“Show me the cup”: Reference with Continuous Representations

Marco Baroni* Gemma Boleda* Sebastian Padó†

*Center for Mind/Brain Sciences

University of Trento

`firstname.lastname@unitn.it`

†Institut für Maschinelle Sprachverarbeitung

Universität Stuttgart

`sebastian.pado@ims.uni-stuttgart.de`

Abstract. One of the most basic functions of language is to *refer* to objects in a shared scene. Modeling reference with continuous representations is challenging because it requires *individuation*, i.e., tracking and distinguishing an arbitrary number of referents. We introduce a neural network model that, given a definite description and a set of objects represented by natural images, points to the intended object if the expression has a unique referent, or indicates a failure, if it does not. The model, directly trained on reference acts, is competitive with a pipeline manually engineered to perform the same task, both when referents are purely visual, and when they are characterized by a combination of visual and linguistic properties.

1 Introduction

Humans use language to talk about the world, and one of its most basic functions is to *refer* to objects [1]. This makes reference one of the fundamental devices to *ground* linguistic symbols in extralinguistic reality [2].¹ For successful reference, the speaker must choose an expression allowing the hearer to pick the right referent. For instance, assume that Adam and Barbara are in the context of Figure 1, and consider the dialogues in (1).

- (1) Adam: Can you please give me...
- a. ...the mug?
Barbara: Sure.
 - b. ...the pencil?
Barbara (*searching*): Ahem, I can’t see any pencil here...
 - c. ...the book?
Barbara: Sorry, which one?

In dialogue (1-a), reference is successful. It fails in (1-b) and (1-c), but for different reasons: in (1-b), the word “pencil” does not apply to any object in the scene; in

¹ We ignore the thorny philosophical issues of reference, such as its relationship to reality. For an overview and references (no pun intended), see [3].

Fig. 1. Example scene.

(1-c), the use of singular “the” implies that Adam refers to a unique object, while the scene contains three matching objects. These examples show how reference involves both *characterization* mechanisms that capture object properties, mainly through the use of content words (e.g. “mug” vs. “pencil”), and *individuation* mechanisms, prominently encoded in function words and morphology (e.g., “the” vs. “some”, singular vs. plural), which allow us to track and distinguish referents.

Existing computational approaches to meaning account for one of these aspects at the expense of the other: Data-driven approaches, including distributional semantic and neural network models, typically model the conceptual level [4], accounting well for characterization, but not for individuation. The converse holds for logics-based approaches [5].

In this paper, we propose a neural network model aimed at both aspects of reference, and that can be trained directly on reference acts. Just like in the typical reference scenario, the model works across modalities, looking for the referent of a verbal expression in the visual world, or in a setting in which entities are characterized by joint visual and linguistic information. The model, *Point-or-Protest (PoP)*, behaves like Barbara: It identifies (*points* to) the image that corresponds to a given linguistic expression, or *protests* in case of reference failure. While the model is generic and could be extended to other reference types, our starting point in this paper is reference to (concrete) entities using single-entity denoting noun phrases (as in (1)). This case clearly illustrates the joint workings of characterization (reference requires recognizing the right sort of entity in the scene) and individuation (reference succeeds only if there is *exactly one* entity of the right kind: in (1), Barbara cannot simply recognize the presence of some “pencil mass”, but she must check that there is only one pencil to unambiguously refer to). We show, in two experiments, that PoP is competitive with a state-of-the-art pipeline requiring specific heuristics.

2 Models

Point-or-Protest Point-or-Protest (PoP) is a feed-forward neural network learning from examples how to react to successful and failed reference acts.² Given a

² For neural network design and training see, e.g., [6].

Fig. 2. The point-or-protest (PoP) model. Network inputs are marked with salmon-pink frames. Uppercase Latin letters represent linear transformations and lowercase Greek letters nonlinearities. $\parallel$ stands for vector concatenation, the period represents dot products, Σ stands for summation across the values of a vector. Vectors containing ellipses may have different dimensionality than the one depicted; among such vectors, those with the same color belong to the same space (have the same number of dimensions). The intensity of a cell’s fill is informally meant to express the size of the value it contains.

variable-length sequence of objects depicted in images (possibly coupled with other information characterizing them, e.g., verbal attributes) and a natural language query, PoP must either *point* to the object denoted by the query, returning its index in the sequence, or *protest* if the query phrase is not an appropriate referring expression. The PoP architecture builds an “entity array” whose entries are vectors storing information about the objects in the scene, and uses similarity-based reasoning about the vectors in the array and the query to decide its response. We currently focus on singular definite article semantics, as in (1), with failure if there is no possible referent (missing-referent anomaly) or if there is more than one (multiple-referent anomaly). We discussed above the linguistic appeal of this case. From a machine-learning perspective, one-entity individuation requires a non-linear separation of the anomalous reference acts (0 or more than 1) from the felicitous ones.

