a correct triple/relationship and the bounding boxes that include the subject and the object of their relationship. If both bounding boxes overlap at least 50% of the corresponding ones in the ground truth, the triple is a true positive. In addition, the labels for the predicted triple must match with the corresponding ones in the ground truth.

[1]All the source codes, models, and figures are available on https://github.com/PavlicLab/AAAI2022-CLeaR2022-Visual_Relationship_Detection-RWFN.

- *Predicate detection.* For a given set of bounding boxes, this task predicts a set of correct binary predicates between them. Because the prediction does not depend on the performance of an object detector, the performances of LTNs/RWFNs are determined by their abilities to predict binary predicates, which is our interest.

Comparison of RWFNs and LTNs. The task is to complete the partial knowledge in the dataset by finding a grounding $\mathcal{G}^*_{\text{SII}}$. Therefore, for LTNs, it extends $\hat{\mathcal{G}}_{\text{LTN}}$ using Eq. (1) such that:

$$\mathcal{G}^*_{\text{SII}}(\mathcal{C}(b)) \to [0,1] \quad \text{and} \quad \mathcal{G}^*_{\text{SII}}(\mathcal{R}(b_1, b_2)) \to [0,1]$$

for every unary predicate $\mathcal{C}$ and binary predicate $\mathcal{R}$ and every (pair of) bounding box in the dataset. Because for RWFNs $\hat{\mathcal{G}}_{\text{RWFN}}$ using Eq. (6) is instead used for grounding unary and binary predicates, we can directly compare the performances between LTNs and RWFNs.

Donadello and Serafini (2019) test the performance of LTNs with two grounded theories, $\mathcal{LTN}^{\text{expl}}$ and $\mathcal{LTN}^{\text{prior}}$. The first one indicates $\mathcal{LTN}^{\text{expl}} = \langle \mathcal{K}_{\text{expl}}, \hat{\mathcal{G}}_{\text{LTN}} \rangle$ where $\mathcal{K}_{\text{expl}}$ includes only positive and negative examples for predicates predictions. The second grounded theory means $\mathcal{LTN}^{\text{prior}} = \langle \mathcal{K}_{\text{prior}}, \hat{\mathcal{G}}_{\text{LTN}} \rangle$ where $\mathcal{K}_{\text{prior}}$ contains examples as well as logical constraints. Therefore, we can check the contribution of using logical constraints. By referring to the above setting of LTNs, the performance of RWFNs can also be evaluated by creating two grounded theories, $\mathcal{RWFN}^{\text{expl}}$ and $\mathcal{RWFN}^{\text{prior}}$. Furthermore, we tested RWFNs with weight sharing by building two grounded theories, $\mathcal{RWFN}_{\mathcal{WS}}{}^{\text{expl}}$ and $\mathcal{RWFN}_{\mathcal{WS}}{}^{\text{prior}}$. In the setting for RWFNs, all predicates have their own encoders whereas in the setting for RWFNs with weight sharing, we created one predefined encoder for unary predicate and another for binary predicate in advance, and those encoders were used as the shared encoders for grounding predicates. Thus, we can readily confirm how the shareable encoders contribute to achieve the performance.

We first train all the models – RWFNs, RWFNs with weight sharing, and LTNs – on the VRD training set, and then we test them on the VRD test set. All the models have been evaluated their abilities to generalize to the 1877 relationships never seen in the training phase.

Evaluation Metric. Following Donadello and Serafini (2019), for each image in the test set, we use the grounded theories for each model to compute the ranked set of groundings with a pair of bounding boxes computed with an object detector (the R-CNN model from Lu et al. (2016)) or taken from the ground truth (for the predicate detection). We then use the recall@100/50 (Lu et al. 2016) as evaluation metrics because the annotation is not complete, and precision would wrongly penalize true positives. In addition, every pair of bounding boxes with all the binary predicates are classified as many predicates can occur between two objects, and it is not always possible to define a preference between predicates. This choice is counterbalanced by predicting the correct relationships within the top 100 and 50 positions.

