Solving Math Word Problems with Minimal Recursion Semantics

Rik Koncel-Kedziorski
University of Washington
kedzior@uw.edu

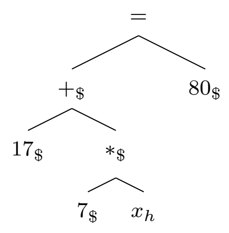
Abstract

This paper provides a method for automatically solving algebra word problems by learning operations between quantities from Minimal Recursion Semantics. The domain-independent semantic representation is acted upon by a general learning method which reduces the reliance on domain-specific features compared to previous semantic-based systems. We compare favorably against a state-of-the-art semantics-based method for solving word problems, recording a 3.0% absolute improvement in operator prediction.

1 Introduction

Automatically solving math word problems is a long-standing challenge for AI (Bobrow, 1964; Charniak, 1969; Seo et al., 2015; Shi et al., 2015). A popular strain of current work focuses on solving algebra word problems which read like small narratives (Roy et al., 2015b; Roy et al., 2015a; Hosseini et al., 2014; Kushman et al., 2014; Koncel-Kedziorski et al., 2015; Zhou et al., 2015). Unlike a child, an NLP system struggles to solve such problems not because of the mechanics of the algebraic manipulations involved, but rather due to the immense difficulty of understanding the narrative which occludes the equational form. These narratives may draw from arbitrary aspects of human life with which school children are familiar, such as harvesting crops or trading Pokemon cards. The open-domain aspect of these problems compounds the difficult of automatically solving them.

Oceanside Bike Rental Shop charges 17 dollars plus 7 dollars an hour for renting a bike. Tom paid 80 dollars to rent a bike. How many hours did he pay to have the bike checked out?



solution : 9 $\qquad 17 + (7 * x) = 80$

Figure 1: Example problem and solution

A recent line of work has focused on a semantics-based solution to the narrative understanding task presented by algebra word problems. Sets of relevant entities are extracted from the text of the problem and the algebraic relationships between these sets are learned from data. Hosseini et al. (2014) uses a state-based model of entities and containers and learns to determine state transitions based on verb categorization. Koncel-Kedziorski et al. (2015) deterministically extracts a recursive formal object called a Qset for each quantity in a word problem text. Qsets are constructed from a host of syntactic dependency relations deemed significant in the solving of math word problems. Learning then takes place over all properties of the Qsets combined by a given operator, as well as global properties of the problem.

Both approaches, however, suffer from a number of scope limitations. Hosseini et al. (2014) can handle only addition, subtraction, and mixed addition subtraction problems. Koncel-Kedziorski et al. (2015) can handle addition, subtraction, multiplication, division, and mixed problems in one variable.

However, template-based methods such as Kushman et al. (2014) and Zhou et al. (2015) can solve all these as well as problems involving simultaneous equations. Additionally, it's difficult to see how either semantic-based method can be extended to solve other narrative understanding problems such as science or reading comprehension questions, or provide for general tasks like character representation or event sequencing.

One reason the semantic-based approaches suffer these scope limitations is due to the "home-brewed" semantic representations used by each. Both works extract unique, domain-specific semantic representations of sets from the dependency parse structure of the problems. These representations significantly reduce the information provided in the text signal, making it difficult to learn the more complex set relationships necessary to solve harder problems.

This paper explores the use of Minimal Recursion Semantics (Copestake et al., 2005) in the solving of algebra word problems. Minimal Recursion Semantics is a rich compositional semantic formalism that can be automatically induced for English sentences via the English Resource Grammar (Flickinger, 2000; Flickinger, 2011). The robustness of this formalism, if utilized well, should significantly reduce the information loss cited above. Moreover, the domain-independence of this semantic formalism will allow for the development of learning methods which can be extended to solve other narrative understanding problems (Bender et al., 2015).

Our (preliminary) method makes use of the entities expressed in the MRS representation of a word problem. The entities which have been quantified with a cardinal number are processed into Number Entities, a lightweight formalism for working with text quantities. Those entities which are the argument of a "much-many_a_rel" predicate are considered variables. We then learn from the predications that take these entities as arguments which operations obtain between them. Our method achieves 87.0% accuracy in predicting +,-,*, or / operations between text quantities, a 3.0% absolute accuracy improvement versus the state-of-the-art classifier used in (Koncel-Kedziorski et al., 2015). We also offer the code publicly, which includes Scala language tools for processing MRS representations of text.

