Towards A Model Theory for Distributed Representations

R.V. Guha
Google
guha@guha.com

Abstract
Distributed representations (such as those based on embeddings) and discrete representations (such as those based on logic) have complementary strengths. We explore one possible approach to combining these two kinds of representations. We present a model theory/semantics for first order logic based on vectors of reals. We describe the model theory, discuss some interesting properties of such a system and present a simple approach to query answering.

Introduction

Knowledge Representation based approaches to AI involve encoding knowledge in a logical language and performing logical inference to derive conclusions. Such systems have certain highly desirable properties.

- They are teachable. We can add both specific facts and general axioms/heuristics concisely. E.g., we can just tell them that every human has a biological mother, without having to feed it a large number of examples in the hope that a learning system appropriately generalizes
- There is a well defined notion of entailment, that allows us to draw conclusions from the general axioms we add to the system

These systems, which are usually based on some form of first order logic, are very good for writing axioms to represent (and reason about) complex domains. These axioms are typically hand written, because of which building a broad artificial intelligence using this approach has proven to be rather daunting (Lenat et al. 1990). Completely automating the construction of these systems using learning has also proven difficult. Complex first order statements are extremely hard to automatically learn.

Recent work on distributed representations (Socher et al. 2012, Bowman, Potts, and Manning 2014, Bordes et al. 2011, Le and Mikolov 2014) has explored the use of embeddings as a representation tool. These approaches typically 'learn an embedding', which maps terms and statements in a knowledge base (such as Freebase (Bollacker et al. 2008)) to points in an N-dimensional vector space. Vectors between points can then be interpreted as relations between the terms. The most attractive property of these distributed representations is the fact that they are learnt from a set of examples.

But this benefit comes at the cost of not being able to do some of the things that are relatively trivial for logic based systems.

Goals & Outline of Approach

We would like to have systems that are largely learnt, which we can also teach. In this work we take the first steps towards building a representation system that combines the strengths of logical and distributed representations. The first step is to create a system that has a common representation for both embeddings and logical sentences. The representation needs to be common not just in syntax, but also in terms of semantics, i.e., in what operations can be carried out on them.

Model theory (Enderton 2001) is the mathematical foundation for logic. It tells us what logical sentences may be construed to mean, which operations make sense and what can be said to follow from a set of statements in a knowledge base. We believe that an essential step in bringing logic and distributed representations closer is to create a model theory based on embeddings.

Rather than look at the geometric properties of learnt embeddings as validation of the system having a semantic understanding, we take the concept of an embedding as the starting point and try to build a model theory out of it.

Our model theory is structurally similar to the standard Tarskian semantics for first order logic. Tarskian semantics is based on the concept of an interpretation for a set of statements in a language. An interpretation maps symbols in the language into objects and relations (n-tuples of these objects). In contrast, our interpretations map symbols in the language to points and vectors in an N-dimensional space. Intuitively, a good/correct embedding maps to a single satisfying interpretation. We define satisfaction and entailment as in Tarskian semantics.

This small change (from objects to points) in Tarskian semantics is not enough to reflect object similarity as captured by the geometry of embeddings. To recapture this, we introduce a class of preferred models, where the relative geometric location of objects reflects their similarity. We present an approach to simple inference in these preferred models.
We also discuss how this machinery allows us to combine axioms from a KB with learnt embeddings. This paper is an early exploration along this direction. Much work needs to be done before we can actually build systems based on the approaches described here.

Finally we revisit some old thorny problems that come up in representing common sense knowledge and discuss how a vector space approach might help.

Model Theory

Recap of Tarskian Semantics

For the sake of simplicity, we restrict our attention to logical languages with no function symbols and with only binary predicatbes. We also don’t worry about free variables.

