Large-Scale Visual Relationship Understanding

Ji Zhang*[1,2], Yannis Kalantidis[1], Marcus Rohrbach[1], Manohar Paluri[1], Ahmed Elgammal[2], Mohamed Elhoseiny[1]

[1]Facebook Research
[2]Department of Computer Science, Rutgers University

Abstract

Large scale visual understanding is challenging, as it requires a model to handle the widely-spread and imbalanced distribution of ⟨subject, relation, object⟩ triples. In real-world scenarios with large numbers of objects and relations, some are seen very commonly while others are barely seen. We develop a new relationship detection model that embeds objects and relations into two vector spaces where both discriminative capability and semantic affinity are preserved. We learn both a visual and a semantic module that map features from the two modalities into a shared space, where matched pairs of features have to discriminate against those unmatched, but also maintain close distances to semantically similar ones. Benefiting from that, our model can achieve superior performance even when the visual entity categories scale up to more than $80,000$*, with extremely skewed class distribution. We demonstrate the efficacy of our model on a large and imbalanced benchmark based of Visual Genome that comprises* $53,000+$ *objects and* $29,000+$ *relations, a scale at which no previous work has ever been evaluated at. We show superiority of our model over carefully designed baselines on the original Visual Genome dataset with* $80,000+$ *categories. We also show state-of-the-art performance on the VRD dataset and the scene graph dataset which is a subset of Visual Genome with* 200 *categories.*

1. Introduction

Scale matters. In the real world, people tend to describe visual entities with open vocabulary, e.g., the raw ImageNet[5] dataset has 21,841 synsets that covers even more object classes. The vocabulary size becomes significantly larger for relationships since the combinations of ⟨subject, relation, object⟩ are orders of magnitude more than objects[21, 27, 39]. Moreover, the long-tailed distribution of objects could be an obstacle for a model to learn all classes sufficiently, and such challenge is exacerbated in relationship detection because either the subject, the object, or the relation could be infrequent, or their triple might be jointly infrequent. Figure 1 shows an example from the Visual Genome dataset, which contains commonly seen relationship (e.g., "man, wearing, glasses") along with uncommon ones (e.g., "dog, next to, woman").

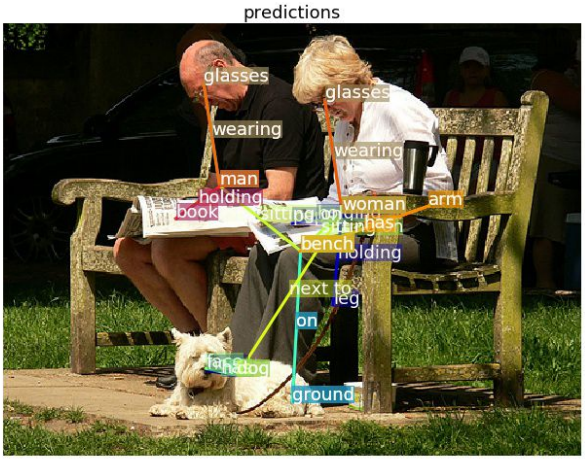


Figure 1: Relationships predicted by our approach on an image. Different relationships are colored differently with a relation line connecting each subject and object. Our model is able to recognize relationships composed of over $53,000$ object categories and over $29,000$ relation categories.

The second challenge lies in the fact that object categories are often semantically associated[5, 16, 4], and such connections could be more subtle for relationships. For example, an image of "person ride horse" could look like one of "person ride elephant" since they both belong to the kind of relationships where a person is riding an animal, but "person carry bike" would look very different from "person ride bike" even though they have the same subject and object. It

is critical for a model to be able to leverage such semantic connections.

In this work, we aim to study relationship detection at an unprecedented scale where the total number of visual entities is more than 80,000. We carefully develop a scalable approach that is semantically-guided and a loss that enables discriminative learning in such a large-scale setting. We use a continuous output space for objects and relations instead of discrete labels, which enables knowledge to be easily transferred from frequent classes to infrequent ones, and such ability stimulates accurate recognition of large amount of rarely seen relationships.

Meanwhile, we also show that in order to enable knowledge transfer in the continuous space and preserve discriminative power at the same time, the well-known multi-class logistic loss (i.e., the loss used in softmax models) fails in the former and triplet loss[15] (that is widely used when learning in continuous space) fails in the latter, while our *triplet-softmax* loss is competent to the task. On the Visual Genome (VG) dataset, we design an intuitively straightforward baseline and show our superior performance over it. We demonstrate our model's competence over triplet loss on the whole dataset and over multi-class logistic loss on the long tail data. Our model also achieves state-of-the-art performance on the Visual Relationship Detection (VRD) dataset, and the scene graph dataset[33] which is a subset of Visual Genome with 200 categories.

2. Related Work

Our work is at the intersection between semantic-embedding models and visual relationship detection and hence we discuss both angles. **Visual Relationship Detection** A large number of visual relationship detection approaches have emerged during the last couple of years. Almost all of them are based on a small vocabulary, e.g., 100 object and 70 relation categories from the VRD dataset[21], or a subset of VG with the most frequent object and relation categories (e.g., 200 object and 100 relation categories from [37]).

In one of the earliest works, Lu *et al.* [21] utilize the object detection output of an an R-CNN detector and leverage language priors from semantic word embeddings to fine-tune the likelihood of a predicted relationship. Very recently, Zhuang *et al.* [41] use language representations of the subject and object as "context" to derive a better classification result for the relation. However, similar to Lu *et al.* [21] their language representations are pre-trained. Unlike these approach, we fine-tune subject and object representations *jointly* and employ the interaction between branches also at an earlier stage before classification.

In [35], the authors employ knowledge distillation from a large Wikipedia-based corpus and get state-of-the-art results for the VRD [21] dataset. In ViP-CNN [18], the authors pose the problem as a classification task on limited classes and therefore cannot scale to the scenarios that we can cope with. In our model we exploit the training set concept co-occurrences at the relationship level to model such knowledge. Our approach directly targets the large category scale and is able to utilize semantic associations to compensate for infrequent classes, while at the same time achieves competitive performance in the smaller and constrained VRD [21] dataset.

Very recent approaches like [40, 27] target open-vocabulary for scene parsing and visual relationship detection, respectively. In [27], the related work closest to ours, the authors learn a CCA model on top of different combinations of the subject, object and union regions and train a Rank SVM. They however consider each relationship triplet as a class and learn it as a whole entity, thus cannot scale to our setting. Our approach embeds the three components of a relationship separately to the independent semantic spaces for object and relation, but implicitly learns connections between them via visual feature fusion and semantic meaning preservation in the embedding space.

Semantically Guided Visual Recognition. Another parallel category of vision and language tasks is known as zero-shot learning. In [10], [24] and [30], word embedding language models (e.g., [22]) were adopted to represent class names as vectors and hence allow zero-shot recognition. For fine-grained objects like birds and flowers, several works adopted Wikipedia Articles to guide zero-shot recognition since they are easy-collect language and provides more signal for recognition (e.g., [8, 7, 1, 6]. However, for relations and actions, these methods are not designed with the capability of locating the objects or interacting objects for visual relations. Several approaches have been proposed to model the visual-semantic embedding in the context of the image-sentence similarity task (e.g., [15, 31, 9, 32, 11]). Most of them focused on leaning semantic connections between the two modalities, which we not only aim to achieve, but with a manner that does not sacrifice discriminative capability since our task is detection instead of similarity-based retrieval. In contrast, visual relations also has a structure ⟨subject, relation, object⟩ and we show in our results that proper design of a visual-semantic embedding architecture and loss is critical for good performance. We compare our approach against most of the aforementioned related works on common benchmarks in Section 4.