2 Related Work

The current work is situated within a tradition of work on mapping natural language text to executable formal representations (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Ge and Mooney, 2006; Kwiatkowski et al., 2010). More closely related are works on automatic problem solving (Mitra and Baral, 2015; Seo et al., 2014; Hixon et al., 2015). The techniques developed in this line of work are especially relevant to the problem of solving algebra word problems, and merit discussion here.

2.1 Question Answering

A number of systems seek to provide answers for single sentence questions such as "What is the tallest mountain in California?" or "What are social networking sites used for?".

Fader et al. (2013) treats the problem of open domain question answering as a machine learning problem. Their method uses a community-authored question paraphrase database (the WikiAnswers corpus) to learn a semantic lexicon and ranking function. Notably, their method does not require manually annotating questions with semantic forms. Instead, they aggressively generalize a seed lexicon using just 16 question templates.

Weston et al. (2014) show that Memory Networks can be used to improve question answering systems. A memory network consists of a memory, input feature map, a generalization component which updates the memory based on the input, an output layer of the network, and a response - a translation of the output to the desired format (i.e. text). They demonstrate the power of this formalism for modeling natural language on 20 "toy" QA tasks with great success.

Hixon et al. (2015) learn to augment knowledge bases through dialogues with users, resulting in improved question answering in the science domain. Their system learns to relate concepts in science questions to a knowledge graph, and learns to update links and nodes in this graph from open, conversational dialogues with users. In an example dialog, the system, investigating an error in its QA performance, asks the user "What is the relationship be-

```
SENT: How many bales did he store in the barn ?
[ LTOP: h0
INDEX: e2 [ e SF: ques TENSE: past MOOD: indicative PROG: - PERF: - ]
RELS: < [ udef_q_rel<0:14> LBL: h4 ARG0: x5 [ x PERS: 3 NUM: pl IND: + ] RSTR: h6
        BODY: h7 ]
 [ abstr_deg_rel<0:3> LBL: h8 ARG0: i9 ]
 [ which_q_rel<0:3> LBL: h10 ARG0: i9 RSTR: h11 BODY: h12 ]
 [ measure_rel<0:3> LBL: h13 ARG0: e14 [ e SF: prop TENSE: untensed MOOD: indicative ]
        ARG1: e15 [ e SF: prop TENSE: untensed MOOD: indicative ] ARG2: i9 ]
 [ much-many_a_rel<4:8> LBL: h13 ARG0: e15 ARG1: x5 ]
 [ "_bale_n_1_rel"<9:14> LBL: h13 ARG0: x5 ]
 [ pron_rel<19:21> LBL: h16 ARG0: x3 [ x PERS: 3 NUM: sg GEND: m PT: std ] ]
 [ pronoun_q_rel<19:21> LBL: h17 ARG0: x3 RSTR: h18 BODY: h19 ]
 [ "_store_v_cause_rel"<22:27> LBL: h1 ARG0: e2 ARG1: x3 ARG2: x5 ]
 [ _in_p_rel<28:30> LBL: h1 ARG0: e20 [ e SF: prop TENSE: untensed MOOD: indicative ]
        ARG1: e2 ARG2: x21 [ x PERS: 3 NUM: sg IND: + ] ]
 [ _the_q_rel<31:34> LBL: h22 ARG0: x21 RSTR: h23 BODY: h24 ]
 [ "_barn_n_1_rel"<35:41> LBL: h25 ARG0: x21 ] >
HCONS: < h0 qeq h1 h6 qeq h13 h11 qeq h8 h18 qeq h16 h23 qeq h25 > ]
```

Figure 2: MRS representation of a part of a Math Word Problem

tween ‘electricity’ and ‘iron’, if any?” The user’s reply, “iron conducts electricity because its metal”, allows the system to link iron and metal and solve the question correctly.