We use the diagram in Figure 2 to introduce PoP. In this example, the input set contains a harrier and two cups, with the corresponding linguistic query being *cup*.³ PoP should thus raise the anomaly flag.

The linguistic query is first mapped to a dense space by using pre-compiled cbow embeddings, whereas images are mapped to vector representations by passing them through a pre-trained convolutional neural network (cnn), and extracting the activation patterns in one of the top layers of the network (see Section 4 for further details). If the input consists of objects with linguistic

³ We do not enter the determiner in the query, since it does not vary across data points: our setup is equivalent to always having “the” in the input. The network learns the intended semantics through training.

attributes, we simply concatenate the corresponding **cnn** and **cbow** vectors to get their input representation, and analogously we concatenate **cbow** vectors to represent multi-word linguistic phrases. Conceptually, using **cbow** embeddings means that the listener we model already possesses large amounts of unembodied knowledge about word meaning, as gathered from linguistic co-occurrence patterns independently of reference. This assumption is unrealistic, and we abandon it with the TRPoP model described below.

PoP maps the input object representations in the sequence to an array of entity vectors by applying a linear transformation. The corresponding mapping matrix $\mathbf{V}$ is shared across objects, as the position of objects in the input sequence is arbitrary, and PoP should not learn associations between objects and specific sequence slots (e.g., from the Figure 2 example, it should not learn to associate cups with positions 2 and 3 in general). Each vector in the entity array corresponds to one input object. In parallel, PoP maps the linguistic expression to a “query” vector through a separate linear transformation $\mathbf{L}$. The query vector lives in the same space as the entity vectors to enable pairwise similarity computations. We can thus interpret the matrices $\mathbf{V}$ and $\mathbf{L}$ as mapping input vectors into a shared multimodal space, in which it is possible to probe visual (or mixed) entities with linguistic queries. Next, the network takes the dot product of the query with each entry in the entity array. The resulting vector (containing as many dimensions as dot products, and thus objects) encodes the similarity profile of the query with the entity vectors: the larger the value in dimension n , the more likely it is that the n -th object in the input sequence is a good referent for the query.

PoP also needs to assess whether the reference act was felicitous. The cumulative “similarity mass” across entity vectors should provide the network with good evidence to reason about anomaly. For the specific aim of modeling singular reference, the network should discover that, when cumulative similarity is too low or too high, the reference is not appropriate for the current sequence: in the first case, because no object matches the query; in the second, because there is more than one object that matches the query. More precisely, along the “anomaly pathway” shown in grey in Figure 2, we first pass the similarity vector through a nonlinearity ψ to sharpen the contrasts, particularly zeroing out low similarities. For example, a **relu** transformation might set all low similarities to 0, making it easier to detect anomalies: in Figure 2, the whitening of the harrier similarity cell is meant to suggest this process. We then sum across all values in the resulting vector, obtaining a cumulative similarity score. We concatenate it with the *cardinality* of the input sequence and feed them, via a linear transformation $\mathbf{A}_i$, to a vector of “anomaly sensor” cells. Cardinality enables the model to take the number of inputs into account when assessing the cumulative similarity score: the same score that looks suspiciously high for two objects is bound to be low for ten objects. More specifically, through cardinality the model can compare the average similarity to arbitrary thresholds, and subsets of anomaly sensor cells can learn different thresholds to pick up anomalies (the presence of multiple anomaly sensor cells allows the model to pick up “non-linear” patterns, such as the one for single-entity reference we are addressing here). Their output is linearly combined

Fig. 3. Example sequences from the Object-Only dataset: Multi-referent anomaly (top) and successful reference (bottom, correct image marked with blue dashed frame).

via matrix $\mathbf{A}_o$ into a single value. The latter is passed through nonlinearity ϕ , that is bounding the anomaly score to approximate a discrete yes/no response.

We finally concatenate the similarity profile with the cell containing the anomaly score, and pass the resulting vector through a `softmax` nonlinearity (π). The model output for an input sequence of n objects will thus contain a probability distribution over $n + 1$ indices. We take the index with the maximum value for this distribution as PoP’s response: if it is one of the first n indices, then PoP “pointed” at the corresponding object, whereas if PoP assigned maximum probability to the $n + 1$ th cell, that means that it “protested”. In the figure diagram, PoP has correctly raised the anomaly flag. We train PoP by backpropagating the error of the log-likelihood cost function when comparing its output (either the index of the correct object, or the anomaly flag) with the ground truth for the training reference acts.