Note: in this paper we use "relation" to refer to what is also known as 'predicate" in previous works, and "relationship" or "relationship triplet" to refer to a ⟨subject, relation, object⟩ tuple.

3. Method

The task of relationship detection naturally requires a model to have discriminative power among a set of cate-

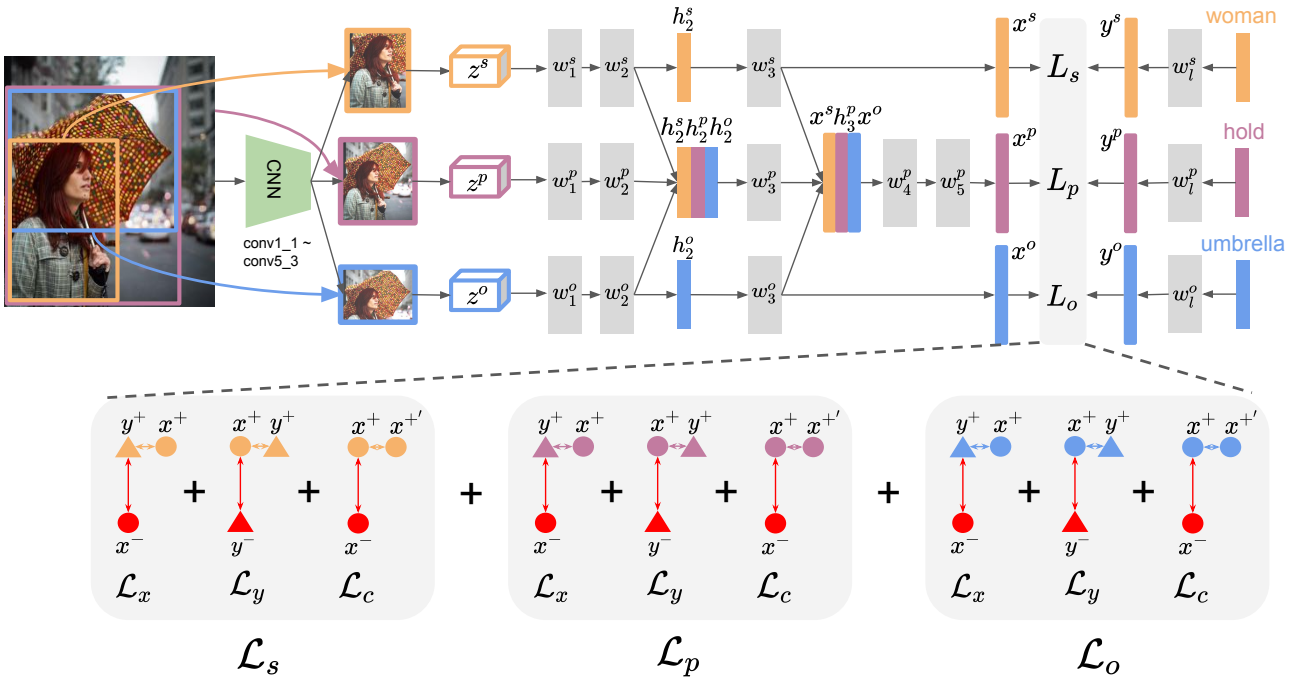


Figure 2: (a) Overview of the proposed approach. L_s, L_p, L_o are the losses of subject, relation and object. Orange, purple and blue colors represent subject, relation, object, respectively. Grey rectangles are fully connected layers, which are followed by ReLU activations except the last ones, i.e. w_3^s, w_5^p, w_3^o. We share layer weights of the subject and object branches, i.e. w_i^s and w_i^o, $i = 1, 2...5$.

gories. It is well known that softmax + multi-class logistic loss is the best choice when the categories are not too many and have little association among each other [17, 29, 13]. However, in the real-world setting these two assumptions are often not true, since people might describe things in unlimited ways with potentially similar meanings. In such scenarios, it is critical for a model to be able to preserve semantic similarities, especially for those rarely seen categories since they can "borrow" the knowledge learned from other frequently seen categories, while not sacrifice discriminative power for that. In order to achieve that, our model adopts a two-way pipeline that maps images and labels to a shared embedding space. We use a new loss that is better than both softmax-based and triplet loss in the large-vocabulary setting. During testing, recognition is done by nearest neighbor search in the embedding space and is linear with the size of the vocabulary.

3.1. Visual Module

The design logic of our visual module is that a relation exists when its subject and object exist, but not vice versa. Namely, relation recognition is conditioned on subject and object, but object recognition is independent from relations. The main reason is that we want to learn embeddings for subject and object in a separate semantic space from the relation space. That is, we want to learn a mapping from visual feature space (which is shared among subject/object and relation) to the two separate semantic embedding spaces (for objects and relations). Therefore, involving relation features for subject/object embeddings would have the risk of entangling the two spaces. As shown in Figure 2 an image is fed into a CNN ($conv1\_1$ to $conv5\_3$ of VGG16) to get a global feature map of the image, then the subject, relation and object features z^s, z^p, z^o are ROI-pooled with the corresponding regions $\mathcal{R}_\mathcal{S}$, $\mathcal{R}_\mathcal{P}$, $\mathcal{R}_\mathcal{O}$, each branch followed by two fully connected layers which output three intermediate hidden features h_2^s, h_2^p, h_2^o. For the subject/object branch, we add another fully connected layer w_3^s to get the visual embedding x^s, and similarly for the object branch to get x^o. For the relation branch, we we first concatenate the three hidden features h_2^s, h_2^p, h_2^o and feed it to a fully connected layer w_3^p to get a higher-level hidden feature h_3^p, then we concatenate the subject and object embeddings x^s and x^o with h_3^p and feed it to two fully connected layers w_4^p w_5^p to get the relation embedding x^p.

There are several things worth highlighting. First, w_1^l and w_2^l are the counterparts of the $fc6$ and $fc7$ layers in VGG16, thus both of them are necessary in order to preserve the capacity of VGG16. Second, both concatenations in the relation branch are necessary and sufficient, as shown

in Section 4.4. The first concatenation integrates relatively low level visual features intrinsically captured by the feature extractor, while the second fuses embeddings of subject and object with h_3^p. It can be interpreted as integration of three higher level features. Third, unlike most previous works that concatenate spatial and visual features, we found that such concatenation barely helps in our setting, because spatial layouts of relationships are much more diverse due to the large scale of both instances and classes, which is also observed in Zhang *et al.* [39].

3.2. Semantic Module

On the semantic side, we feed word vectors of subject, relation and object labels into a small MLP of one or two fc layers which outputs the embeddings. As in the visual module, the subject and object branches share weights while the relation branch is independent. The purpose of this module is to map word vectors into an embedding space that are more discriminative than the raw word vector space while preserving semantic similarity. During training, we feed the ground-truth labels of each relationship triplet as well as labels of negative classes into the semantic module, as the following subsection describes; during testing, we feed the whole sets of object and relation labels into it for nearest neighbors searching among all the labels to get the top k as our prediction.

A good word vector representation for object/relation labels is critical as it provides proper initialization that is easy to fine-tune on. We consider the following word vectors:

Pre-trained word2vec embeddings (wiki) We rely on the pre-trained word embeddings provided by [22] which are widely used in prior work. We use this embedding as a baseline, and show later that by combining with other embeddings we achieve better discriminative ability.