Hyperparameter Setting. For hyperparameter setting of LTNs, we followed the setting of Donadello and Serafini (2019). In equation (7), p was set to -1 (harmonic mean). The chosen t-norm is the Lukasiewicz one. The number of tensor layers in Eq. (1) is set as $k = 5$ and $\lambda = 10^{-10}$ in Eq. (7). In order to mimic the results from Donadello and Serafini, the optimization is performed separately on $\mathcal{LTN}^{\text{expl}}$ and $\mathcal{LTN}^{\text{prior}}$ with 10000 training epochs of the RMSProp optimizer (Tieleman and Hinton 2012) in TensorFlow (Abadi et al. 2016).

For hyperparameter setting of RWFNs and RWFNs with weight sharing, the value of p for harmonic mean, t-norm, λ and the number of training epochs are set as above. In Eq. (6), the size of B is set to 500 for unary predicate whereas it is set to 1000 for binary predicate. In contrast with Hong and Pavlic (2021), in order to train RWFNs and RWFNs with weight sharing, we use the Follow-The-Regularized-Leader (FTRL) optimizer (McMahan et al. 2013) instead of the RMSProp optimizer. It has been known that the FTRL optimizer is suitable for shallow models with large and sparse feature spaces, and this is properly applicable to optimize RWFNs because the feature representations in RWFNs can be sparse due to the insect-brain–inspired representation. For the hyperparameters of the FTRL optimizer, the learning rate is set to 1, and g, λ_1 and λ_2 are set to the standard values that TensorFlow provides, where are $-0.5, 0$, and 0, respectively. See the Appendix for additional experimental details.

Results

Table 1 shows the results of comparison between LTNs, RWFNs, and RWFNs with weight sharing. We performed the training 5 times obtaining 5 models for $(\cdot)^{\text{expl}}$ and $(\cdot)^{\text{prior}}$, respectively. For each task and each grounded theory, we report 95% confidence intervals (CIs) on the means for the sample results given by these models.

Most differences in performance between RWFN variants (with and without weight sharing) and LTNs are marginal. Across all tasks, whenever LTNs were shown to have higher sample mean performance, LTN CIs overlapped with CIs of both RWFNs variants, indicating no clear significant difference in performance. For predicate detection, RWFNs variants did show significantly better performance than LTNs. In particular, without the use of logical constraints, RWFNs and RWFNs with weight sharing achieved better performance for detecting predicates than LTNs with logical constraints. This indicates the randomized projections of inputs in RWFNs effectively capture and learn the visual relationships among inputs. Furthermore, RWFNs with weight sharing do not suffer a deterioriation in performance. In fact, making use logical constraints in RWFNs with weight sharing may be more beneficial than in RWFNs because the gap of the performance of the model with and without logical constraints is larger than the original RWFNs.

Relative Complexity of RWFNs and LTNs. We compare the number of parameters for grounding a unary predicate for each model to comprehend the relative performance of RWFNs and LTNs. The input dimension in the dataset for

Task:	Phrase Det.	Phrase Det.	Relation Det.	Relation Det.	Predicate Det.	Predicate Det.
Evaluation:	R@100	R@50	R@100	R@50	R@100	R@50
$\mathcal{LTN}^{\text{expl}}$	*14.99 ± 1.2*	10.52 ± 0.59	*13.42 ± 1.1*	9.39 ± 0.39	68.59 ± 0.93	50.61 ± 1.7
$\mathcal{LTN}^{\text{prior}}$	**16.15 ± 0.75**	**11.82 ± 0.37**	**14.81 ± 0.81**	**10.69 ± 0.32**	74.66 ± 1.6	*54.9 ± 2.1*
$\mathcal{RWFN}^{\text{expl}}$	*15.48 ± 0.51*	10.7 ± 0.74	*14.13 ± 0.51*	9.61 ± 0.61	*76.97 ± 0.08*	*56.17 ± 0.61*
$\mathcal{RWFN}^{\text{prior}}$	*15.93 ± 0.39*	*10.78 ± 0.81*	*14.44 ± 0.47*	*9.89 ± 0.65*	*77.3 ± 0.6*	**57.01 ± 1.3**
$\mathcal{RWFN}_{\mathcal{WS}}{}^{\text{expl}}$	*15.19 ± 0.31*	10.45 ± 0.84	*14.03 ± 0.36*	*9.49 ± 0.99*	*77.11 ± 1.4*	*55.62 ± 0.91*
$\mathcal{RWFN}_{\mathcal{WS}}{}^{\text{prior}}$	*15.67 ± 0.75*	*11.1 ± 0.99*	*14.4 ± 0.49*	*10.08 ± 0.85*	**77.33 ± 0.32**	*56.89 ± 0.91*