Tarskian semantics for first order logic is based on the concept of an interpretation for a set of logical statements in a language. The interpretation is defined using a model. A model for a first order language assigns an interpretation to all the non-logical constants in that language. More specifically,

1. A model \( M \) specifies a set of objects \( D (d_1, d_2, ...), \) the domain of discourse.

2. To each term \( t_i \) in the language, \( M \) assigns an object in \( M(t_i) \) in \( D \)

3. Each (binary) predicate symbol \( P \) is assigned to a relation \( M(P) \) over \( D^2 \)

A sentence in the language evaluates to \( True \) or \( False \) given a model \( M \) if

1. Atomic formulas: A formula \( P(t_1, t_2) \) evaluates to \( True \) iff \( \langle d_1, d_2 > \in M(P) \)

2. Formulas with logical connectives, such as \( \neg \phi, \phi \rightarrow \psi \) are evaluated according to propositional truth tables

3. \( \exists x \phi(x) \) is \( True \) if there exists some element of \( D, d_i \) for which \( \phi(d_i) \) is true.

4. \( \forall x \phi(x) \) is true if \( \phi(d_i) \) is true for every element of \( D \).

If a sentence \( \phi \) evaluates to \( True \) under a given interpretation \( M \), one says that \( M \) satisfied \( \phi \); this is denoted \( M \models \phi \).

A sentence is satisfiable if there is some interpretation/model under which it is \( True \). A formula is logically valid (or simply valid) if it is \( True \) in every interpretation.

A formula \( \psi \) is a logical consequence of a formula \( \phi \) if every interpretation that makes \( \phi \) \( True \) also makes \( \psi \) \( True \).

In this case one says that \( \psi \) is logically implied by \( \phi \). It is this notion of logical implication that allows us to do inference in knowledge based systems.

Embeddings based Semantics

We now describe how Tarskian semantics can be modified to be based on a vector space model. We do this by using a different kind of model, wherein the domain is a set of points in an \( N \) dimensional vector space of reals. More specifically,

1. A model \( M \) specifies an \( N \) dimensional vector space.

2. To each term \( t_i \) in the language, \( M \) assigns a point \( M(t_i) \) in this vector space

3. Each (binary) predicate symbol \( P \) is assigned to a unit vector \( M(P) \) in \( K \leq N \) dimensions of the vector space

\( P(t_1, t_2) \) evaluates to \( True \) iff the projection of the vector from \( M(t_1) \) to \( M(t_2) \) onto the \( K \) dimensions of \( M(P) \) has the same direction as \( M(P) \).

The definitions for evaluating formulas with logical constants, formulas with quantifiers, of satisfaction and logical entailment are the same as with Tarskian semantics.

Each of our models is also a Tarskian model in a fashion that is consistent with the Tarskian definition of satisfaction, entailment, etc. Consequently, the soundness of classical inference rules (modus ponens, resolution, etc.) carry over.

This kind of model corresponds closely to the kind of embeddings described in (Bordes et al. 2011). In that work, the authors present a mechanism for computing mappings from terms to points in the vector space that is maximally consistent with and predictive of a database of atomic formulas such as those in Freebase. In (Le and Mikolov 2014) the authors use a similar approach to map words (Paris, France, Rome, Italy, etc.) into a vector space so as to maximize skipgram recall. The vectors between pairs such as (Paris, France) tend out to be parallel to those between (Rome, Italy), i.e., are ‘semantically’ meaningful.

We have taken this as a starting point, but instead of treating such embeddings, where terms/words map to points in an \( N \)-dimensional vector space and relations map to vectors, as the target of a learning function, we have used them as the starting point for a model theory.

Preferred Models

Typically, in machine learning based approaches, a number of examples are used to try construct a single model. The goal is to construct the ‘best’ model, one that maximizes some combination of objectives such as precision, recall, compactness, etc.