Relationship-level co-occurrence embeddings (relco). This embedding encodes co-occurrence in the relationship level. By considering each relationship triplet seen in the training set as the "sentence" or "document" in a word2vec setting, each relation will have as context the subjects and object it usually appears with. Similarly, relations that tend to appear with a subject or object are used as context for learning their embeddings. We train a skip-gram word2vec model that tries to maximize classification of a word based on another word in the same context. As is in our case we define context via our training set's relationships, we effectively learn to maximize the likelihoods of $P(P|S,O)$ as well as $P(S|P,O)$ and $P(O|S,P)$. Although maximizing $P(P|S,O)$ is directly optimized in [35], we achieve similar results by reducing it to a skip-gram model and enjoy the scalability of a word2vec approach.

Node2vec embeddings (node2vec). As the Visual Genome dataset further provides image-level relation graphs, we also experimented with training *node2vec* embeddings as in [12]. These are effectively also word2vec embeddings, but the context is determined by random walks on a graph. In this setting, nodes correspond to subjects, objects and relations from the training set and edges are directed from $S \rightarrow P$ and from $P \rightarrow O$ for every image-level graph. This embedding can be seen as an intermediate between image-level and relationship level co-occurrences, with proximity to the one or the other controlled via the length of the random walks.

3.3. Training Loss

To learn the joint visual and semantic embedding we employ a modified triplet loss. Traditional triplet loss [15] encourages matched embeddings from the two modalities to be closer than the mismatched ones by a fixed margin, while our version tries to maximize this margin in a softmax form. In this subsection we review the traditional triplet loss and then introduce our triplet-softmax loss in a comparable fashion. To this end, we denote the two sets of triplets for each positive visual-semantic pair by $(\mathbf{x}^l, \mathbf{y}^l)$:

$$tri_{\mathbf{x}}^l = \{\mathbf{x}^l, \mathbf{y}^l, \mathbf{x}^{l-}\} \tag{1}$$

$$tri_{\mathbf{y}}^l = \{\mathbf{x}^l, \mathbf{y}^l, \mathbf{y}^{l-}\} \tag{2}$$

where $l \in \{s, p, o\}$, and the two sets $tri_{\mathbf{x}}, tri_{\mathbf{y}}$ correspond to triplets with negatives from the visual and semantic space, respectively.

Triplet loss. If we omit the superscripts $\{s, p, o\}$ for clarity, the triplet loss $\mathcal{L}^{Tr}$ for each branch is summation of two losses $\mathcal{L}_{\mathbf{x}}^{Tr}$ and $\mathcal{L}_{\mathbf{y}}^{Tr}$:

$$\mathcal{L}_{\mathbf{x}}^{Tr} = \frac{1}{NK} \sum_{i=1}^{N} \sum_{j=1}^{K} \max[0, m + s(\mathbf{y}_i, \mathbf{x}_{ij}^-) - s(\mathbf{x}_i, \mathbf{y}_i)] \tag{3}$$

$$\mathcal{L}_{\mathbf{y}}^{Tr} = \frac{1}{NK} \sum_{i=1}^{N} \sum_{j=1}^{K} \max[0, m + s(\mathbf{x}_i, \mathbf{y}_{ij}^-) - s(\mathbf{x}_i, \mathbf{y}_i)] \tag{4}$$

$$\mathcal{L}^{Tr} = \mathcal{L}_{\mathbf{x}}^{Tr} + \mathcal{L}_{\mathbf{y}}^{Tr} \tag{5}$$

where N is the number of positive ROIs, K is the number of negative samples *per positive* ROI, m is the margin between the distances of positive and negative pairs, and $s(\cdot, \cdot)$ is a similarity function.

We can observe from Equation (3) that as long as the similarity between positive pairs is larger than that between negative ones by margin m, $[m + s(\mathbf{x}_i, \mathbf{x}_{ij}^-) - s(\mathbf{x}_i, \mathbf{y}_i)] \leq 0$, and thus $max(0, \cdot)$ will return zero for that part. That means, during training once the margin is pushed to be larger than m, the model will stop learning anything from that triplet. Therefore, it is highly likely to end up with an embedding space where points are not discriminative enough for a classification-oriented task.

Triplet-Softmax loss. The issue of triplet loss mentioned above can be alleviated by applying softmax top of each triplet, i.e.:

$$\mathcal{L}_{\mathbf{x}}^{TrSm} = \frac{1}{N}\sum_{i=1}^{N} -\log \frac{e^{s(\mathbf{x}_i,\mathbf{y}_i)}}{e^{s(\mathbf{x}_i,\mathbf{y}_i)} + \sum_{j=1}^{K} e^{s(\mathbf{y}_i,\mathbf{x}_{ij}^-)}} \quad (6)$$

$$\mathcal{L}_{\mathbf{y}}^{TrSm} = \frac{1}{N}\sum_{i=1}^{N} -\log \frac{e^{s(\mathbf{x}_i,\mathbf{y}_i)}}{e^{s(\mathbf{x}_i,\mathbf{y}_i)} + \sum_{j=1}^{K} e^{s(\mathbf{x}_i,\mathbf{y}_{ij}^-)}} \quad (7)$$

$$\mathcal{L}^{TrSm} = \mathcal{L}_{\mathbf{x}}^{TrSm} + \mathcal{L}_{\mathbf{y}}^{TrSm} \quad (8)$$

where $s(\cdot,\cdot)$ is the same similarity function (we use cosine similarity in this paper). All the other notations are the same as above. For each positive pair $(\mathbf{x}_i, \mathbf{y}_i)$ and its corresponding set of negative pairs $(\mathbf{x}_i, \mathbf{y}_{ij}^-)$, we calculate similarities between each of them and put them into a softmax layer followed by multi-class logistic loss so that the similarity of positive pairs would be pushed to be 1, and 0 otherwise. Compared to triplet loss, this loss always tries to enlarge the margin to its largest possible value (i.e., 1), thus has more discriminative power than the traditional triplet loss.

It is worth noting that although theoretically traditional triplet loss can pushes the margin as much as possible when $m = 1$, most previous works (e.g., [15, 31, 9]) adopted a small m to allow slackness during training. It is also unclear how to determine the exact value of m given a specific task. We follow previous works and set $m = 0.2$ in all of our experiments.

Visual Consistency loss. To further force the embeddings to be more discriminative, we add a loss that pulls closer the samples from the same category while pushes away those from different categories, i.e.:

$$\mathcal{L}_c = \frac{1}{NK}\sum_{i=1}^{N}\sum_{j=1}^{K} \max[0, m + s(\mathbf{x}_i, \mathbf{x}_{ik}^-) - \min_{l\in\mathcal{C}(i)} s(\mathbf{x}_i, \mathbf{x}_l)] \quad (9)$$

where N is the number of positive ROIs, $\mathcal{C}(l)$ is the set of positive ROIs in the same class of $\mathbf{x}_i$, K is the number of negative samples *per positive* ROI and m is the margin between the distances of positive and negative pairs. The interpretation of this loss is: the minimum similarity between samples from the same class should be larger than any similarity between samples from different classes by a margin. Here we utilize the traditional triplet loss format since we want to introduce slackness between visual embeddings to prevent embeddings from collapsing to the class centers.

Empirically we found it the best to use triplet-softmax loss for $\mathcal{L}_{\mathbf{y}}$ while using triplet loss for $\mathcal{L}_{\mathbf{x}}$. The reason is similar with that of the visual consistency loss: mode collapse should be prevented by introducing slackness. On the other hand, there is no such issue for y since each label y is a mode by itself, and we encourage all modes of y to be separated from each other. In conclusion, our final loss is:

$$\mathcal{L} = \mathcal{L}_{\mathbf{y}}^{TrSm} + \alpha\mathcal{L}_{\mathbf{x}}^{Tr} + +\beta\mathcal{L}_c \quad (10)$$

where we found that $\alpha = \beta = 1$ works reasonably well for all scenarios.