Table 1: Results on Visual Relationship Dataset (R@N: Recall at N) for LTN, RWFN, and RWFN with weight sharing. 95% CIs (i.e., MEAN $\pm$ 2 $\times$ SD) shown for all models. Best-mean CIs shown in **bold**; CIs that overlap with best-mean CI in *italic*.

both RWFNs and LTNs is $n = 105$ for grounding a unary predicate. As shown in Eq. (1), the parameters to learn in LTNs are $\{u_P \in \mathbb{R}^k, W_P^{[1:k]} \in \mathbb{R}^{n\times n\times k}, V_P \in \mathbb{R}^{k\times n}, b_P \in \mathbb{R}^k\}$, where $k = 5$ in the setting of the LTNs. Therefore, the number of parameters in LTNs is $(n^2 + n + 2) \cdot k = (105^2 + 105 + 2) \cdot 5 = 55660$. On the other hand, in Eq. (4) and Eq. (5), the number of parameters in RWFNs are $\{\mathbf{W} \in \mathbb{R}^{n\times B}, \mathbf{R} \in \mathbb{R}^{n\times B}, \mathbf{b} \in \mathbb{R}^B, \boldsymbol{\beta} \in \mathbb{R}^{2B}\}$, where $B = 500$ in the setting of the RWFNs. Thus, the number of parameters in RWFNs is $(2n + 3) \cdot B = (2 \cdot 105 + 3) \cdot 500 = 106500$. Although RWFNs require more space complexity compared to LTNs, the parameters $\{\mathbf{W}, \mathbf{R}, \mathbf{b}\}$ in RWFNs are randomly drawn weights. Thus, it is necessary to compare the number of learnable parameters across the two models as well.

All of the above parameters of LTNs have to be learnable whereas the parameters to learn in RWFNs for object type classification are only $\boldsymbol{\beta} \in \mathbb{R}^{2B}$. Thus, the number of learnable parameters is 1000, which is much smaller than that of LTNs. It means that the ratio of the two numbers of parameters to learn is about $1000 : 55660 \approx 1 : 56$. Consequently, non-learnable parameters in RWFNs can have significant potential to represent the latent relationship among objects and efficiently extract relational knowledge even though using fewer adaptable parameters. Furthermore, the number of LTN parameters heavily depends on the number of features, whereas RWFNs are independent of features.

Space Complexity of RWFNs with Weight Sharing. The unique property of RWFNs, weight sharing, allows reducing space complexity greatly when multiple classifiers are used simultaneously. By referring to the depicted case of learning i classifiers in Fig. 2, we compute the space complexity for RWFNs with and without weight sharing for the detection of unary predicates in the tasks. Hong and Pavlic (2021) show that the space complexity for the original RWFNs is $(2 \cdot n + 3) \cdot B \cdot i = (2 \cdot 105 + 3) \cdot 500 \cdot 100 = 1.065 \times 10^7$ because the number of classifiers i is 100. However, with weight sharing, RWFNs can achieve much better space complexity, which is $2 \cdot n \cdot B + B + 2 \cdot B \cdot i = 2 \cdot 105 \cdot 500 + 500 + 2 \cdot 500 \cdot 100 = 205500$ and this complexity is much smaller than one of the original RWFNs. Furthermore, because the LTNs require 55660 parameters for grounding a single unary predicate, the space complexity of LTNs for grounding all unary predicates is $55660 \times 100 = 5566000$, which is much larger than the space complexity of RWFNs with weight sharing. It indicates that the ratio of the two space complexities between RWFNs with weight sharing and LTNs is $205500 : 5566000 \approx 1 : 27$ and that the weight sharing property allows RWFNs to be more cost efficient and economical than LTNs even though the performance of RWFNs with weight sharing for predicate detection is better than LTNs.

