Earth Mover's Distance Pooling over Siamese LSTMs for Automatic Short Answer Grading

Sachin Kumar
Conduent Labs India, Bangalore
sachin.kumar5@conduent.com

Soumen Chakrabarti
IIT Bombay
soumen@cse.iitb.ac.in

Shourya Roy
American Express Big Data Labs
shourya.roy@aexp.com

Abstract

Automatic short answer grading (ASAG) can reduce tedium for instructors, but is complicated by free-form student inputs. An important ASAG task is to assign ordinal scores to student answers, given some "model" or ideal answers. Here we introduce a novel framework for ASAG by cascading three neural building blocks: Siamese bidirectional LSTMs applied to a model and a student answer, a novel pooling layer based on earth-mover distance (EMD) across all hidden states from both LSTMs, and a flexible final regression layer to output scores. On standard ASAG data sets, our system shows substantial reduction in grade estimation error compared to competitive baselines. We demonstrate that EMD pooling results in substantial accuracy gains, and that a support vector ordinal regression (SVOR) output layer helps outperform softmax. Our system also outperforms recent attention mechanisms on LSTM states.

1 Introduction

Grading student work is critical for assessing their understanding, and providing teachers instructive feedback. However, answer grading can become monotonous and tedious for teachers. Computer assisted assessment has been in practice in schools and colleges for many years now, but primarily for recognition questions with constrained answers such as multiple choice questions. Prior research has shown that such recognition questions are deficient in that they do not capture multiple aspects of acquired knowledge, such as reasoning and self-explanation [Conole and Warburton, 2005]. Consequently, open-ended recall questions that seek responses constructed by students are more commonly used in academia. The focus of this paper is automatic grading of such constructed answers with reference to instructor-provided model answers. In particular, we are interested in short answers between a few words and a few sentences long, i.e., everything in between fill-in-the-gap and essay-type answers [Burrows *et al.*, 2015; Roy *et al.*, 2015]. Table 1 shows an example of a short answer grading task.

Grading student-constructed short answers, given instructor-provided model answers, is a complex natural language understanding task owing to linguistic variations (the same answer could be articulated in different ways), the subjective nature of grading (multiple possible correct answers or no correct answer) and lack of consistency in human rating (non-binary scoring on an ordinal scale within a range). Despite state of the art results in most natural language processing tasks, neural models have not been applied extensively for ASAG, barring the use of word embeddings for supervised ASAG [Sakaguchi *et al.*, 2015] and neural networks for essay grading [Alikaniotis *et al.*, 2016]. In this paper, we present a new ASAG system comprising a novel pooling method over Siamese LSTMs, followed by a flexible regression layer for generating a score. Specifically, we make the following contributions.

Question	How are overloaded functions differentiated by the compiler?
Model Answer	Based on the function signature. When an overloaded function is called, the compiler will find the function whose signature is closest to the given function call.
Student#1	It looks at the number, types, and order of arguments in the function call
Student#2	By the number, and the types and order of the parameters.

Table 1: Sample question, model answer, and student answers from an undergraduate computer science course [Mohler *et al.*, 2011].

1.1 EMD Pooling over LSTM States

Although natural language inference (NLI) has some critical differences from ASAG (see Section 2), we first tried two network architectures established in NLI. In the *Siamese* architecture, two networks are applied to two texts, and their final states combined to make predictions [Severyn and Moschitti, 2015]. This has been superseded by a more global comparison across all pairs of states, using various attention mechanisms [Rocktäschel *et al.*, 2015; Yin *et al.*, 2015; He and Lin, 2016]. Surprisingly, attention mechanisms did not perform well for ASAG (see Section 6.4).

Instead, we solve an *Earth Mover Distance* (EMD) problem on a matrix of pairwise distances between each state vector of the model and student answers. EMD is a metric and more generally known as Wasserstein distance [Levina and Bickel, 2001]. EMD has been used between distributional word vectors [Kusner *et al.*, 2015], but not to recurrent states, to our knowledge. Part of the technical challenge is to backprop from ASAG-appropriate losses through the EMD pooling to the LSTMs, in the face of certain aggregate constraints in EMD. This pooling layer may be of independent interest beyond the ASAG task.

1.2 Sinkhorn-Knopp Matrix Scaling

To alleviate the cubic time complexity of EMD calculation, we replace the linear program EMD solver with the Sinkhorn-Knopp matrix scaling [Sinkhorn and Knopp, 1967; Cuturi, 2013] procedure, which replaces EMD with the Sinkhorn distance [Knight, 2008], an excellent approximation. Another potential benefit is that matrix scaling is a differentiable operator [Huang *et al.*, 2016].

1.3 Ordinal Loss Optimization

Another important difference between ASAG and NLI is that the desired output in ASAG is a real or ordinal *score*, not a categorical label (entailment, contradiction, irrelevance) as in NLI. The primary goal in ASAG is *fair grading*, i.e., to minimize numeric difference between gold and system scores. Large correlation usually follows, but is of secondary importance. Therefore we must append a final ordinal prediction layer to the EMD layer. Whereas softmax is most popular for categorical prediction, we found that a two-way hinge loss used in support-vector ordinal regression (SVOR) [Chu and Keerthi, 2007] is superior to a natural non-convex choice [Cheng, 2007].