Connection between Softmax and Triplet-Softmax. Our triplet-softmax module can be interpreted as a variant of a regular softmax. Specifically, for softmax we have:

$$l = softmax(fc(\mathbf{x})) = softmax(\mathbf{Y}^T\mathbf{x}) = \underset{i}{\text{argmax}}(\mathbf{y}_i^T\mathbf{x}) \quad (11)$$

where $\mathbf{x}$ is the input feature to the last fc layer, $\mathbf{Y}$ is the learned weights of that fc layer, and $\mathbf{y}_i$ is the i-th column of $\mathbf{Y}$. For triplet-softmax we have:

$$l = \underset{i}{\text{argmax}}\, s(\hat{\mathbf{x}}, \hat{\mathbf{y}}_i) = \underset{i}{\text{argmax}}(\hat{\mathbf{y}}_i^T\hat{\mathbf{x}}) = \underset{i}{\text{argmax}}(\mathrm{S}(\mathbf{v}_i)^T\hat{\mathbf{x}}) \quad (12)$$

where $\hat{\mathbf{x}}$ is normalized vector of $\mathbf{x}$, $\mathbf{v}_i$ is the input word vector to the semantic module $\mathrm{S}()$, and its output is $\hat{\mathbf{y}}_i$ which is also normalized by default. It is clear that triplet-softmax can be seen as a special softmax whose last fc layer weights is provided by the semantic module's output instead of learned independently within the visual module. Such structure with semantic guidance before softmax is the very reason that our model is both discriminative and similarity-preserving.

3.4. ROI Sampling

One of the critical things that powers Fast-RCNN is the well-designed ROI sampling during training. It ensures that for most ground-truth boxes, each of them has 32 positive ROIs and $128 - 32 = 96$ negative ROIs, where positivity is defined as overlap IoU $>= 0.5$. In our setting, ROI sampling is similar for the subject/object branch, while for the relation branch, positivity is defined as both subject and object IoUs $>= 0.5$. Accordingly, we sample 64 subject ROIs with 32 unique positives and 32 unique negatives, and do the same thing for object ROIs. Then we pair all the 64 subject ROIs with 64 object ROIs to get 4096 ROI pairs as relationship candidates. For each candidate, if both ROIs' IoU $>= 0.5$ we mark it as positive, otherwise negative. We finally sample 32 positive and 96 negative relation candidates and use the union of each ROI pair as a relation ROI. In this way we end up with a consistent number of positive and negative ROIs for relation branch as subject and object branches.

4. Experiments

Datasets We present experiments on two datasets, the *Visual Genome* (VG) [16] and *Visual Relationship Detection* (VRD) dataset [21].

- **VRD** The VRD dataset [21] contains 5,000 images with 100 object categories and 70 relations. In total, VRD contains 37,993 relation annotations with 6,672 unique relations and 24.25 relationships per object category. We follow the same train/test split as in [21] to get 4,000 training images and 1,000 test images. We use this dataset to demonstrate that our model can work reasonably well on small dataset with small category space, even though it is designed for large-scale settings.
- **VG200** We also train and evaluate our model on a subset of VG80k which is widely used in previous methods[33, 23, 36, 34]. There are totally 150 object categories and 50 predicate categories in this dataset. We use the same train/test splits as in [33]. Similarly with VRD, the purpose here is to show our model is also state-of-the-art in large-scale sample but small-scale category settings.
- **VG80k** We use the latest release Visual Genome dataset (VG v1.4) [16] that contains $108,077$ images with 21 relationships on average per image. Each relationship is of the form $\langle$subject, relation, object$\rangle$ with annotated subject and object bounding boxes. We follow [14] and split the data into $103,077$ training images and $5,000$ testing images. Since text annotations of VG are noisy, we first clean it by removing non-alphabet characters and stop words, and use the `autocorrect` library to correct spelling. Following that, we check if all words in an annotation exist in the word2vec dictionary [22] and remove those that don't. If an image ends up with all annotations being removed, we delete it from the corresponding image set. Furthermore, we apply two rules to remove obvious annotation noise: 1) **merge**: we remove duplicate consecutive words since they are clearly noises, such as "horse horse" and "banana banana". We also merge phrases with the same set of words and the same meaning, such as "man smiling" and "smiling man". Specifically, we select the one with more occurrence in the training set and replace the other with it. 2) **join**: we consider phrases with the same letters in the same order as one phrase, such as "surfboard" and "surf board". Similarly, we select the one that occurs more and replace the other with it. We run this cleaning process on both training and testing set and end up with $99,961$ training images and $4,871$ testing images, with $53,304$ object categories and $29,086$ relation categories. We further split the training set into $97,961$ training and $2,000$ validation images.*

Evaluation protocol. For VG80k, we evaluate all methods on the whole $53,304$ object and $29,086$ relation categories. We use ground-truth boxes as relationship proposals, meaning there is no localization errors and the results directly reflect recognition ability of a model. We use the following metrics to measure performance: (1) top1, top5, and top10 accuracy, (2) mean reciprocal ranking (rr), defined as $\frac{1}{M}\sum_{i=1}^{M}\frac{1}{rank_i}$, (3) mean ranking (mr), defined as $\frac{1}{M}\sum_{i=1}^{M} rank_i$, smaller is better.

For VRD, we use the same evaluation metrics used in [35], which reports recall rates using the top 50 and 100 relationship predications, with $k = 1, 10, 70$ relations per relationship proposal before taking the top 50 and 100 predictions.

For VG200, we use the same evaluation metrics used in [36], which uses three modes: 1) **predicate classification**: predict predicate labels given ground truth subject and object boxes and labels; 2) **scene graph classification**: predict subject, object and predicate labels given ground truth subject and object boxes; 3) **scene graph detection**: predict all the three labels and two boxes. Recalls under the top 20, 50, 100 predications are used as the measurements.

Implementation details. For all the three datasets, we train our model for 7 epochs using 8 GPUs. We set learning rate as 0.001 for the first 5 epochs and 0.0001 for the rest 2 epochs. We initialize each branch with weights pre-trained on COCO [20]. For the word vectors, we used the `gensim` library [28] for both word2vec and node2vec† [12]. For the triplet loss, we set $m = 0.2$ for all experiments.

There is a critical factor that significantly affects our triplet-softmax loss. Since we use cosine similarity, $s(\cdot, \cdot)$ is equivalent to dot product of two normalized vectors as shown in Section 3.3. We empirically found that simply feeding normalized vector could cause gradient vanishing problem, since gradients are divided by the norm of input vector when back-propagated. This is also observed in Bell *et al.* [2] where it is necessary to scale up normalized vectors for successful learning. Similar with [2], we set the scalar to a value that is close to the mean norm of the input vectors and multiply $s(\cdot, \cdot)$ before feeding to the softmax layer. We set the scalar to 3.2 for VG80k and 3.0 for VRD in all experiments.

4.1. Comparison on VRD

We first validate our model on the small VRD dataset with comparison to state-of-the-art methods using the metrics presented in [35] in Table 1. Note that there is a variable k in this metric which is the number of relation candidates when selecting top50/100. Since not all previous methods specified k in their evaluation, we first report performance in the "free k" column when considering k as a hyper-parameter that can be cross-validated. For methods where the k is reported for 1 or more values, the column reports the performance using the best k. We then list all available results with specific k in the right two columns.

Another inconsistent factor among related works is the

*We will release the cleaned annotations along with our code.