Jansen et al. (2014) show that discourse theory, both shallow and based in Rhetorical Structure Theory, can be used to improve non-factoid question answering compared to approaches using lexical semantics (in the form of word vectors) alone. They focus on answering manner and reason questions from both Yahoo answers and biology domains. Their results show that discourse-based models that make use of inter-sentential features do better than single-sentence models when the information leading to the correct answer is located in several sentences.

Sachan et al. (2015) also focus on modeling structure to improve question answering. They make use of Latent Structural SVMs to model latent structure between narrative text and multiple choice questions. They test their method over the MCTest (Machine Comprehension Test) dataset. MCTest consists of 650 short narratives each paired with several multiple choice questions which can be answered given only the narrative text (absent any world knowledge). Sachan et al. (2015) include a bevy of features in their model including predicate argument structure, discourse structural features, dependency parse features, semantic role labels, and lexical semantic features. They are able to show that LSSVM outperform Long Short Term Memory neural networks on this task on this dataset.

2.2 Math Word Problems

Closer still to the current work’s line of inquiry are those works who take on the task of solving math word problems.

In two recent works, Seo et al. (2014) and Seo et al. (2015), researchers combine techniques from computer vision with natural language processing and logical inference to solve SAT geometry problems. The vision components are used to parse diagrams which are referenced in the question text. Their work shows that grounding the text in the diagram improves accuracy versus text-only systems.

Shi et al. (2015) provide a method for solving number word problems such as “Nine plus the sum of an even integer and its square is 3 raised to the power of 4. What is the number?” Their method involves training a CFG parser which translates sentences of a number word problem to a meaning representation language called DOlphin Language (DOL). DOL consists of constants, classes, and functions, a semantics which allows for the mathematical computation of a DOL tree. The method for translating a natural language sentence to a DOL tree involves the “semi-automatic” learning of 9600 CFG rules from text to semantic forms.

Roy et al. (2015a) learn to solve multi-step single equation problems using equation trees. They define a canonical form for equations and then learn to identify the least governing intermediate node of any two quantities. However, their canonical form divorces them from the text and destroys the narrative structure of the problem, reducing the value of this approach to the general goal of narrative understanding.

Hosseini et al. (2014) solve elementary addition and subtraction problems by learning verb categories. They ground the problem text to a semantics of *entities* and *containers*, and decide if quantities are increasing or decreasing in a container based upon the learned verb categories. While relying only on verb categories works well for $+, -$, modeling $*, /$ requires going beyond verbs. For instance, "Tina has 2 cats. John has 3 more cats than Tina. How many cats do they have together?" and "Tina has 2 cats. John has 3 times as many cats as Tina. How many cats do they have together?" have identical verbs, but the indicated operation (+ and * resp.) is different.

Kushman et al. (2014) introduce a general method for solving algebra problems. This work can align a word problem to a system of equations with one or two unknowns. They learn a mapping from word problems to equation templates using global and local features from the problem text. However, the large space of equation templates makes it challenging for this model to learn to find the best equation directly, as a sufficiently similar template may not have been observed during training.

Zhou et al. (2015) improve upon the technique of Kushman et al. (2014) by using quadratic programming. Interestingly, a helpful innovation of their work is the reduction of semantic information. Specifically, they no longer pair numbers with variables, significantly reducing the space of possible template alignments and use only numbers in their solution. However, these innovations move the problem only further away from the task of narrative understanding.

2.3 HPSG and MRS

The syntactic theory behind this work is Head-driven Phrase Structure Grammar (Pollard and Sag, 1994). HPSG is a constraint-based grammar which makes use of a large lexicon related by a type hierarchy. The mechanism for composition is based on unification of types. Words or phrases represented by a *typed feature structure* which includes a semantic component.

The semantic formalism used in this work is Minimal Recursion Semantics (Copestake et al., 2005). MRS is a flat semantics which is sufficiently expressive, computationally tractable, and can be integrated with an HPSG implementation. In our case, we use a broad coverage HPSG for the English language which provides parses with MRS semantics called the English Resource Grammar (Flickinger, 2000; Flickinger, 2011). The English Resource Grammar is a hand built HPSG grammar that has been in continual development for over 20 person-years.