Conclusion

We showed that Randomly Weighted Feature Networks can be extended to a zero-shot approach that learns the similarity with other seen triples in the presence of logical background knowledge. The results on the Visual Relationship Dataset show that RWFNs outperform LTNs with far fewer parameters to train. The proposed method addresses not only an emerging problem in AI datasets due to the high annotation effort and their consequent incompleteness but also a critical problem in the neuro-symbolic domain – the reduction of the number of training parameters – which could allow for online training of the neuro-symbolic models with real-time performance possible in the future. In addition, the combination of bio-inspired neural models with logical prior knowledge shows how biologically inspired neural networks *plus knowledge* can learn with few parameters with respect to artificial neural networks and can be applied to even more complex computer vision tasks.

The proposed method can be improved in various ways. For one, RWFNs can be employed in tasks that should extract structural knowledge from images as well as text, such as visual question answering using Visual Genome dataset (Krishna et al. 2016). Moreover, other perspectives from neuroscience may lead to biologically plausible learning algorithms that might apply to further optimizations of RWFNs (Krotov and Hopfield 2019; Kasai et al. 2021; Kappel et al. 2018). Furthermore, RWFNs may be able to incorporate a recurrent component for representing dynamic features of time-series data, similar to reservoir computing (Ferreira and Ludermir 2009; Sun et al. 2017; Wang, Jin, and Hao 2019); this approach may allow for extracting time-varying relational knowledge necessary for developing a framework for data-driven reasoning over temporal logic.

Appendix: Details of Experiments

Hyperparameter Searching for RWFNs We used the Optuna framework (Akiba et al. 2019) with 500 iterations in the range of $[64, 1024]$ to determine the best number of hidden nodes β in Eq. (6). Because in the Optuna framework, we can formalize hyperparameter optimization as the maximization or minimization process of an objective function that takes a set of hyperparameters as input and returns a validation score, we can easily construct the parameter search space dynamically. In addition, the framework provides efficient sampling methods, such as relational sampling that exploits the correlations among the parameters.

Hardware specification of the server. The hardware specification of the server that we used to experiment is as follows:

- CPU: Intel® Core^{TM} i7-6950X CPU @ 3.00GHz (up to 3.50 GHz)
- RAM: 128 GB (DDR4 2400MHz)
- GPU: NVIDIA GeForce Titan Xp GP102 (Pascal architecture, 3840 CUDA Cores @ 1.6 GHz, 384 bit bus width, 12 GB GDDR G5X memory)

References

Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M.; et al. 2016. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; and Koyama, M. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2623–2631.

Atif, J.; Hudelot, C.; and Bloch, I. 2013. Explanatory reasoning for image understanding using formal concept analysis and description logics. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 44(5): 552–570.

Avarguès-Weber, A.; Deisig, N.; and Giurfa, M. 2011. Visual cognition in social insects. *Annual review of entomology*, 56: 423–443.

Avarguès-Weber, A.; and Giurfa, M. 2013. Conceptual learning by miniature brains. *Proceedings of the Royal Society B: Biological Sciences*, 280(1772): 20131907.

Badreddine, S.; d'Avila Garcez, A.; Serafini, L.; and Spranger, M. 2021. Logic Tensor Networks. arXiv:2012.13635.

Baker, C. F.; Fillmore, C. J.; and Lowe, J. B. 1998. The berkeley framenet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, 86–90.