†`https://github.com/aditya-grover/node2vec`

	Relationship		Phrase		Relationship Detection						Phrase Detection					
	free k				k = 1		k = 10		k = 70		k = 1		k = 10		k = 70	
Recall at	50	100	50	100	50	100	50	100	50	100	50	100	50	100	50	100
w/ proposals from [21]																
CAI*[41]	15.63	17.39	17.60	19.24	-	-	-	-	-	-	-	-	-	-	-	-
Language cues[27]	16.89	20.70	15.08	18.37	-	-	16.89	20.70	-	-	-	-	15.08	18.37	-	-
VRD[21]	17.43	22.03	20.42	25.52	13.80	14.70	17.43	22.03	17.35	21.51	16.17	17.03	20.42	25.52	20.04	24.90
Ours	**19.18**	**22.64**	**21.69**	**25.92**	**16.08**	**17.07**	**19.18**	**22.64**	**18.89**	**22.35**	**18.32**	**19.78**	**21.69**	**25.92**	**21.39**	**25.65**
w/ better proposals																
DR-Net*[3]	17.73	20.88	19.93	23.45	-	-	-	-	-	-	-	-	-	-	-	-
ViP-CNN[18]	17.32	20.01	22.78	27.91	17.32	20.01	-	-	-	-	22.78	27.91	-	-	-	-
VRL[19]	18.19	20.79	21.37	22.60	18.19	20.79	-	-	-	-	21.37	22.60	-	-	-	-
PPRFCN*[38]	14.41	15.72	19.62	23.75	-	-	-	-	-	-	-	-	-	-	-	-
VTransE*	14.07	15.20	19.42	22.42	-	-	-	-	-	-	-	-	-	-	-	-
SA-Full*[25]	15.80	17.10	17.90	19.50	-	-	-	-	-	-	-	-	-	-	-	-
CAI*[41]	20.14	23.39	23.88	25.26	-	-	-	-	-	-	-	-	-	-	-	-
KL distilation[35]	22.68	31.89	26.47	29.76	19.17	21.34	22.56	29.89	22.68	31.89	23.14	24.03	26.47	29.76	26.32	29.43
Ours	**26.98**	**32.63**	**32.90**	**39.66**	**23.68**	**26.67**	**26.98**	**32.63**	**26.98**	**32.59**	**28.93**	**32.85**	**32.90**	**39.66**	**32.90**	**39.64**

Table 1: Results on the VRD[21] dataset (− means unavailable / unknown)

	Scene Graph Detection			Scene Graph Classification			Predicate Classification		
Recall at	20	50	100	20	50	100	20	50	100
VRD[21]	-	0.3	0.5	-	11.8	14.1	-	27.9	35.0
Message Passing[33]	-	3.4	4.2	-	21.7	24.4	-	44.8	53.0
Message Passing+	14.6	20.7	24.5	31.7	34.6	35.4	52.7	59.3	61.3
Associative Embedding[23]	6.5	8.1	8.2	18.2	21.8	22.6	47.9	54.1	55.4
Frequency	17.7	23.5	27.6	27.7	32.4	34.0	49.4	59.9	64.1
Frequency+Overlap	20.1	26.2	30.1	29.3	32.3	32.9	53.6	60.6	62.2
MotifNet-LeftRight	**21.4**	27.2	30.3	32.9	35.8	36.5	58.5	65.2	67.1
Ours	20.7	**27.9**	**32.5**	**36.0**	**36.7**	**36.7**	**66.8**	**68.4**	**68.4**

Table 2: Results on the VG200 dataset (− means unavailable / unknown)

object proposals used. The only consistent known setting is when approaches use the publicly available proposals provided by [21], like [21, 27, 41]. Other methods either train an offline Fast/Faster-RCNN detector on VRD to extract boxes [3, 19, 38, 37, 26, 41, 35], or use the built-in RPN modules to generates proposals on-the-fly [18].

For fairness, we split the table in two parts. The top part lists methods that use the same proposals from [21], while the bottom part lists methods that are based on a different set of proposals, and ours uses better proposals obtained from Faster-RCNN as previous works. We can see that we outperform all other methods with proposals from [21] even without using message-passing-like post processing as in [18, 3], and also very competitive to the overall best performing method from [35]. Note that although spatial features could be advantageous for VRD according to previous methods, we do not use them in our model in concern of large-scale settings as explained in Section 3.1. We expect better performance if integrating spatial features for VRD, but for model consistency we do experiments without it everywhere.

4.2. Comparison on VG200

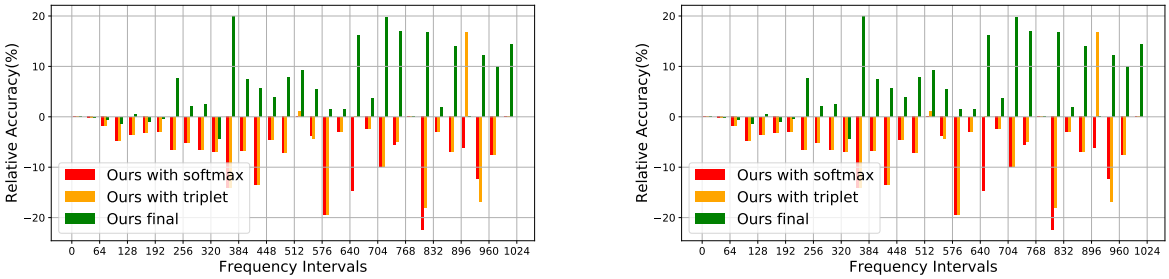
We present our results in Table 2. The results of method other than ours are borrowed from [36]. Note that scene graph classification isolates the factor of subject/object localization accuracy by using ground truth subject/object boxes, meaning that it focuses more on the relationship recognition ability of a model, and predicate classification focuses even more on it by using ground truth subject/object boxes and labels. It is clear that the gaps between our model and others are higher on scene graph/predicate classification, meaning our model has larger superiority on relation recognition ability.

4.3. Relationship Recognition on VG80k

Baselines. Since there is no previous method that works in our large-scale setting, we carefully design 3 baselines to compare with. 1) 3-branch Fast-RCNN: an intuitively straightforward model is a Fast-RCNN with a shared $conv1$ to $conv5$ backbone and 3 fc branches for subject, relation and object respectively, where the subject and object branches share weights since they are essentially an object

	Relationship Triplet					Relation				
	top1	top5	top10	rr	mr	top1	top5	top10	rr	mr
All classes										
3-branch Fast-RCNN	9.73	41.95	55.19	52.10	16.36	36.00	69.59	79.83	50.77	7.81
ours w/ triplet	8.01	27.06	35.27	40.33	32.10	37.98	61.34	69.60	48.28	14.12
ours w/ softmax	14.53	46.33	57.30	55.61	16.94	49.83	76.06	82.20	61.60	8.21
ours final	**15.72**	**48.83**	**59.87**	**57.53**	**15.08**	**52.00**	**79.37**	**85.60**	**64.12**	**6.21**
Tail classes										
3-branch Fast-RCNN	0.32	3.24	7.69	24.56	49.12	0.91	4.36	9.77	4.09	52.19
ours w/ triplet	0.02	0.29	0.58	7.73	83.75	0.12	0.61	1.10	0.68	86.60
ours w/ softmax	0.00	0.07	0.47	20.36	58.50	0.00	0.08	0.55	1.11	65.02
ours final	**0.48**	**13.33**	**28.12**	**43.26**	**45.48**	**0.96**	**7.61**	**16.36**	**5.56**	**45.70**

Table 3: Results on all relation classes and tail classes ($\#occurrence \leq 1024$) in VG80k. Note that since VG80k is extremely imbalanced, classes with no greater than 1024 occurrences are still in the tail. In fact, there are more than 99% of relation classes but only 10.04% instances of these classes that occur for no more than 1024 times.



(a) Top 5 rel triplet

(b) Top 5 relation

Figure 3: Top-5 relative accuracies against the 3-branch Fast-RCNN baseline in the tail intervals. The intervals are defined as bins of 32 from 1 to 1024 occurrences of the relation classes.

detector; 2) our model with softmax loss: we replace our loss with softmax loss; 3) our model with triplet loss: we replace our loss with triplet loss.