Logical approaches on the other hand, try to get all the correct predictions (i.e., completeness) while avoiding all wrong predictions (i.e., soundness). To do this they deal not with a single ‘best’ model, but with the set of all satisfying models. The only statements that follow are those that are true in all these models. For example, consider a knowledge base about American history. It will likely contain a symbol like ’AbrahamLincoln’, which the author of the KB intends to denote the 16th American President. The logical machinery doesn’t care if the satisfying models map it to the President, flying cars, real numbers or just the symbol itself. It will draw a conclusion only if the conclusion follows under all these interpretations that satisfy the KB. This is at the heart of soundness in logical inference.

We want something between ‘all the models’ that satisfy the KB and ‘a single’ model that satisfies the KB. Our goal is to create an ensemble of models that are somehow representative of the phenomenon being modelled.

Research in non-monotonic reasoning has explored relaxing the heavy constraint of only drawing conclusions

---

4 If \( M(P) \) is \( N \) dimensions, then if \( P(A, B) \) and \( P(A, C) \), \( B \) will have to be equal to \( C \). Hence \( M(P) \) is in a subspace.
true in all satisfying models. For example, circumscription ([Hintikka 1988], [McCarthy 1980]) allows conclusions that are true in only a preferred subset of satisfying models, those that minimize the extent of certain predicates (typically the ‘ab’ predicates). Such systems sacrifice soundness for the sake of non-monotonicity.

A vector space/embedding model such as [Bordes et al. 2011] is attractive because it gives a geometric interpretation to a set of logical formulas. In particular, terms with embeddings that are near each other are expected to be similar to each other. This geometric property is a result of the mechanism by which embeddings are created.

Unfortunately, though we map terms to points in our models, there is no similarity metric that is built into this mapping. Consequently, that we map objects to points in an N-dimensional space does not mean very much. Points that are extremely close may denote extremely dissimilar terms in a satisfying model.

Further, in order to determine if something is logically entailed by a knowledge base, we have to consider the set of all models that satisfy the knowledge base. Given that different satisfying models might have completely different coordinates for the same term and different vectors for the same predicate, we completely loose the geometric interpretation.

The key here is to introduce some concept of similarity, tied to geometric distance, into our models. Assume that we have a similarity function $S(t_1, t_2)$ which measures the similarity between two terms and evaluates to a number between 0 and 1, with $S(t_1, t_2)$ being closer to 1 if $t_1$ and $t_2$ are more similar. We want models where the distance between the points denoting $t_1$ and $t_2$ is correlated (inversely) to $S(t_1, t_2)$. When this is the case for every pair of points, we have model where the geometry has significance. Let $D(t_i, t_j)$ be the distance between the points that $t_i$ and $t_j$ map to. Then when

$$SD(t_i, t_j) = (1 - S(t_i, t_j))/D(t_i, t_j) \approx 1$$

for every pair of terms, the proximity in the model correlates with similarity between the objects denoted by the terms. There are multiple ways of picking such models. For example, we can minimize

$$(\sum_{t=0}^{L} \sum_{j=0}^{L} \log(SD(t_i, t_j)))/L^2$$

where $L$ is the number of terms. This measures the average disparity between the similarity measure and the distance between (dis)similar objects. The preferred models are those where this average is less than some threshold. Alternatively, we can pick all models where a measure such as $\log(SD(t_i, t_j))$ is less than some threshold for each pair of terms. The particular strategy followed will be a function of the kind of KB we are trying to build.

Model Checking / Inference

As mentioned earlier, since every model in our framework is also Tarskian model and our definition of satisfaction and entailment are the same, every sound proof procedure for first order logic is also sound in our system. Note that they may not be complete.

However, we would like inference mechanisms that are cognizant of preferred models, i.e., that exploit the geometric structure of our models. Much work needs to be done before we have general purpose inference mechanisms, of the sort found in knowledge based systems, that exploit the geometric properties of preferred models.

One approach to approximate inference, that works for domains with a small number of objects, is as follows. We build a representative ensemble of preferred models. Queries are answered by checking the query against each of the models in the ensemble. If the query formula is true (or is true for the same variable bindings) in every model in the ensemble, it is true.