Previous work has applied the effective parses and semantic forms provided by the ERG to other problems of Artificial Intelligence. Packard (2014) maps MRS to Robot Control Language instructions. Here manual rules are used to translate between MRS and RCL. Lien and Kouylekov (2015) apply Elementary Dependency Structures, a reduced form of MRS, to the task of recognizing textual entailment. They create a battery of rules for enriching EDS graphs for the purposes of textual entailment, including abstraction, simplification, and structure-improving rules. After such enriched graphs are generated for both the text and hypothesis, entailment is determined by graph subsumption.

3 Overview of Approach

Several methods for solving math word problems work by mapping problem text to underlying equations (Kushman et al., 2014; Koncel-Kedziorski et al., 2015; Roy et al., 2015a). The approach outlined here follows the methods of Koncel-Kedziorski et al. (2015) and (Roy et al., 2015a) in structuring underlying equations in tree form (See Figure 1). The structure of equation trees allows for their decomposition, so that learning can take place locally. Each intermediate node represents some combination of quantities, the result of which is itself a quantity that may enter into further combinations. The text of the problem, if well understood, indicates the correct operations for recursively combining quantities.

This paper focuses on improving operator identification for two problem quantities, be they leaf or intermediate nodes in the equation tree.

Training In the training phase, we first produce a set of equation trees which yield the correct answer for a given word problem utilizing the Integer Linear Program described in Koncel-Kedziorski et al. (2015). This program uses ILP to rank generate the top-k best trees according to a set of possibly soft constraints including equations complexity and type constraints. Quantity "types" are the nouns associated via a dependency parse relation with a numerical value in the problem. An example type constraint is a penalty against multiplying two quantities of the same types. We take the correct equations from the top 100 trees generated by this method.

We then decompose these trees the possible subtrees headed by an intermediate node and labeled with a specific operation *op*. We treat each subtree as a triple consisting of two quantities corresponding to the left and right children and *op*. We associate each quantity with a Number Entity derived from the domain-independent semantic representation of the problem text provided by the ERG. The details of this representation are described in Section 4.1 We then construct a vector from the two Number Entities which serves as positive example for the *op* operation, and a negative sample for all other operators. The details of this vector representation are described in Section 4.2

Inference At inference, we consider an unseen vector and apply the model learned above to determine the correct operator. We report the accuracy of the inference procedure in determining correct operators.

4 Representations

We make use of layers of representation between the surface text of a word problem and the learned model. The first of these is a domain-independent semantic representation derived directly from the text using pre-existing language processing tools. This semantic representation is then transformed into a domain-specific intermediate representation of sets similar to the *Qset* representation used by (Koncel-Kedziorski et al., 2015). Finally, a pair of intermediate representations are used to produce a vector for training. Each step of this process is described below.

4.1 Semantic Representation

The text of each sentence of a word problem is represented as a collection of predicates as provided by the Answer Constraint Engine parser (Packard, 2015). ACE parses text using HPSG grammars, and we use the English Resource Grammar (Flickinger, 2000; Flickinger, 2011), a high precision, hand-built grammar for the English language. The predications instantiate a collection of entity and event variables and express the different interactions of these variables according to the compositional meaning of the text. We track all predications in which a given entity variable appears as an argument. We also track relations between entities appearing as arguments of the same predication.

Relevant to the task of solving math word problems are those entities that are related to numbers. These entities often appear as the ARG1 of a CARD_REL whose CARG (and text span) is a number. If a CARD_REL has a non-entity ARG1, we associate it's CARG to an instantiated entity which shares the same label. These distinguished entities form our domain-specific intermediate representation, which for the sake of simplicity we will call *Number Entities*.

4.2 Vector Representation

As mentioned above, our system learns from triples consisting of two text quantities and an operator which appears in an equation leading to the correct solution. We represent a triple as a vector by considering the Number Entities corresponding to the text quantities in the triple. We evaluate several methods of vectorizing these semantic representations:

All Predicates For each Number Entity we add a 196 dimension vector, the number of predicates associated with any Number Entity in the training data. The vector contains a 1 if the Number Entity appears as an ARGN for $N > 0$ of the corresponding predicate.