Results. As shown in Table 3, we can see that our loss is the best for the general case where all instances from all classes are considered. The baseline has reasonable performance but is clearly worse than ours with softmax, demonstrating that our visual module is critical for efficient learning. Ours with triplet is worse than ours with softmax in the general case since triplet loss is not discriminative enough among the massive data. However it is the opposite for tail classes (i.e., $\#occurrence \leq 1024$), since recognition of infrequent classes can benefit from the transferred knowledge learn for frequent classes, which the softmax-based model is not capable of. Another observation is that although the 3-branch Fast-RCNN baseline works poorly in the general case, it is better than our model with softmax. Since the main difference of them is with and without visual feature concatenation, it means that integrating subject and object features does not necessarily helps infrequent relation classes. This is because subject and object features could lead to strong constraint on the relation, resulting in lower chance of predicting infrequent relation when using softmax. For example, when seeing a rare image where the relationship is "dog ride horse", subject being "dog" and object being "horse" would give very little probability to the relation "ride", even though it is the correct answer. Our model alleviates this problem by not mapping visual features directly to the discrete categorical space, but to a continuous embedding space where visual similarity is preserved. Therefore, when seeing the visual features of "dog", "horse" and the whole "dog ride horse" context, our model is able to associate them with a visually similar relationship "person ride horse" and correctly output the relation "ride".

4.4. Ablation Study

4.4.1 Variants of our model

We explore variants of our model in 4 dimensions: 1) the semantic embeddings fed to the semantic module; 2) structure of the semantic module; 3) structure of the visual module; 4) the losses. The default settings of them are 1) using *wiki* + *relco*; 2) 2 semantic layer; 3) with both visual concatenation; 4) with all the 3 loss terms. We fix the other 3

	Relationship Triplet					Relation				
Methods	top1	top5	top10	rr	mr	top1	top5	top10	rr	mr
wiki	15.59	46.03	54.78	52.45	25.31	51.96	78.56	84.38	63.61	8.61
relco	15.58	46.63	55.91	54.03	22.23	52.00	79.06	84.75	63.90	7.74
wiki + relco	**15.72**	**48.83**	**59.87**	**57.53**	**15.08**	**52.00**	**79.37**	**85.60**	**64.12**	**6.21**
wiki + node2vec	15.62	47.58	57.48	54.75	20.93	51.92	78.83	85.01	63.86	7.64
0 sem layer	11.21	28.78	34.84	38.64	43.49	44.66	60.06	64.74	51.60	24.74
1 sem layer	**15.75**	48.23	58.28	55.70	19.15	51.82	78.94	85.00	63.79	7.63
2 sem layer	15.72	**48.83**	**59.87**	**57.53**	**15.08**	**52.00**	**79.37**	**85.60**	**64.12**	**6.21**
3 sem layer	15.49	48.42	58.75	56.98	15.83	**52.00**	79.19	85.08	63.99	6.40
no concat	10.47	42.51	54.51	51.51	20.16	36.96	70.44	80.01	51.62	9.26
early concat	15.09	45.88	55.72	54.72	19.69	49.54	75.56	81.49	61.25	8.82
late concat	15.57	47.72	58.05	55.34	19.27	51.06	78.15	84.47	63.03	7.90
both concat	**15.72**	**48.83**	**59.87**	**57.53**	**20.62**	**52.00**	**79.37**	**85.60**	**64.12**	**6.21**
$\mathcal{L}_y$	15.21	47.28	57.77	55.06	19.12	50.67	78.21	84.70	62.82	7.31
$\mathcal{L}_y + \mathcal{L}_x$	15.07	47.37	57.85	54.92	19.59	50.60	78.06	84.40	62.71	7.60
$\mathcal{L}_y + \mathcal{L}_c$	15.53	47.97	58.49	55.78	18.55	51.48	78.99	84.90	63.59	7.32
$\mathcal{L}_y + \mathcal{L}_x + \mathcal{L}_c$	**15.72**	**48.83**	**59.87**	**57.53**	**15.08**	**52.00**	**79.37**	**85.60**	**64.12**	**6.21**

Table 4: Ablation study of our model on VG80k.

dimensions as the default when exploring one of them.

Which semantic embedding to use? We explore 4 settings: 1) *wiki* and 2) *relco* use wikipedia and relationship-level co-occurrence embedding alone, while 3) *wiki + relco* and 4) *wiki + node2vec* use concatenation of two embeddings. The intuition of concatenating *wiki* with *relco* and *node2vec* is that *wiki* contains common knowledge acquired outside of the dataset, while *relco* and *node2vec* are trained specifically on VG80k, and their combination provides abundant information for the semantic module. As shown in Table 4, fusion of *wiki* and *relco* outperforms each one alone with clear margins. We found that using *node2vec* alone does not perform reasonably, but *wiki + node2vec* is competitive to others, demonstrating the efficacy of concatenation.

Number of semantic layers. We also study how many, if any, layers are necessary to embed the word vectors. As it is shown in Table 4, directly using the word vectors (0 semantic layers) is not a good substitute of our learned embedding; raw word vectors are learned to represent as much associations between words as possible, but not to distinguish them. We find that either 1 or 2 layers give similarly good results and 2 layers are slightly better, though performance starts to degrade when adding more layers.

Are both visual feature concatenations necessary? In Table 4, "early concat" means using only the first concatenation of the three branches, and "late concat" means the second. Both early and late concatenation boost performance significantly compared to no concatenation, and it is the best with both. Another observation is that late concatenation is better than early alone. We believe the reason is, as mentioned in Section 4.3 as well, relations are naturally conditioned on and constrained by subjects and objects, e.g., given "man" as subject and "chair" as object, it is highly likely that the relation is "sit on". Since late concatenation is at a higher level, it integrates features that are more semantically close to the subject and object labels, which gives stronger prior to the relation branch and affects relation prediction more than the early concatenation.

Do all the losses help? In order to understand how each loss term helps training, we trained 3 models of which each excludes one or two loss terms. We can see that using $\mathcal{L}_y + \mathcal{L}_x$ is similar with $\mathcal{L}_y$, and it is the best with all the three losses. This is because $\mathcal{L}_x$ pulls positive x pairs close while pushes negative x away. However, since (x, y) is a many-to-one mapping (i.e., multiple visual features could have the same label), there is no guarantee that the set of x with the same y would be embedded closely, if not using $\mathcal{L}_c$. By introducing $\mathcal{L}_c$, x with the same y are forced to be close to each other, and thus the structural consistency of visual features is preserved.