Pipeline As a strong competitor, we implemented a method that performs our task by manual pipelining of a set of separately trained/tuned components. The Pipeline first induces a set of multimodal embeddings by optimizing similarities between matched pairs of queries and objects, compared to random confounders. It uses a max-margin cost function forcing query representations to be (much) more similar to the objects they denote than to irrelevant ones. This has been shown to produce excellent multimodal embeddings [7–9]. Once these embeddings have been separately trained, the model computes similarities between the query and each of the objects in each referential act in our test sets, picking the object with largest similarity as candidate object to point at. Then, two separately-tuned heuristics are used to catch anomalous acts: Missing reference is predicted if no query-object similarity is above an (optimized) threshold. Multiple reference is guessed if the difference between the two largest similarities is below another optimized threshold.

Convolutional Neural Network Since PoP uses input image embeddings based on a pre-trained convolutional neural network (CNN), we also test a model matching the categorical labels produced by the same CNN for the input images against the query. For the example of Figure 2, it would pass each of the images through

the full CNN, obtaining 3 labels. We take a lax approach to label matching, in which the model scores a hit even when, e.g., the gold label is a substring of the model-predicted one. Anomaly detection is straightforward (although again implemented ad hoc): CNN deems a reference act anomalous if no produced label matches the query, or if more than one does. Thus, the CNN would be successful if it predicted a synonym of *cup* for both image 2 and 3.

Tabula Rasa PoP Through the `cbow` vectors, PoP can rely on pre-acquired text-induced word similarity knowledge. The assumption that word meanings are first learned separately, purely from language statistics, and then fine-tuned in the referential setup, is unrealistic. Ideally, we would want a model that learns word representations in parallel from reference acts *and* language statistics. For the time being, we consider instead the other extreme, where word representations are entirely induced from the reference acts during training. The “Tabula Rasa” PoP model (TRPoP) is identical to the one in Figure 2, except that input query representations (and attributes in the Object+Attribute setup explained below) are one-hot vectors. This model will thus induce distributed representations from scratch when estimating the weights of matrices $\mathbf{L}$ and $\mathbf{V}$. Such representations will then depend entirely on the role of words as queries or attributes in the referential acts we model.

3 Data

We test our model in two experiments, for each of which we have automatically created a large-scale dataset. Both datasets contain 40,000 sequences for training, 5,000 for validation and 10,000 for testing, each with 15% missing-referent and 15% multi-referent anomalies. The sequences are of varying length, from 2 to 5 candidate referents. Appendices A and B contain the algorithms used to generate the dataset. Appendix C reports detailed statistics.

Object-Only Experiment. Our first experiment represents a base case of reference, namely matching noun phrases consisting of single nouns with visually represented entities. Figure 3 shows two examples. The objects and images are sampled uniformly at random from a set of 2,000 objects and 50 ImageNet⁴ images per object, itself sampled from a larger dataset used in [10]. As the examples show, we use natural objects and images, which makes the task very challenging (even humans might wonder which image in the second row depicts a *darling*). We generate data with an algorithm sampling sets of sequences with uniform distributions over sequence lengths (2 to 5) and indices of the queried object within a sequence.

Object+Attribute Experiment. Our second experiment, illustrated in Figure 4, goes one step further in testing the model’s individuation capabilities. In the scene from Section 1, imagine that Adam points to the book on top and says “I recommend

⁴ <http://imagenet.stanford.edu/>

query	candidate 1	candidate 2	candidate 3
instruct bartender	inform	instruct	instruct
	procure	need

Fig. 4. Short Object+Attribute example sequences: successful reference (top) and missing-referent anomaly (bottom).

this book”. This linguistically conveyed information will be associated to Barbara’s representation of the entity, together with its visual features. Crucially, it can be used to identify the first book if later on Adam asks her “Can you bring the book I recommended?”. We test this situation in a simplified form. Each referent is associated with both an image and a linguistically-expressed *attribute*, more specifically a verb (the only word class from which we could sample a sufficient number of attributes with the characteristics outlined below). The query and the sequence items are all pairs like *spend:bill*, where we interpret the attribute analogously to an object relative clause, that is, a *bill* that is being *spent* (we ignore tense for simplicity).

We restrict the attributes under consideration for each object to the 500 highest-associated syntactic neighbors of the object according to the DM resource [11], such that the attributes be compatible with the objects (to exclude nonsensical combinations such as *repair:dog*). Of these, we retain only verbs taking the target item as direct object, in line with the “relative clause” interpretation sketched above. Moreover, we focus on (relatively) abstract verbs, for two reasons. First, a concrete verb is more likely than an abstract one to have strong visual correlates that do not match what is actually depicted in an image (cf. *groom:dog* vs. *like:dog*). Second, successful reference routinely mixes concrete and abstract cues (e.g., a noun referring to a concrete object combined with a modifier recording an event associated to it: *the book I lent you*), and we are interested in simulating this scenario. We thus filter verbs through the concreteness norms of [12], retaining only those with a concreteness score of at most 2.5 (on a 1–5 scale).