Bergmann, M. 2008. *An introduction to many-valued and fuzzy logic: semantics, algebras, and derivation systems*. Cambridge University Press.

Caron, S. J.; Ruta, V.; Abbott, L.; and Axel, R. 2013. Random convergence of olfactory inputs in the Drosophila mushroom body. *Nature*, 497(7447): 113–117.

Chen, N.; Zhou, Q.-Y.; and Prasanna, V. 2012. Understanding web images by object relation network. In *Proceedings of the 21st international conference on World Wide Web*, 291–300.

Chen, X.; Mottaghi, R.; Liu, X.; Fidler, S.; Urtasun, R.; and Yuille, A. 2014. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1971–1978.

Cope, A. J.; Vasilaki, E.; Minors, D.; Sabo, C.; Marshall, J. A.; and Barron, A. B. 2018. Abstract concept learning in a simple neural network inspired by the insect brain. *PLoS computational biology*, 14(9): e1006435.

Dai, B.; Zhang, Y.; and Lin, D. 2017. Detecting visual relationships with deep relational networks. In *Proceedings of the IEEE conference on computer vision and Pattern recognition*, 3076–3086.

Donadello, I.; and Serafini, L. 2019. Compensating supervision incompleteness with prior knowledge in semantic image interpretation. In *2019 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Donadello, I.; Serafini, L.; and Garcez, A. D. 2017. Logic tensor networks for semantic image interpretation. *arXiv preprint arXiv:1705.08968*.

Ferreira, A. A.; and Ludermir, T. B. 2009. Genetic algorithm for reservoir computing optimization. In *2009 International Joint Conference on Neural Networks*, 811–815. IEEE.

Glorot, X.; Bordes, A.; and Bengio, Y. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 315–323. JMLR Workshop and Conference Proceedings.

Hájek, P. 2013. *Metamathematics of fuzzy logic*, volume 4. Springer Science & Business Media.

Hong, J.; and Pavlic, T. P. 2021. An Insect-Inspired Randomly, Weighted Neural Network with Random Fourier Features For Neuro-Symbolic Relational Learning. In *Proceedings of the 15th International Workshop on Neural-Symbolic Learning and Reasoning (Ne/Sy 2022)*.

Hudelot, C.; Atif, J.; and Bloch, I. 2008. Fuzzy spatial relation ontology for image interpretation. *Fuzzy Sets and Systems*, 159(15): 1929–1951.

Inada, K.; Tsuchimoto, Y.; and Kazama, H. 2017. Origins of cell-type-specific olfactory processing in the Drosophila mushroom body circuit. *Neuron*, 95(2): 357–367.

Kappel, D.; Legenstein, R.; Habenschuss, S.; Hsieh, M.; and Maass, W. 2018. A dynamic connectome supports the emergence of stable computational function of neural circuits through reward-based learning. *Eneuro*, 5(2).

Kasai, H.; Ziv, N. E.; Okazaki, H.; Yagishita, S.; and Toyoizumi, T. 2021. Spine dynamics in the brain, mental disorders and artificial neural networks. *Nature Reviews Neuroscience*, 22(7): 407–422.

Krishna, R.; Zhu, Y.; Groth, O.; Johnson, J.; Hata, K.; Kravitz, J.; Chen, S.; Kalantidis, Y.; Li, L.-J.; Shamma, D. A.; et al. 2016. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *arXiv preprint arXiv:1602.07332*.

Krishna, R.; Zhu, Y.; Groth, O.; Johnson, J.; Hata, K.; Kravitz, J.; Chen, S.; Kalantidis, Y.; Li, L.-J.; Shamma, D. A.; et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1): 32–73.

Krotov, D.; and Hopfield, J. J. 2019. Unsupervised learning by competing hidden units. *Proceedings of the National Academy of Sciences*, 116(16): 7723–7731.