Abstract Predicates Due to the nature of predicate naming, specific content noun and verb lexemes are encoded in the predicates, and these can be used to determine the correct operation between Number Entities. For example, problems about *seashells* (and thus "_seashell_n_1_rel") are generally addition or subtraction problems due to the nature of the data sources. Such lexical overlap was shown to be a significant factor for the success of template-based methods; however, it is theoretically unsatisfying as we would like a method which does not "memorize" spurious features of the data but rather learns something more fundamental about the nature of the natural language representation of mathematical operations. To combat this, we consider abstract predicates to be those MRS predicates whose names begin with an underscore (rather than an open-quotation). 45 predicates meet this definition.

Abstract Predicates + Word2Vec Previous work (Koncel-Kedziorski et al., 2015) has shown that verb lexemes, along with distances between the two Number Entity's nouns and verbs respectively improve classification performance. We use Word2Vec (Mikolov et al., 2013) to obtain 200-dimensional representations of verbs and nouns associated with each Number Entity. We incorporate the verb vectors of both Number Entities directly into our feature vector, as well as the cosine distances of their nouns and verbs.

5 Experiments

We evaluate the effectiveness of an MRS-based semantic representation for determining the correct operator for combining two text quantities in a word problem. Our MRS representations come from the ERG version 1214 parsed by ACE parser version 0.9.20. Our data is taken from the SINGLEEQ dataset of 508 math word problems presented in (Koncel-Kedziorski et al., 2015). We take a subset of these where every sentence is parsable and the quantities needed to solve each problem appear in the text (e.g. it is not necessary to resolve "week" to "7 days" in order to solve the problem). So as to leave some problems untouched for future work, we take 3/5ths of the remaining problems for use in this work, leaving us with 191 problems. Using ILP, we obtain 1082 subtrees of correct equations for these problems, which we convert to quantity/operator triples. We then train a multi-class SVC to distinguish operators based on the vector representations of the text quantities described in the previous section. We use LIBSVM version 3.20 (Chang and Lin, 2011) to build an SVC with an RBF kernel. Our results are obtained using the best value of the cost hyper-parameter as discovered through a grid search of the parameter space. We report 5-fold cross validation in all results.

Results Table 1 shows our results. We see that the use of MRS predicates compared the the home-brewed semantics outlined in (Koncel-Kedziorski et al., 2015) yields a 6.9% relative reduction in errors. This is the vector representation that is closest to the semantics as provided by the ERG: all predicates related to number quantities are directly incorporated into the feature vector without domain-specific pruning or manipulation. However, as noted above, the brittleness of such lexical systems has been recorded.

Reducing to only the abstract predicates, we note a significant drop in performance, showing that lexical verb and noun information is necessary in classifying operations. When abstract predicates are augmented by noun and verb distances and verb vectors we see a significant boost, performing better than the "bag-of-predicates" vector.

6 Quantitative Results

The system outlined here performs quite well, even on somewhat difficult questions. For example, for the questions "Mary earns $46 cleaning a home. How many homes did she clean, if she made 276 dollars?" the system is able to correctly identify that "$46" should be multiplied by "x homes" (remember that variables are instantiated by "much-many_a_rel" predicates, and they are associated with the ARG1 entity, here "homes"). Here, the system can use only the distance between the entity noun vectors and the verbs associated with each, as no other abstract predications obtain (besides "much-many_a_rel", common to all variables regardless of operation).

However, the system does still make many significant errors. For the questions "There are 8 calories in a candy bar. How many calories are there in 3 candy bars?" the system erroneously predicts that "x calories" should be subtracted from "3 candy bars", whereas in a correct equation the latter would

System	Accuracy	Rel. Error Reduction (%)
Koncel-Kedziorski	0.840	-
All Predicates	0.851	6.9
Abstract Predicates Only	0.660	-
Abstr. Preds + Verbs & Dist.	0.870	18.8

Table 1: Results of classifying quantity/operator triples using different semantic representations

be divided by the former. This error is in part due to the fact that "3 candy bars" is not associated with any verbal predication, and "x calories" is only associated with a relatively neutral "_be_v_there_rel" verbal predication. However, according to the parse, the ARG1 of the "_in_p_rel" associated with the candy bars is the ARG0 of the verbal predicate associated with the calories. At present, our system is not capable of exploring such connections between event predicates, but we plan to improve this in future work.