	Exp 1: Object Only				Exp 2: Object+Attribute			
	Total	Pointing	MissRef	MultRef	Total	Pointing	MissRef	MultRef
PoP	66	71	57	51	69	77	57	46
TRPoP	65	70	58	44	62	70	38	48
Pipeline	67	75	51	45	65	74	37	55
CNN	35	9	100	94	-	-	-	-
Random	17	17	17	17	17	17	17	17
Majority	30	0	100	100	30	0	100	100
Probability	22	18	30	30	22	18	30	30
AttrRandom	-	-	-	-	47	64	16	0
ImgShuffle	-	-	-	-	50	58	31	32

Table 1. Results. Figure of merit is percentage accuracy. See text for details.

The object-attribute structure of the stimuli in this experiment also enables us to introduce challenging confounders into the sequences – namely, pairs that share either the attribute or the object with the query. For each sequence, we start by picking an attribute-object query. Given the query, we generate two more compatible attributes for the query object, and alternative objects compatible to these attributes as well as the initial attribute. Starting from all attribute-object combinations, we randomly drop as many as necessary to obtain the final sequences of 2 to 5 items. A consequence of this design is that the objects within sequences tend to be somewhat related since they share compatible attributes, and vice versa. The first sequence in Fig. 4 illustrates the effect: For the query object *bartender*, we generate the confounder object *soldier*, connected through the attributes *instruct* and *inform*. The full sequence also includes the confounder object *emperor*, not shown in the figure.

4 Experiments

Method PoP and Pipeline’s input word representations are 400-dimensional `cbow` embeddings from [13], trained on about 2.8 billion tokens of unannotated text. These models, as well as TRPoP, use 4096-dimensional vectors as input visual representations, which are produced by passing images through the pre-trained VGG 19-layer CNN of [14] (trained on the ILSVRC-2012 data), and extracting the corresponding activations on the topmost fully connected layer.⁵ The same pre-trained network was used to generate the labels of our CNN competitor model. The parameters of PoP/TRPoP and of the Pipeline max-margin embeddings are estimated by online stochastic gradient descent on the training portions of the two datasets. For Pipeline, we extract all possible pairs of positive and negative query-object tuples from each reference act in the relevant training data. Details on model hyperparameter tuning are in appendix D. We consider three baselines

⁵ We use the MatConvNet toolkit, <http://www.vlfeat.org/matconvnet/>

for both experiments. **Random** assigns all labels randomly. **Majority** assigns the most frequent output label, namely anomaly, accounting for 30% of the sequences (the non-anomalous labels are distributed among predicted indices). **Probability** randomly assigns labels based on their relative frequency in the training data.

Experiment 1: Object-Only Results are reported on the left-hand side of Table 1. Besides overall accuracy (*Total*), we show accuracy itemized by successful reference acts (*Pointing*), missing-referent (*MissRef*) and multiple-referent anomalies (*MultRef*).

(TR)PoP and Pipeline are clearly above the baselines (Majority and Probability reach deceptively high anomaly-detection scores by over-raising the anomaly flag, at the cost of pointing performance). PoP’s absolute performance is close to that of the manually-crafted Pipeline. By jointly learning to point and handling anomalies in reference acts, PoP loses some performance in pointing, but in exchange it does better on anomaly detection. As could be expected, MissRef is more difficult than MultRef for all three models. Interestingly, TRPoP, which does not rely on pre-trained word embeddings, performs comparably to PoP (but it requires more than twice as many epochs to converge, see supplementary materials). This suggests that useful representations of word meaning can be learned solely from examples of successful and failed reference acts.

CNN performance is barely above baseline, and, like Majority and Probability, it trivially reaches high performance on anomaly cases because it raises the anomaly flag whenever it fails to produce the name of the target object (and it rarely produces the right label). For instance, in the first example in Figure 3, CNN can get a hit as long as it doesn’t produce “cup”. As for its extremely low pointing performance, note that CNN, unlike PoP, cannot make reasonable pointing guesses for objects it did not see during training. The large performance asymmetry between these two models sharing the same visual processing network shows that PoP generalizes well beyond the knowledge it inherited from this pre-trained network. Importantly, PoP reasons about similarity in multimodal space, rather than assigning hard labels. For example, the CNN, when presented with an image associated to the out-of-training query *academician*, tags it as *academic gown* – not unreasonably but incorrectly. PoP points to the correct slot because its multimodal *academician* query vector is most similar to the correct entity vector than to the other candidates, with no need to perform explicit label matching. Intriguingly, even when considering the subset of test data that CNN is trained on, we still observe an asymmetry: CNN reaches 58% accuracy, while PoP’s performance is at 67%. This suggests that reference-based training has fine-tuned better representations also for the objects the CNN was explicitly trained for.