Kulkarni, G.; Premraj, V.; Ordonez, V.; Dhar, S.; Li, S.; Choi, Y.; Berg, A. C.; and Berg, T. L. 2013. Babytalk: Understanding and generating simple image descriptions. *IEEE transactions on pattern analysis and machine intelligence*, 35(12): 2891–2903.

Lampert, C. H.; Nickisch, H.; and Harmeling, S. 2013. Attribute-based classification for zero-shot visual object categorization. *IEEE transactions on pattern analysis and machine intelligence*, 36(3): 453–465.

Li, Y.; Ouyang, W.; Wang, X.; and Tang, X. 2017. Vip-cnn: Visual phrase guided convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1347–1356.

Liang, X.; Lee, L.; and Xing, E. P. 2017. Deep variation-structured reinforcement learning for visual relationship and attribute detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 848–857.

Lu, C.; Krishna, R.; Bernstein, M.; and Fei-Fei, L. 2016. Visual relationship detection with language priors. In *European conference on computer vision*, 852–869. Springer.

McMahan, H. B.; Holt, G.; Sculley, D.; Young, M.; Ebner, D.; Grady, J.; Nie, L.; Phillips, T.; Davydov, E.; Golovin, D.; et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1222–1230.

Menzel, R. 2001. Searching for the memory trace in a mini-brain, the honeybee. *Learning & memory*, 8(2): 53–62.

Mobbs, P. 1982. The brain of the honeybee *Apis mellifera*. I. The connections and spatial organization of the mushroom bodies. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, 298(1091): 309–354.

Neumann, B.; and Möller, R. 2008. On scene interpretation with description logics. *Image and Vision Computing*, 26(1): 82–101.

Peng, F.; and Chittka, L. 2017. A simple computational model of the bee mushroom body can explain seemingly complex forms of olfactory learning and memory. *Current Biology*, 27(2): 224–230.

Rahimi, A.; and Recht, B. 2007. Random Features for Large-Scale Kernel Machines. In *NIPS*, volume 3, 5.

Ramanathan, V.; Li, C.; Deng, J.; Han, W.; Li, Z.; Gu, K.; Song, Y.; Bengio, S.; Rosenberg, C.; and Fei-Fei, L. 2015. Learning semantic relationships for better action retrieval in images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1100–1109.

Schuler, K. K. 2005. *VerbNet: A broad-coverage, comprehensive verb lexicon*. University of Pennsylvania.

Serafini, L.; and Garcez, A. S. d. 2016. Learning and reasoning with logic tensor networks. In *Conference of the Italian Association for Artificial Intelligence*, 334–348. Springer.

Smola, A. J.; and Schölkopf, B. 1998. *Learning with kernels*, volume 4. Citeseer.

Socher, R.; Chen, D.; Manning, C. D.; and Ng, A. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, 926–934.

Sun, X.; Li, T.; Li, Q.; Huang, Y.; and Li, Y. 2017. Deep belief echo-state network and its application to time series prediction. *Knowledge-Based Systems*, 130: 17–29.

Sutherland, D. J.; and Schneider, J. 2015. On the error of random Fourier features. *arXiv preprint arXiv:1506.02785*.

Tieleman, T.; and Hinton, G. 2012. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

Wang, X.; Jin, Y.; and Hao, K. 2019. Echo state networks regulated by local intrinsic plasticity rules for regression. *Neurocomputing*, 351: 111–122.

Xu, D.; Zhu, Y.; Choy, C. B.; and Fei-Fei, L. 2017. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5410–5419.

Yin, G.; Sheng, L.; Liu, B.; Yu, N.; Wang, X.; Shao, J.; and Loy, C. C. 2018. Zoom-net: Mining deep feature interactions for visual relationship recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 322–338.

Yu, R.; Li, A.; Morariu, V. I.; and Davis, L. S. 2017. Visual relationship detection with internal and external linguistic knowledge distillation. In *Proceedings of the IEEE international conference on computer vision*, 1974–1982.

Zellers, R.; Yatskar, M.; Thomson, S.; and Choi, Y. 2018. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5831–5840.