4.4.2 The margin m in triplet loss

We show results of triplet loss with various values for the margin m in Table 5. As described in Section 3.3 in the paper, this value allows slackness in pushing negative pairs away from positive ones. We observe similar results with previous works [15, 31, 9] that it is the best to set $m = 0.1$ or $m = 0.2$ in order to achieve optimal performance. It is clear that triplet loss is not able to learn discriminative embeddings that are suitable for classification tasks, even with larger m that can theoretically enforce more contrast against negative labels. We believe that the main reason is that in a hinge loss form, triplet loss treats all negative pairs equally "hard" as long as they are within the margin m. However, as shown by the successful softmax models, "easy" negatives

	Relationship Triplet					Relation				
m =	top1	top5	top10	rr	mr	top1	top5	top10	rr	mr
0.1	7.77	29.84	**38.53**	**42.29**	**28.13**	36.50	**63.50**	**70.20**	47.48	14.20
0.2	**8.01**	**27.06**	35.27	40.33	32.10	**37.98**	61.34	69.60	**48.28**	**14.12**
0.3	5.78	24.39	33.26	37.03	34.55	36.75	58.65	64.86	46.62	20.62
0.4	3.82	22.55	31.70	34.10	36.26	34.89	57.25	63.74	45.04	21.89
0.5	3.14	19.69	30.01	31.63	38.25	33.65	56.16	62.77	43.88	23.19
0.6	2.64	15.68	27.65	29.74	39.70	32.15	55.08	61.68	42.52	24.25
0.7	2.17	11.35	24.55	28.06	41.47	30.36	54.20	60.60	41.02	25.23
0.8	1.87	8.71	16.30	26.43	43.18	29.78	53.43	60.01	40.29	26.19
0.9	1.43	7.44	11.50	24.76	44.83	28.35	51.73	58.74	38.89	27.27
1.0	1.10	6.97	10.51	23.57	46.60	27.49	50.72	58.10	37.97	28.13

Table 5: Performances of **triplet loss** on VG80k with different values of margin m. We use margin $m = 0.2$ for all our experiments in the main paper.

	Relationship Triplet					Relation				
λ =	top1	top5	top10	rr	mr	top1	top5	top10	rr	mr
1.0	0.00	0.61	3.77	22.43	48.24	0.04	1.12	5.97	4.11	21.39
2.0	8.48	27.63	34.26	35.25	46.28	44.94	70.60	76.63	56.69	13.20
3.0	14.19	39.22	46.71	48.80	29.65	51.07	74.61	78.74	61.74	10.88
4.0	**15.72**	47.19	56.94	54.80	20.85	51.67	78.66	84.23	63.53	8.68
5.0	**15.72**	**48.83**	**59.87**	**57.53**	**15.08**	**52.00**	**79.37**	**85.60**	**64.12**	**6.21**
6.0	15.32	47.99	58.10	55.57	18.67	51.60	78.95	85.05	63.62	7.23
7.0	15.11	44.72	54.68	54.04	20.82	51.23	77.37	83.37	62.95	7.86
8.0	14.84	45.12	54.95	54.07	20.56	51.25	77.67	83.36	62.97	7.81
9.0	14.81	45.72	55.81	54.29	20.10	50.88	78.59	84.70	63.08	7.21
10.0	14.71	45.62	55.71	54.19	20.19	51.07	78.64	84.78	63.21	7.26

Table 6: Performances of our model on VG80k with different values of the scaling factor. We use scaling factor $\lambda = 5.0$ for all our experiments on VG80k in the main paper.

(e.g., those that are close to positives) should be penalized less than those "hard" ones, which is a property our model has since we utilize softmax for contrastive training.

4.4.3 The scaling factor before softmax

As mentioned in the implementation details of Section 4, this value scales up the output by a value that is close to the average norm of the input and prevents gradient vanishing caused by the normalization. Specifically, for Eq(7) in the paper we use $s(\mathbf{x}, \mathbf{y}) = \lambda \frac{\mathbf{x}^T \mathbf{y}}{||\mathbf{x}||||\mathbf{y}||}$ where λ is the scaling factor. In Table 6 we show results of our model when changing the value of the scaling factor applied before the softmax layer. We observe that when the value is close to the average norm of all input vectors (i.e., 5.0), we achieve optimal performance, although slight difference of this value does not change results too much (i.e., when it is 4.0 or 6.0). It is clear that when the scaling factor is 1.0, which is equivalent to training without scaling, the model is not sufficiently trained. We therefore pick 5.0 for this scaling factor for all the other experiments on VG80k.

4.4.4 Performance trend analysis

Similar with Figure 3 in the paper, in Figure 4 we show top1/top10 and reciprocal rank/mean rank performances as the relation class frequency increases in a log-linear scale. For reciprocal rank and mean rank, we compute them based on the top 250 predictions from each model. This gives a wider view of each model by looking at what position a model ranks the correct answer at. It is clear that our model is superior over baselines in those infrequent classes under the reciprocal/mean rank metric.

4.4.5 Qualitative results

The VG80k has densely annotated relationships for most images with a wide range of types. In Figure 5 and Figure 6 there are interactive relationships such as "boy flying kite", "batter holding bat", positional relationships such as "glass on table", "man next to man", attributive relationships such as "man in suit" and "boy has face". Our model is able to cover all these kinds, no matter frequent or infrequent, and even for those incorrect predictions, our answers are still semantic meaningful and similar to the ground-truth, e.g.,

the ground-truth "lamp on pole" v.s. the predicted "light on pole", and the ground-truth "motorcycle on sidewalk" v.s. the predicted "scooter on sidewalk".

5. Conclusions

In this work we study visual relationship detection at an unprecedented scale and propose a novel model that can generalize better on long tail class distributions. We find it is crucial to integrate subject and object features at multiple levels for good relation embeddings and further design a loss that learns to embed visual and semantic features into a shared space, where semantic correlations between categories are kept without hurting discriminative ability. We validate the effectiveness of our model on multiple datasets, both on the classification and detection task, and demonstrate the superiority of our approach over strong baselines and the state-of-the-art. Future work includes integrating a relationship proposal into our model that would enable end-to-end training.

References

[1] J. Ba, K. Swersky, S. Fidler, and R. Salakhutdinov. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *ICCV*, 2015.

[2] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] B. Dai, Y. Zhang, and D. Lin. Detecting visual relationships with deep relational networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3298–3308. IEEE, 2017.

[4] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. Large-scale object classification using label relation graphs. In *European conference on computer vision*, pages 48–64. Springer, 2014.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*. IEEE, 2009.

[6] M. Elhoseiny, S. Cohen, W. Chang, B. Price, and A. Elgammal. Sherlock: Scalable fact learning in images. In *AAAI*, 2017.

[7] M. Elhoseiny, A. Elgammal, and B. Saleh. Write a classifier: Predicting visual classifiers from unstructured text descriptions. *TPAMI*, 2016.

[8] M. Elhoseiny, B. Saleh, and A. Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *ICCV*, 2013.

[9] F. Faghri, D. J. Fleet, J. R. Kiros, and S. Fidler. Vse++: Improved visual-semantic embeddings. *arXiv preprint arXiv:1707.05612*, 2017.

[10] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov, et al. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013.

[11] Y. Gong, Q. Ke, M. Isard, and S. Lazebnik. A multi-view embedding space for modeling internet images, tags, and their semantics. *IJCV*, 2014.

[12] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

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

[14] J. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[15] R. Kiros, R. Salakhutdinov, R. S. Zemel, and et al. Unifying visual-semantic embeddings with multimodal neural language models. *TACL*, 2015.

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

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[18] Y. Li, W. Ouyang, and X. Wang. Vip-cnn: A visual phrase reasoning convolutional neural network for visual relationship detection. *CVPR*, 2017.

[19] X. Liang, L. Lee, and E. P. Xing. Deep variation-structured reinforcement learning for visual relationship and attribute detection. *arXiv preprint arXiv:1703.03054*, 2017.

[20] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*. Springer, 2014.

[21] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei. Visual relationship detection with language priors. In *ECCV*, 2016.

[22] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

[23] A. Newell and J. Deng. Pixels to graphs by associative embedding. In *Advances in neural information processing systems*, pages 2171–2180, 2017.

[24] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean. Zero-shot learning by convex combination of semantic embeddings. In *ICLR*, 2014.

[25] J. Peyre, I. Laptev, C. Schmid, and J. Sivic. Weakly-supervised learning of visual relations. In *ICCV*, 2017.

[26] J. Peyre, I. Laptev, C. Schmid, and J. Sivic. Weakly-supervised learning of visual relations. *ICCV*, 2017.