Experiment 2: Object+Attribute Results are shown on the right side of Table 1. CNN is not tested here, as it does not handle attributes. PoP’s results are slightly higher than in the previous experiment, while those of TRPoP and Pipeline are slightly lower, such that now PoP is clearly above them. The three models are

exploiting both visual and verbal information, as shown by their comparison to two additional baselines, shown at the bottom of the table. AttrRandom randomly picks one of the objects that shares the attribute with the query, if any, and raises the anomaly flag otherwise. This baseline has, by construction, 0% accuracy on MultRef anomalies, and it performs at random in MissRef detection. However, even in the pointing case, its performance is still well below that of the models. ImgShuffle is a variant of PoP trained after shuffling image vectors, so that each image ID is (consistently) associated with the CNN representation of another image (mostly depicting objects that do not match the image label). The only reliable signal that this baseline can then exploit is attribute information. Again, its performance is clearly below that of the models.

As for anomaly handling, while PoP still finds MissRef easier than MultRef, this time TRPoP and Pipeline actually perform worse on MissRef. Comparing the MissRef cases in which the models failed to raise the anomaly flag, we observe that Pipeline and TRPoP wrongly pointed to an entity sharing the query attribute much more often than PoP (943 and 883 vs. 629). They are thus over-relying on matching attributes, assigning too high a similarity to pairs that are simply sharing the verbal attribute. This also explains their higher performance on MultRef: Attribute sharing makes both repeated referents very similar to the query, triggering the relevant heuristic. Thus, PoP seems better at integrating verbal and visual cues than them. Compared to TRPoP, PoP has an important prior in the semantics encoded in its pre-trained word embeddings, which helps it discover systematic relations between words and objects while keeping the attribute information apart from that of the head noun. Compared to Pipeline, by jointly learning to point and to spot anomalies, it might be able to attain a better balance between visual and verbal information.

To conclude, our model PoP and its variant TRPoP can learn to refer directly from examples. While PoP is not clearly superior to Pipeline, it has a fundamental advantage: It learns to refer in one integrated architecture. Pipeline (as well as CNN) learns to *characterize* objects (e.g., to recognize cups as referents for “cup”), but uses an ad hoc strategy, needing a manually coded heuristic, to simulate *individuating* capabilities (distinguishing cases where there are several or no cups). As soon as the referential setup gets more complex, as in the Object+Attribute experiment in which visual and verbal information need to be combined, the heuristics break down.

5 Related work

Modeling. The PoP model “reasons” about the similarity between a query and a set of candidates in vector space, akin to soft attention mechanisms in recent neural network architectures [15, 16]. While attention is standardly used to retrieve auxiliary information when producing an output, we directly expose the similarity vector as (part of the) output, in order to obtain a model that learns to point robustly across input sequence orders and lengths. The idea of exposing an attention mechanism functioning as a pointer over the input has recently been

employed by [17] in the context of sequence-to-sequence RNNs. PoP’s entity array emulates traditional memory locations within a fully differentiable architecture. This is akin to the memory vectors of the recently proposed Memory Networks framework [18, 19]. However, the Memory Networks array has fixed size, whereas our entity array adapts to input object cardinality.

Multimodal reference resolution. Our task is a special case of reference resolution. Various studies in this area have proposed multimodal approaches jointly handling vision and language, e.g., [20–23]. These papers focus on aspects of the resolution process we are not currently addressing, such as full compositionality or gesture, but they work with very limited perceptual input, such as simple shapes and colours. Probably the most relevant study in this area is the one by [24]. They consider visual scenes with more objects than our sequences, but more limited in nature (tables with 36 puzzle pieces). They handle spatial relations and flexible compositionality. However, they must train a separate classifier for each word in their set, which means that their method can’t process unseen words, and would probably perform badly for words that are not observed frequently enough during training. Moreover, they do not present an integrated architecture for the whole resolution process, as we do, but separate components that are manually combined. Crucially, they assume referring expressions are always felicitous. We are not aware of prior work that, like our Object+Attribute setup, considers referents disambiguated by a mixture of perceptual and verbally-expressed abstract properties.

Referring expression generation and other related work. The task of referring expression generation [25] has recently received new impulses from the study of multimodal language/vision scenarios. The task is converse to ours: given a scene, generate the optimal linguistic expression to pick out a given referent. The focus is generally on considerably more complex (but artificial or heavily controlled) scenes than our sequences, and correspondingly on linguistically more complex referring expressions. Some recent efforts collect and analyze large corpora of referring expressions for multimodal tasks [26, 27]. A method to generate unambiguous referring expressions for objects in natural images has been recently proposed by [28]. Our task is more distantly related to visual question answering [29–31], in the sense that we model one specific type of question that could easily be asked about an image. Even more generally, our approach fits into the multimodal distributional semantics paradigm. See [32] for a discussion of how the problem of reference is addressed in that line of work. There is of course a large body of work on modeling reference with symbolic/logical methods [33], that provides the framework for our problem, but is not directly relevant to our empirical aims. Our task can finally also be seen as a special case of the much broader problem of content-based image retrieval [34].

