Automatically Solving Word Algebra Problems with Structured Prediction Energy Networks

Arpit Jain
jain@umass.edu

Gota Gando
ggando@umass.edu

Krishna Prasad Sankaranarayanan
ksankaranara@umass.edu

Nikhil Yadav
nikhilyadav@umass.edu

1 Overview

Solving word albegra problems automatically is a long-standing task in natural language processing (NLP). We plan to propose a new approach for solving algebra word problems automatically. In this task, the system needs to find a transformation from the given word problem into a set of equations which correctly represents the input word problem. To tackle this task, we apply structured prediction energy networks (SPENs), which is an energy based model where we can inject the structure knowledge of the task. We plan to start off from a simple feed-forward baseline model, and then evaluate and explore the possibilities of the SPEN model.

2 Introduction

Algebra word problems in general describe some particular world situation and pose a question about it. In this project, we only consider the problems where we can use a system of equations to solve the questions. Figure 2 shows one example of such problems. In this case, we can express the problem as a set of algebraic equations that descrbe the mathematical relationship of entities in the problem: for example $m = 4 \times n$ and $m - 4 = 6 \times (n - 4)$ where m and n represent the age of Maria and Kate, respectively. To answer the question, we need to solve these equations and get the solution. However, we focus on finding a mapping from the word problem into an equation system in this project, as there are many automated solvers for this process.

> Maria is now four times as old as Kate. Four years ago, Maria was six times as old as Kate. Find their ages now.

2.1 Structured Prediction

If we consider each character of the output equations as a random variable, then this task can be considered as a structured prediction task, as every output random variable would be dependent on some other variables. Furthermore, as both the input and output are sequences, we can also interpret this task as a sequence-to-sequnce task.

In many machine learning tasks, we try to predict $\mathbf{y}$ given an input $\mathbf{x}$. In some cases it is sufficient to use a discriminative function F to predict the output such as $\mathbf{y} = F(\mathbf{x})$. However, this model fails to model the structure of the output and the interactions among output components and may perform poor on more complex tasks. We can instead use *an energy function* E that both depends on $\mathbf{x}$ and $\mathbf{y}$ to model such structure and seek the optimal $\mathbf{y}$ that minimizes the energy:

$$\hat{\mathbf{y}} = argmin_{\mathbf{y}} E(\mathbf{y}, \mathbf{x}) \tag{1}$$

Energy function can be seen as a scoring function that returns how compatible $\mathbf{x}$ and $\mathbf{y}$ are, or how likely they occur together. One thing we need to note here is that there is no guarantee of the returning value of energy functions will be in the range of $\{0, 1\}$. One common way to address this problem and calibrate the energy is putting it through the Gibbs distribution.

$$P(Y|X) = \frac{\exp -\beta E(Y, X)}{\int_{y \in (Y)} \exp -\beta E(y, X)} \tag{2}$$

However, it should be noted that this transformation is only possible when computing the integral term $\int_{y \in (Y)} \exp -\beta E(y, X)$ is tractable.

3 Related Works

Solving word algebra problems automatically is a long-standing task in natural language process-

ing (NLP).Solving word algebra problems automatically is a long-standing task in natural language processing (NLP). In this context, solving arithmetic word problems is of specific interest. One key challenge of solving algebra word problems is the lack of fully annotated data. Solving these problems require reasoning and logic across sentence boundaries to find a system of equations that precisely models the described semantic and mathematical relationships.

(Kushman et al., 2014) presented a baseline approach to solving algebra word problems using varied supervision, including either full equations or just the final answers . In this two step algorithm, a template is first selected to match the overall structure of the equation system followed by which it is instantiated with numbers and nouns from the test. In this probabilistic inference mode, the probability of the answer is marginalized over the template selection and alignment.Following template induction wherein labeled equations are generalized to templates, inference is performed y beam search. Beam search provides a mapping to each template based on a canonical ordering. The results on primarily answer annotations (weakly supervised data) were 70%of the accuracies over the complete set of equations which clearly justified the benefits of supervision.