Model Generation

Here, we present a simple approach for generating a set of preferred models that are consistent with a given KB of ground atomic formulae (GAFs). The problem of handling quantified axioms is the subject of future work. For now, we deal with the relatively simple case where we are trying to create a model out of a set of triples which have either been given to us or have already been inferred through forward chaining.

Each triple $P(t_j, t_k)$ gives us the equation:

$$M(t_j) - M(t_k) = M(P_i)$$

where $M(t_j)$ is the location in the N dimensional space of the term $t_j$ and $M(P_i)$ is the vector corresponding to $P_i$. Further, we have a set of constraints based on the similarity function $S(t_j, t_k)$. This is now a fairly generic constraint satisfaction/optimization problem.

Putting it all together

Learning generalizations from a set of ground atomic formulae (GAFs) is the strong point of systems like [Bordes et al. 2011]. Our model theory provides the basis for combining such learning systems with more complex axioms which might be too general/difficult to learn.

The outline of an approach for doing this is as follows. First, we map the learnt embeddings back into GAFs. Then we combine them with the GAFs from our KB to create a larger set of GAFs. We then run the general axioms in a forward direction to generate a much larger set of GAFs. We then use the mechanism described earlier to generate a sample set of preferred models. We can then use model checking to answer queries.

The key here is to have stable mappings between points in the vector space generated by the learning system and terms in the logical KB. This is comparatively straightforward in systems like [Bordes et al. 2011], because the input to the system is a logical KB. We are hopeful that the geometric properties of the vector space (i.e., points close to each other correspond to terms with similar meanings) will allow us to extend this approach to other kinds of embeddings, such as those described in [Le and Mikolov 2014], where the points correspond to words in natural language.
Further Thoughts

The strengths of Knowledge Representation based systems come from the origins of these systems, namely, in mathematical logic. Unfortunately, these origins also bring some unwanted baggage. Mathematical logic was developed for the purpose of stating mathematical truths in a manner where the terms in these statements have precise and unambiguous meaning. Most axioms we add to our knowledge based systems are transliterations of natural language utterances. And as with all such utterances, despite our best attempts, terms and axioms in knowledge based systems end up having many of the characteristics of natural language. In particular, our experiences with systems such as Cyc ([Lenat et al. 1990] and Schema.org [Guha 2011]) have shown that:

- Context dependence: As described in [(Guha 1991), (Guha and McCarthy 2003)] many terms and axioms retain much of the context dependence of words and sentences found in natural language.
- Fluidity of meaning: Many terms allow for considerable fluidity in their meaning. Not just concepts like 'chair', but even terms like 'person' afford a wide range of meanings, substantially complicating axioms referring to them.
- Explosion in the number of terms: Natural language has very powerful (but implicit) mechanisms for composing words to create new concepts. Lacking such built in mechanisms, most large scale knowledge base systems experience an explosion in the number of terms.

The vector space representation very loosely corresponds to a 'semantics' or 'understanding' that the system has. In logical systems, the semantics is only in the meta-theory of the system (i.e., governs what the system should and should not do), not in the system itself. Having a distinct layer that captures the system’s understanding of the symbols is very attractive.

A vector space based model theory offers many different opportunities. Here are a couple of old problems faced by the logical approach to AI, that we might be able to revisit using this tool.

Approximate/Vague Objects

Certain concepts (e.g., the number one) will have a fairly crisp meaning, whereas certain other concepts (e.g., chair), can have a rather broad/vague/approximate meaning. Systems based on logic have found it very difficult to capture this. The continuous nature of the embedding space offers the hope of being able to capture this.

We now introduce the concept of a Aggregate Model, which uses the geometric basis of our model to capture the approximateness that carries over from natural language into terms in our KBs.

Consider the set of points across different preferred models corresponding to a particular term. Consider a cluster of these points (from a subset of the satisfying models) which are sufficiently close to each other. This cluster or cloud of points (each of which is in a different model), corresponds to an aggregate of possible interpretations of the term. We can extend this approach for all the (atomic) terms in the language. We pick a subset of models where every term forms such a cluster. The set of clusters and the vectors between them gives us the aggregate model.