Acknowledgments

We are grateful to Elia Bruni for the CNN baseline idea, and to Angeliki Lazaridou for providing us with the visual vectors used in the paper. This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 655577 (LOVe) and ERC grant agreement No 715154 (AMORE); ERC 2011 Starting Independent Research Grant n. 283554 (COMPOSES); DFG (SFB 732, Project D10); and Spanish MINECO (grant FFI2013-41301-P). This paper reflects the authors’ view only, and the EU is not responsible for any use that may be made of the information it contains.

References

1. Russell, B.: On denoting. *Mind* **14** (1905) 479–493
2. Harnad, S.: The symbol grounding problem. *Physica D: Nonlinear Phenomena* **42** (1990) 335–346
3. Reimer, M., Michaelson, E.: Reference. In Zalta, E.N., ed.: *The Stanford Encyclopedia of Philosophy*. Winter 2014 edn. (2014)
4. Turney, P.D., Pantel, P.: From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research* **37** (2010) 141–188
5. Bos, J., Clark, S., Steedman, M., Curran, J.R., Hockenmaier, J.: Wide-coverage semantic representations from a CCG parser. In: *Proceedings of the COLING, Geneva, Switzerland* (2004) 1240–1246
6. Nielsen, M.: *Neural Networks and Deep Learning*. Determination Press, New York (2015) Published online: <http://neuralnetworksanddeeplearning.com/>.
7. Frome, A., Corrado, G., Shlens, J., Bengio, S., Dean, J., Ranzato, M., Mikolov, T.: DeViSE: A deep visual-semantic embedding model. In: *Proceedings of NIPS, Lake Tahoe, NV* (2013) 2121–2129
8. Lazaridou, A., Dinu, G., Baroni, M.: Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In: *Proceedings of ACL, Beijing, China* (2015) 270–280
9. Weston, J., Bengio, S., Usunier, N.: WSABIE: Scaling up to large vocabulary image annotation. In: *Proceedings of IJCAI, Barcelona, Spain* (2011) 2764–2770
10. Lazaridou, A., Pham, N., Baroni, M.: Combining language and vision with a multimodal skip-gram model. In: *Proceedings of NAACL, Denver, CO* (2015) 153–163
11. Baroni, M., Lenci, A.: Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics* **36** (2010) 673–721
12. Brysbaert, M., Warriner, A.B., Kuperman, V.: Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods* **46** (2014) 904–911
13. Baroni, M., Dinu, G., Kruszewski, G.: Don’t count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In: *Proceedings of ACL, Baltimore, MD* (2014) 238–247
14. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: *Proceedings of ICLR Conference Track, San Diego, CA* (2015) Published online: <http://www.iclr.cc/doku.php?id=iclr2015:main>.

15. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: Proceedings of ICLR Conference Track, San Diego, CA (2015) Published online: <http://www.iclr.cc/doku.php?id=iclr2015:main>.
16. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. In: Proceedings of ICML, Lille, France (2015) 2048–2057
17. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: Proceedings of NIPS, Montreal, Canada (2015) 2692–2700
18. Sukhbaatar, S., Szlam, A., Weston, J., Fergus, R.: End-to-end memory networks. <http://arxiv.org/abs/1503.08895> (2015)
19. Weston, J., Chopra, S., Bordes, A.: Memory networks. In: Proceedings of ICLR Conference Track, San Diego, CA (2015) Published online: <http://www.iclr.cc/doku.php?id=iclr2015:main>.
20. Gorniak, P., Roy, D.: Grounded semantic composition for visual scenes. *Journal of Artificial Intelligence Research* **21** (2004) 429–470
21. Larsson, S.: Formal semantics for perceptual classification. *Journal of Logic and Computation* **25** (2015) 335–369
22. Matuszek, C., Bo, L., Zettlemoyer, L., Fox, D.: Learning from unscripted deictic gesture and language for human-robot interactions. In: Proceedings of AAIL, Quebec City, Canada. (2014) 2556–2563
23. Steels, L., Belpaeme, T.: Coordinating perceptually grounded categories through language: A case study for colour. *Behavioral and Brain Sciences* **28** (2005) 469–529
24. Kennington, C., Schlangen, D.: Simple learning and compositional application of perceptually grounded word meanings for incremental reference resolution. In: Proceedings of ACL, Beijing, China (2015) 292–301
25. Krahmer, E., van Deemter, K.: Computational generation of referring expressions: A survey. *Computational Linguistics* **38** (2012)
26. Kazemzadeh, S., Ordonez, V., Matten, M., Berg, T.: ReferItGame: Referring to objects in photographs of natural scenes. In: Proceedings of EMNLP, Doha, Qatar (2014) 787–798
27. Tily, H., Piantadosi, S.: Refer efficiently: Use less informative expressions for more predictable meanings. In: Proceedings of the CogSci Workshop on the Production of Referring Expressions, Amsterdam, The Netherlands (2009)
28. Mao, J., Huang, J., Toshev, A., Camburu, O., Yuille, A., Murphy, K.: Generation and comprehension of unambiguous object descriptions. In: Proceedings of CVPR, Las Vegas, NV (2016) In press.
29. Geman, D., Geman, S., Hallonquist, N., Younes, L.: Visual Turing test for computer vision systems. *Proceedings of the National Academy of Sciences* **112** (2015) 3618–3623
30. Malinowski, M., Fritz, M.: A multi-world approach to question answering about real-world scenes based on uncertain input. In: Proceedings of NIPS, Montreal, Canada (2014) 1682–1690
31. Ren, M., Kiros, R., Zemel, R.: Exploring models and data for image question answering. In: Proceedings of NIPS, Montreal, Canada (2015) Published online: <https://papers.nips.cc/book/advances-in-neural-information-processing-systems-28-2015>.
32. Baroni, M.: Grounding distributional semantics in the visual world. *Language and Linguistics Compass* **10** (2016) 3–13
33. Abbott, B.: Reference. Oxford University Press, Oxford, UK (2010)
34. Datta, R., Joshi, D., Li, J., Wang, J.: Image retrieval: ideas, influences, and trends of the new age. *ACM Computing Surveys* **40** (2008) 1–60

A Data Creation for the Object-Only Dataset (Experiment 1)

The process to generate a object sequence is shown in Algorithm 1. We start with an empty sequence and sample the length of the sequence uniformly at random from the permitted sequence lengths (l. 2). We fill the sequence with objects and images sampled uniformly at random (l. 4/5). We assume, without loss of generality, that the the object that we will query for, q , is the first one (l. 6). Then we sample whether the current sequence should be an anomaly (l. 7). If it should be a missing-anomaly (i.e., no matches for the query), we overwrite the target object and image with a new random draw from the pool (l. 9/10). If we decide to turn it into a multiple-anomaly (i.e., with multiple matches for the query), we randomly select another position in the sequence and overwrite it with the query object and a new image (l. 12/13). Finally, we shuffle the sequence so that the query is assigned a random position (l. 14).

Algorithm 1 Creation of Object-Only dataset

Input: Sequence length interval $[i \geq 2, j]$; Set of objects $O = \{o_1, \dots, o_n\}$ and sets of associated images $I(o)$ for each object o ; probability of missing-anomalies P_0 ; probability of multiple-anomalies P_m .

Output: $\langle$ object query q , object-image sequence S $\rangle$

```

1:  $S \leftarrow []$ 
2:  $l \sim U(i, j)$ 
3: for  $k = 1$  to  $l$  do
4:    $o \sim O, i \sim I(o)$ 
5:    $S[k] = \langle o, i \rangle$ 
6:  $q \leftarrow S[1]$ 
7:  $r_0 \sim \text{Bern}(p_0), r_m \sim \text{Bern}(p_0 + p_m)$ 
8: if  $r_0$  then
9:    $o' \sim O, i' \sim I(o)$  so that  $o' \neq q$ 
10:   $S[0] \leftarrow \langle o, i \rangle$ 
11: else if  $r_m$  then
12:   $i \sim U(1, l)$ 
13:   $S[i] \leftarrow \langle o, i' \rangle$  where  $S[1] = \langle o, i \rangle, i' \sim I(o)$ 
14:  $\text{shuffle}(S)$ 
```

B Data Creation for the Object+Attribute Dataset (Experiment 2)

Figure 5 shows the intuition for sampling the Object+Attribute dataset. Arrows indicate compatibility constraints in sampling. We start from the query pair (object 1 – attribute 1). Then we sample two more attributes that are both

Fig. 5. Sampling intuition for Object+Attribute

	Train set avg. frequency				Test set avg. frequency				unseen in test set (%)			
	O	O+I	O+A	O+A+I	O	O+I	O+A	O+A+I	O	O+I	O+A	O+A+I
Object-Only	90.0	2.0	–	–	22.5	1.2	–	–	0.0	23.1	–	–
Object+Attribute	90.9	2.2	8.2	1.1	23.1	1.3	2.7	1.0	0.0	20.2	0.9	82.9

Table 2. Statistics on Object-Only and Object+Attribute datasets. O: object, A: attribute, I: image.

compatible with object 1. Finally, we sample two more objects that are compatible both with the original attribute 1 and one of the two attributes.