(Roy and Roth, 2016) proposed a novel method for solving multi-step arithmetic word problems without the assistance of predefined annotations and templates. The algorithm is based on multiple classification problems on the target arithmetic equation and then combining these results to obtain the solutions. Grounding is the task ig taking an input word problem in the natural language and representing it as a formal language such as a set of equations, expression trees or states. This paper achieves grounding as a single step process. On the other hand,(Mitra and Baral, 2016), splits this up into two parts. In the first step, the system learns to connect the assertions in a word problems to formula and in the second step, maps the formula into an algebraic equation.

(Koncel-Kedziorski et al., 2015) formalizes solving algebraic word problems as that of generating and evaluating equation trees.Unlike in (Roy, 2017), wherein a system for reasoning about quantities is introduced for arithmetic word problems involving only two values from the text and an arithmetic operator, this method (ALGES) learns to solve complex problems with multiple operands where the space of possible solutions is larger. Quantified sets (Qsets) are used a building block for equation trees to model natural language text quantities. These Qsets are combined with operator to yield a semantically augmented equation tree. With respect to grounding, the Qset properties are extracted and then ordered based on semantic and textual constraints followed by which Qsets are combined with arithmetic operators in an equation tree representation. The inference is to find the most likely equation tree with minimum violation of hard and soft constraints. This is facilitated by calculating a likelihood score for each operand t and the Qsets over which it is operated. Comparison results between template based methods and ALGES show that although the template based methods are able to solve a wider range of problems than ALGES, ALGES fairs well fairs well even in systems with fewer repeated templates or less spurious lexical overlap between problems. In conclusion ALGES is a hybrid of template based and verb categorization methods.

(Upadhyay et al., 2016) demonstrates a state of the art approach for solving algebra word problems with little or no manual annotation of equations. In this novel structured-output learning) algorithm (MixedSP), both explicit (eg., equations) and implicit(eg.,solutions) supervision signals are leveraged jointly.In the case of using algorithms that learn from implicit supervision, the system does not model directly the relation between input x and solution z. Also, multiple combinations of templates and alignments could end up with the same solution making implicit supervision slow, intractable and noisy. To overcome this limitation, MixedSP uses a standard structured prediction update procedure to find the best scoring candidate structures among the templates and alignments. Th algorithm starts with just explicit signals since it leads to a better intermediate model which later allows exploring the output space more efficiently using the implicit signals. The results clearly demonstrate that a joint learning approach benefits from noisy implicit supervision. The accuracies are represented for Explicit, Implicit, Pseudo, Pseudo + Explicit and MixedSP. Mixed SP outperforms it's counterparts even without manual annotation.

4 Problem Definition

Following (Kushman et al., 2014), we define a transformation of word problems to equations as a two step process. First, a template is selected to define the structure of the target equation system. Second, the selected template is instantiated with actual texts for unknowns and values for coefficients. Figure 1 shows an example of template selection and instantiation. Each template has u slots that represent unknown variables, and n slots for coefficients of unknowns.

4.1 Derivation

In this section, we review the formulation described in (Kushman et al., 2014) and (Upadhyay et al., 2016). We denote the input word problem as **x**, and the output as **y**. We call the output as a derivation, which consists of a template T and an alignment A. Therefore we can express the output as $\mathbf{y} = (T, A)$. More specifically, template T is a set of equations $= \{t_0, ..., t_l\}$, and an alignment A is defined as a set of pairs (w, s), where where w is a token in **x** and s is a slot instance in T.

5 Approach

5.1 Baseline model

We plan to use two baseline models. One is a regular feed-forward neural network that predicts the output equations directly from the input, interpreting the task as a black-box sequence-to-sequence task. Another baseline is also a neural network that jointly predicts derivation of templates and its alignments. In both cases, first the input text problem is put through a word embedding layer and a vector is assigned to each word. The first baseline then predicts the output directly after forwarding through several intermediate layers. The second baseline model on the other hand, tries to predict the template structure and its alignments at the same time. This is performed by constructing a derivation vector where the first element denotes the index of the predicted template and the other elements denote the textual or integer values of each unknowns / coefficients in the selected template.