If a model satisfies the KB, any linear transform of the model will also satisfy the KB. In order to keep these transforms from taking over, no two models that form an aggregate should be linear transforms of each other.

In an aggregate model, each object corresponds to a cloud in the N-dimensional space and the relation between objects is captured by their relative positions. The size of the cloud corresponds to the vagueness/approximateness (i.e., range of possible meanings) of the concept. This ability to capture the extend of approximateness of a concept is something that simpler embedding based approaches such as [(Bordes et al. 2011), (Le and Mikolov 2014)] can’t handle.

Beyond discrete denotations

Having a set of structures, distinct from the logical statements, that correspond to the system’s understanding gives us a mechanism for dealing with the variation and context sensitivity in the meaning of terms. The same term, in different statements could map to slightly different points in the vector space, thereby having slightly different meanings.

A vector space based model gives us a generative function for objects. Consider a symbol in the language (e.g., ‘Chair’). In classical semantics, this symbol denotes a single object in a given model. There may be other objects in the model that are very similar, but lacking a term that refers to them. The discreteness of the Tarskian model puts beyond the reach of our language. Attempts to incorporate context into logic ([Guha 1991], [Guha and McCarthy 2003]) allow for different occurrences of a term to refer to distinct objects, but do so at the relatively heavy cost of making them completely different. We are hopeful that the vector space model might give us a tool that gives us a more nuanced control over the denotation space. We feel that this is one of the biggest promises of this approach.

Conclusions

In this paper, we took some first steps towards building a representation system that combines the benefits of traditional logic based systems and systems based on distributed representations. We sketched the outline of a Model Theory for a logic, along the lines of Tarskian semantics, but based on vector spaces. We introduced a class of preferred models that capture the geometric intuitions behind vector space models and outlined a model checking based approach to answering simple queries.

References

[Bollacker et al. 2008] Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In 2008 ACM SIGMOD, 1247–1250. ACM.
[Bordes et al. 2011] Bordes, A.; Weston, J.; Collobert, R.; Bengio, Y.; et al. 2011. Learning structured embeddings of knowledge bases. In AAAI.

[Bowman, Potts, and Manning 2014] Bowman, S. R.; Potts, C.; and Manning, C. D. 2014. Recursive neural networks for learning logical semantics. arXiv preprint arXiv:1406.1827.

[Enderton 2001] Enderton, H. 2001. A mathematical introduction to logic. Academic press.

[Guha and McCarthy 2003] Guha, R., and McCarthy, J. 2003. Varieties of contexts. In Modeling and using context. Springer. 164–177.

[Guha 1991] Guha, R. V. 1991. Contexts: a formalization and some applications, volume 101. Stanford University Stanford, CA.

[Guha 2011] Guha, R. 2011. Introducing schema.org: Search engines come together for a richer web. Google Official Blog.

[Hintikka 1988] Hintikka, J. 1988. Model minimization alternative to circumscription. Journal of Automated Reasoning 4(1):1–13.

[Le and Mikolov 2014] Le, Q. V., and Mikolov, T. 2014. Distributed representations of sentences and documents. arXiv preprint arXiv:1405.4053.

[Lenat et al. 1990] Lenat, D. B.; Guha, R. V.; Pittman, K.; Pratt, D.; and Shepherd, M. 1990. Cyc: toward programs with common sense. CACM 33(8):30–49.

[McCarthy 1980] McCarthy, J. 1980. Circumscription: a form of non-monotonic reasoning. Artificial intelligence 13(1):27–39.

[Socher et al. 2012] Socher, R.; Huval, B.; Manning, C. D.; and Ng, A. Y. 2012. Semantic compositionality through recursive matrix-vector spaces. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, 1201–1211. Association for Computational Linguistics.