Algorithm 2 defines the sampling procedure formally. We sample the first triple randomly (l. 2). Then we sample two compatible attributes for this object (l. 3), and one more object for each attribute (l. 4). This yields a set of six confounders (l. 5–10). After sampling the length of the final sequence l (l. 11), we build the sequence from the first triple and $l - 1$ confounders (l. 12–13), with the first triple as query (l. 14). The treatment of the anomalies is exactly as before.

C Statistics on the Datasets

Table 2 shows statistics on the dataset. The first line covers the Object-Only dataset. Objects occur on average 90 times in the train portion of Object-Only, specific images only twice; the numbers for the test set are commensurately lower. While all objects in the test set are seen during training, 23% of the images are not. Due to the creation by random sampling, a minimal number of sequences is repeated (5 sequences occur twice in the training set, 1 four times) and shared between training and validation set (1 sequence). All other sequences occur just once.

The second line covers the Object+Attribute dataset. The average frequencies for objects and object images mirror those in Object-Only quite closely. The new columns on object-attribute (O+A) and object-attribute-image (O+A+I)

Algorithm 2 Creation of Object+Attribute dataset

Input: Sequence length interval $[i \geq 2, j]$; Set of objects $O = \{o_1, \dots, o_n\}$, sets of associated images $I(o)$ and associated abstract attributes $A(o)$ for each object o ; probability of missing-anomalies P_0 ; probability of multiple-anomalies P_m .

Output: $\langle$ object-attribute query q , object-image-attribute sequence S $\rangle$

```
1:  $S \leftarrow []$ ,  $S_c \leftarrow []$ 
2:  $o_1 \sim O, a_1 \sim A(o_1), i_1 \sim I(o_1)$ 
3:  $a_2, a_3 \sim A(o_1)$  so that  $a_1 \neq a_2 \neq a_3$ 
4:  $o_2 \sim A^{-1}(a_2), o_3 \sim A^{-1}(a_3)$ 
5:  $S_c[1] \leftarrow \langle a_2, o_1, i \sim I(o_1) \rangle$ 
6:  $S_c[2] \leftarrow \langle a_1, o_2, i \sim I(o_2) \rangle$ 
7:  $S_c[3] \leftarrow \langle a_2, o_2, i \sim I(o_2) \rangle$ 
8:  $S_c[4] \leftarrow \langle a_3, o_1, i \sim I(o_1) \rangle$ 
9:  $S_c[5] \leftarrow \langle a_1, o_3, i \sim I(o_3) \rangle$ 
10:  $S_c[6] \leftarrow \langle a_3, o_3, i \sim I(o_3) \rangle$ 
11:  $l \sim U(i, j)$ 
12:  $S[1] \leftarrow \langle o_1, a_2, i_1 \rangle$ 
13:  $S[2..l] \leftarrow$  sample candidates from  $S_c$  w.o. replacement
14:  $q \leftarrow S[1]$ 
15:  $r_0 \sim \text{Bern}(p_0)$ ,  $r_m \sim \text{Bern}(p_0 + p_m)$ 
16: if  $r_0$  then
17:    $o' \sim O, a' \sim A(o), i' \sim I(o)$  so that  $\langle o', a' \rangle \neq q$ 
18:    $S[0] \leftarrow \langle a', o', i \rangle$ 
19: else if  $r_m$  then
20:    $i \sim U(1, l)$ 
21:    $S[i] \leftarrow \langle a, o, i' \rangle$  where  $S[1] = \langle a, o, i \rangle$ ,  $i' \sim I(o)$ 
22: shuffle( $S$ )
```

combinations show that object-attribute combinations occur relatively infrequently (each object is paired with many attributes) but that the combination is considerably restricted (almost no combinations are new in the test set). The full entity representations (object-attribute-image triples), however, are very infrequent (average frequency just above 1), and more than 80% of these are unseen in the test set. A single sequence occurs twice in the test set, all others once; one sequence is shared between train and test.

D Hyperparameter Tuning

We tuned the following hyperparameters on the Object-Only validation set and reused them for Object+Attribute without further tuning (except for the Pipeline heuristics' thresholds). Chosen values are given in parentheses.

- **PoP**: multimodal embedding size (300), anomaly sensor size (100), nonlinearities ψ (**relu**) and ϕ (**sigmoid**), learning rate (0.09), epoch count (14).
- **TRPoP**: same settings, except epoch count (36).

- **Pipeline:** multimodal embedding size (300), margin size (0.5), learning rate (0.09), maximum similarity threshold (0.1 for Object-Only, 0.4 for Object+Attribute), top-two similarity difference threshold (0.05 and 0.07).

Momentum was set to 0.09, learning rate decay to 1E-4 for all models, based on informal preliminary experimentation.