5.2 SPEN

Structured Prediction Energy Networks (SPENs) (Belanger and McCallum, 2015)(Belanger et al., 2017) are one of the energy-based models (EBMs) that performs structured prediction, such as conditional random fields (CRFs), structured perceptrons, and structured SVMs (SSVMs). There are two main differences betwen SPEN and other energy-based models. First, SPENs can learn non-linear energy functions, as it is parameterized by a deep neural network. Similar to other EBMs, SPEN also tries to predict y that minimizes the energy function. The different point is that while EBMs listed above usually assume a restricted graphical structure such as a chain or a tree and consider a linear energy function, SPEN instead considers a general energy function and assumes the non-convexity. Then, it approximates the optimal $\mathbf{y}$ with gradient descent:

$$\bar{\mathbf{y}} = argmin_{\bar{\mathbf{y}}} E(\bar{\mathbf{y}}, \mathbf{x}) \tag{3}$$

$$\mathbf{y}^{t+1} = \mathbf{y}^t - \eta \frac{\partial E}{\partial \mathbf{y^t}} \tag{4}$$

Additionally, SPENs are much more efficient at the inferene stage where the model predicts a candidate output $\bar{\mathbf{y}}$, as SPENs use gradient descent to approximate an optimal $\bar{\mathbf{y}}$ instead of using computationally expensive searching methods such as viterbi algorithm or beam search.

5.3 Architecture of SPEN

Although SPEN can take general energy functions, in this work we consider the energy function consists of the global energy and the sum of local energies as follows:

$$E_x^{local}(\bar{y}) = \sum_{i=1}^{L} \bar{y}_i b_i^T F(x) \tag{5}$$

$$E_x^{global}(\bar{y}) = c_2^T g(C_1 \bar{y}) \tag{6}$$

where g is a non-linearity function, each b_i is a vector of parameters for each label, and the product $C_1\bar{y}$ is a set of learned affine (linear + bias) measurements of the output.

We give the same representation of y as described in the baseline section to this energy network.

6 Evaluation

Following the existing works, we evaluate the model's performance with the overall accuracy on the ALG-514 and DRAW datasets. The predicted equation system is marked as correct if all the characters of the output exactly matches the

Figure 1: The structure of the baseline model. Cited from (Kushman et al., 2014)

ground truth equations. We also plan to evaluate the accuracy on several subsets of word problems to further analyze problems the model can perform well.

7 Datasets

We plan to use two datasets; the ALG-514 dataset proposed by (Kushman et al., 2014) which contains 514 word problems, and the DRAW dataset provided by Microsoft that contains about 1000 independent examples. One issue we are currently aware of these datasets is that ,while they include template structure of output equations for each problem, they do not provide the mapping of unknowns to text values. We plan to manually annotate such alignment information on these datasets so that we can fully make use of the template structure and its alignments.

8 Scope

Although it might be interesting to explore how we can improve the model itself, we limit the scope of this project to applying and evaluating the baseline MLP and SPEN model. Application of the SPEN includes designing structured representation of the output variable y. We are also planning to utilize the "implicit" data samples described in (Upadhyay et al., 2016), which only contain the solutions of equations and not equation themselves if we have time after evaluating the baselines and the SPEN model. In this approach, we hope to jointly learn from data samples with equation labels and without such labels.

References

David Belanger and Andrew McCallum. 2015. Structured prediction energy networks. *CoRR*, abs/1511.06350.

David Belanger, Bishan Yang, and Andrew McCallum. 2017. End-to-end learning for structured prediction energy networks. *arXiv preprint arXiv:1703.05667*.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1821–1831.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *TACL*, 3:585–597.

Nate Kushman, Luke S. Zettlemoyer, Regina Barzilay, and Yoav Artzi. 2014. Learning to automatically solve algebra word problems. In *ACL*.

Yann Lecun, Sumit Chopra, Raia Hadsell, Fu Jie Huang, G. Bakir, T. Hofman, B. Schölkopf, A. Smola, and B. Taskar (eds. 2006. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press.

Arindam Mitra and Chitta Baral. 2016. Learning to use formulas to solve simple arithmetic problems. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2144–2153.

Subhro Roy. 2017. *Reasoning about quantities in natural language*. Ph.D. thesis, University of Illinois at Urbana-Champaign.

Subhro Roy and Dan Roth. 2016. Solving general arithmetic word problems. *arXiv preprint arXiv:1608.01413*.

Shyam Upadhyay, Ming-Wei Chang, Kai-Wei Chang, and Wen-tau Yih. 2016. Learning from explicit and implicit supervision jointly for algebra word problems. In *EMNLP*.