[27] B. A. Plummer, A. Mallya, C. M. Cervantes, J. Hockenmaier, and S. Lazebnik. Phrase localization and visual relationship detection with comprehensive image-language cues. In *ICCV*, 2017.

[28] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.

[29] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[30] R. Socher, M. Ganjoo, H. Sridhar, O. Bastani, C. D. Manning, and A. Y. Ng. Zero shot learning through cross-modal transfer. In *NIPS*, 2013.

[31] I. Vendrov, R. Kiros, S. Fidler, and R. Urtasun. Order-embeddings of images and language. *ICLR*, 2016.

[32] L. Wang, Y. Li, and S. Lazebnik. Learning deep structure-preserving image-text embeddings. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[33] D. Xu, Y. Zhu, C. Choy, and L. Fei-Fei. Scene graph generation by iterative message passing. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[34] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh. Graph r-cnn for scene graph generation. *arXiv preprint arXiv:1808.00191*, 2018.

[35] R. Yu, A. Li, V. I. Morariu, and L. S. Davis. Visual relationship detection with internal and external linguistic knowledge distillation. In *The IEEE International Conference on Computer Vision (ICCV)*, 2017.

[36] R. Zellers, M. Yatskar, S. Thomson, and Y. Choi. Neural motifs: Scene graph parsing with global context.

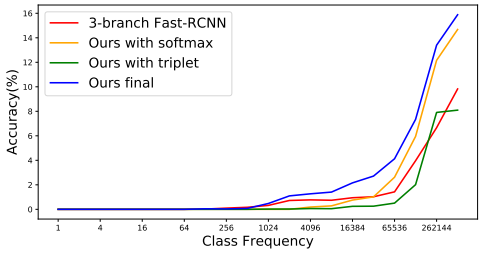
[37] H. Zhang, Z. Kyaw, S.-F. Chang, and T.-S. Chua. Visual translation embedding network for visual relation detection. *arXiv preprint arXiv:1702.08319*, 2017.

[38] H. Zhang, Z. Kyaw, J. Yu, and S.-F. Chang. Ppr-fcn: Weakly supervised visual relation detection via parallel pairwise r-fcn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4233–4241, 2017.

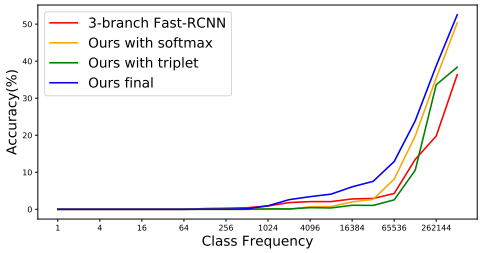
[39] J. Zhang, M. Elhoseiny, S. Cohen, W. Chang, and A. Elgammal. Relationship proposal networks. In *CVPR*, volume 1, page 2, 2017.

[40] H. Zhao, X. Puig, B. Zhou, S. Fidler, and A. Torralba. Open vocabulary scene parsing. *arXiv preprint arXiv:1703.08769*, 2017.

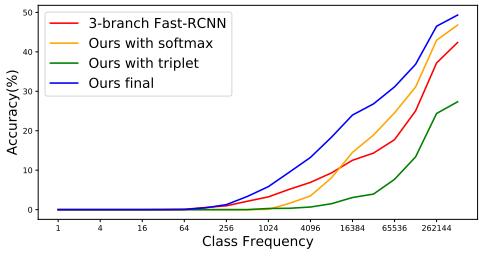
[41] B. Zhuang, L. Liu, C. Shen, and I. Reid. Towards context-aware interaction recognition for visual relationship detection. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.



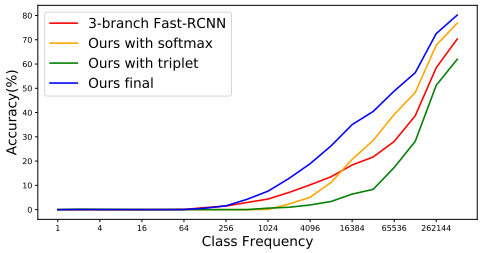
(a) Top 1 rel triplet



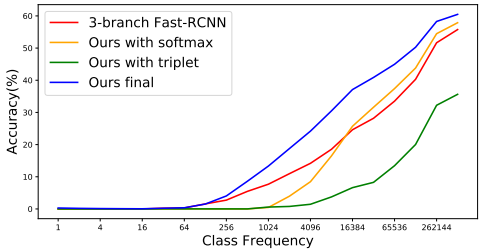
(b) Top 1 relation



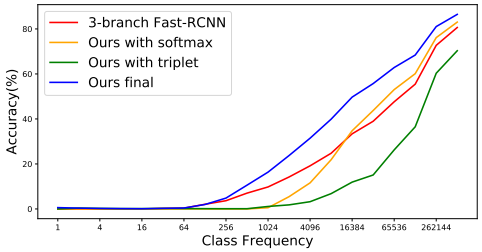
(c) Top 5 rel triplet



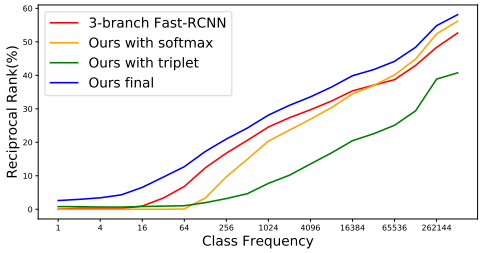
(d) Top 5 relation



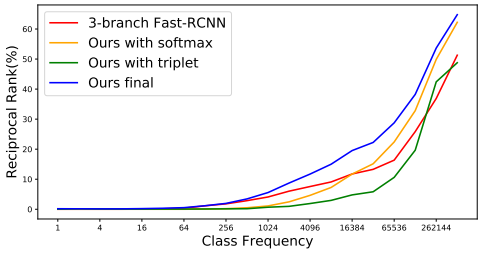
(e) Top 10 rel triplet



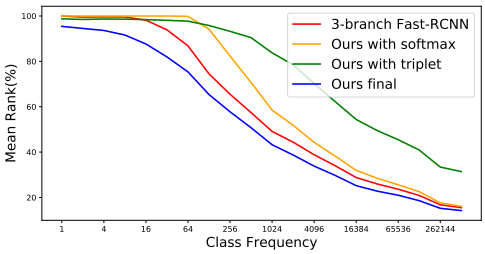
(f) Top 10 relation



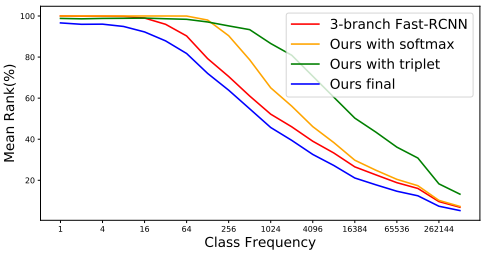
(g) Reciprocal Rank rel triplet



(h) Reciprocal Rank relation



(i) Mean Rank rel triplet



(j) Mean Rank relation

Figure 4: (a)-(f) Top-1,5,10 accuracies and (e)-(h) Reciprocal Rank and Mean Rank of relationship triplets and relations as the relation class frequency increases in a log-linear scale.

Figure 5: Qualitative results. Our model recognizes a wide range of relation ship triples. Even if they are not always matching the ground truth they are frequently correct or at least reasonable as the ground truth is not complete.

Figure 6: Qualitative results (continued). Our model recognizes a wide range of relation ship triples. Even if they are not always matching the ground truth they are frequently correct or at least reasonable as the ground truth is not complete.

Figure 7: Qualitative results (continued). Our model recognizes a wide range of relation ship triples. Even if they are not always matching the ground truth they are frequently correct or at least reasonable as the ground truth is not complete.