7 Conclusion

This paper has presented a method for utilizing Minimal Recursion Semantics for improving a math word problem solving system. We show that with limited adaptation to the math word problem domain, we can achieve a 3.0% absolute improvement in operator prediction accuracy compared to a previous state-of-the-art semantic-based technique. Furthermore, the techniques for relating entities across sentences discussed here are more applicable to other domains besides math word problems due to the general nature of the semantic representation over which they work. Finally, we make our code available to the public to support further development of applications which take advantage of Minimal Recursion Semantics.

References

Emily M. Bender, Dan Flickinger, Stephan Oepen, Woodley Packard, and Ann Copestake. 2015. Layers of interpretation: On grammar and compositionality. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 239–249, London, UK, April. Association for Computational Linguistics.

Daniel G Bobrow. 1964. Natural language input for a computer problem solving system.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27.

Eugene Charniak. 1969. Computer solution of calculus word problems. In *international joint conference on Artificial intelligence*, pages 303–316.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A Sag. 2005. Minimal recursion semantics: An introduction. *Research on Language and Computation*, 3(2-3):281–332.

Anthony Fader, Luke S Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *ACL (1)*, pages 1608–1618. Citeseer.

Dan Flickinger. 2000. On building a more effcient grammar by exploiting types. *Natural Language Engineering*, 6(01):15–28.

Dan Flickinger. 2011. Accuracy vs. robustness in grammar engineering. *Language from a cognitive perspective: Grammar, usage, and processing*, pages 31–50.

Ruifang Ge and Raymond J. Mooney. 2006. Discriminative reranking for semantic parsing. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics*.

Ben Hixon, Peter Clark, and Hannaneh Hajishirzi. 2015. Learning knowledge graphs for question answering through conversational dialog. In *NAACL*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533.

Peter Jansen, Mihai Surdeanu, and Peter Clark. 2014. Discourse complements lexical semantics for non-factoid answer reranking. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Damon Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Annual Meeting of the Association for Computational Linguistics*, pages 271–281.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2010. Inducing probabilistic ccg grammars from logical form with higher-order unification. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1223–1233.

Elisabeth Lien and Milen Kouylekov. 2015. Semantic parsing for textual entailment. *IWPT 2015*, page 40.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Arindam Mitra and Chitta Baral. 2015. Learning to automatically solve logic grid puzzles. In *Proceedings of EMNLP*, Lisbon,Portugal.

Woodley Packard. 2014. Uw-mrs: Leveraging a deep grammar for robotic spatial commands. *SemEval 2014*, page 812.

Woodley Packard. 2015. Ace: the answer constraint engine.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics. The University of Chicago Press and CSLI Publications, Chicago, IL and Stanford, CA.

Subhro Roy, Urbana Champaign, and Dan Roth. 2015a. Solving general arithmetic word problems. In *Proceedings of EMNLP*, Lisbon,Portugal.

Subhro Roy, Tim Vieira, and Dan Roth. 2015b. Reasoning about quantities in natural language. *Transactions of the Association for Computational Linguistics (TACL)*.

Mrinmaya Sachan, Avinava Dubey, Eric P Xing, and Matthew Richardson. 2015. Learning answerentailing structures for machine comprehension. In *Proceedings of ACL*.

Min Joon Seo, Hannaneh Hajishirzi, Ali Farhadi, and Oren Etzioni. 2014. Diagram understanding in geometry questions. In *AAAI*.

Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. 2015. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of EMNLP*, Lisbon,Portugal.

Shuming Shi, Yuehui Wang, Chin-Yew Lin, Xiaojiang Liu, and Yong Rui. 2015. Automatically solving number word problems by semantic parsing and reasoning. In *Proceedings of EMNLP*, Lisbon,Portugal.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory networks. *arXiv preprint arXiv:1410.3916*.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *AAAI*, pages 1050–1055.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *UAI*, pages 658–666.

Lipu Zhou, Shuaixiang Dai, and Liwei Chen. 2015. Learn to solve algebra word problems using quadratic programming. In *Proceedings of EMNLP*, Lisbon,Portugal.