1.4 Data Augmentation

Training neural models often require large amounts of training data, which is rare among public ASAG datasets. Our primary dataset from [Mohler *et al.*, 2011] contains only about 2,200 (model answer, student answer) pairs. In computer vision, images are readily augmented by scaling, rotation, and changes in brightness and contrast. Here we use a simple but effective method to increase the number of training pairs.

Organization: After reviewing related work in section 2, we present the components of our system in section 3. Details of the earth mover pooling layer are in section 4. Section 5 describes how the EMD pooling output is turned into a score. Section 6 report on experiments on public ASAG benchmarks, and section 7 concludes the paper. We intend to place our code in the public domain.

2 Related Work

In this section, we review prior work in two relevant research streams viz. supervised ASAG and neural models for related NLP tasks with emphasis on NLI.

Supervised ASAG: Early supervised ASAG systems incorporated task- and dataset-specific features into standard classification and regression algorithms [Pulman and Sukkarieh, 2005]. Ensembles of classifiers using different subsets of features [Heilman and Madnani, 2013] are widely used. Features are extensively tuned towards specific datasets, and new techniques are rarely evaluated on datasets published earlier. A comprehensive overview of prior supervised ASAG systems can be found in two recent survey papers and references therein [Burrows *et al.*, 2015; Roy *et al.*, 2015]. These surveys have also emphasized the importance of standardized evaluation across datasets. [Ramachandran *et al.*, 2015] proposed a graph based approach to extract patterns from groups of questions and their answers towards constructing regular expression alike patterns and showed better performance than the Tandalla's winning entry in [Kaggle, 2012b].

Recent ASAG efforts have leveraged advancements in distributional word semantics and neural methods for embedding words. Word level similarities between student and model answers based on pre-trained word vectors [Mikolov *et al.*, 2013b] have been used as features to train regression and classification models [Sakaguchi *et al.*, 2015; Sultan *et al.*, 2016]. [Alikaniotis *et al.*, 2016] introduced *score-specific word embeddings* in LSTMs to embed student and model essays to single state vectors. These were turned into an output score via linear regression with square loss.

Natural Language Inference (NLI): In the NLI task [MacCartney, 2009], we are given a premise sentence (e.g., The cat is playing Frisbee on the beach) and have to judge if a hypothesis (e.g., A pet is using a plastic toy on sand; or An elephant is bathing in the river) is entailed by the premise, or contradicts it, or is unrelated. Thus, a special case is recognizing textual entailment (RTE) [Sammons *et al.*, 2011]. A public NLI corpus was recently released [Bowman *et al.*, 2015]. Their baseline approach represented the premise and hypothesis as concatenated word embedding vectors, which were input to a multi-layer LSTM-RNN architecture that outputs one of the above classes.

Various Siamese networks with tied weights have been used to compare or label pairs of short texts. [Severyn and Moschitti, 2015] used Siamese convnets to match candidate answer passages to queries. [Mueller and Thyagarajan, 2016] used Siamese LSTMs for NLI. In recent proposals [Rocktäschel *et al.*, 2015; Wang and Jiang, 2015; Liu *et al.*, 2016], each hypothesis token *attended to* specific focus tokens in the premise sentence. Some NLI approaches [Yin *et al.*, 2015; He and Lin, 2016] compute attention from a matrix of pairwise interactions like us, but these are then used to predict categorical NLI labels.

Critical differences between NLI/RTE and ASAG: ASAG has often been connected to NLI/RTE by regarding the student answer as premise and the model answer as hypothesis [Ostermann *et al.*, 2015; Mohler *et al.*, 2011]. However the differences between the tasks are stronger than their apparent commonality. [Ostermann *et al.*, 2015] conducted an entailment annotation exercise with fine grained labels and compared with scores assigned by teachers. They found:

- Not all RTE tags could be mapped to ASAG scores. ASAG scores are typically ordinal, e.g., on a 5-point scale whereas RTE tasks have categorical labels such as 'entailed', 'not entailed', and 'contradictory'
- Partial entailment instances had poor agreement or correspondence with human-provided ASAG scores.

In ASAG, key concepts in the model answer must be *covered* by the student answer, but the concepts may be diffused over many tokens, and extra material in the answer is not necessarily a disqualification. A student answer which is more specific than the model answer may not entail the latter, but nonetheless is likely to be scored as correct. Finally, the organizers of the "student response analysis" (SRA) task in SemEval-2013 workshop concluded that the correlation between answer assessment judgments and entailment judgments is not perfect.[1]

Finally, there are two other NLP tasks which are related to ASAG. In automatic essay scoring (AES), the objective is to score essays on the basis of composition, fluency, grammatical correctness etc. Unlike ASAG, AES evaluates essays

[1] https://www.cs.york.ac.uk/semeval-2013/task7/

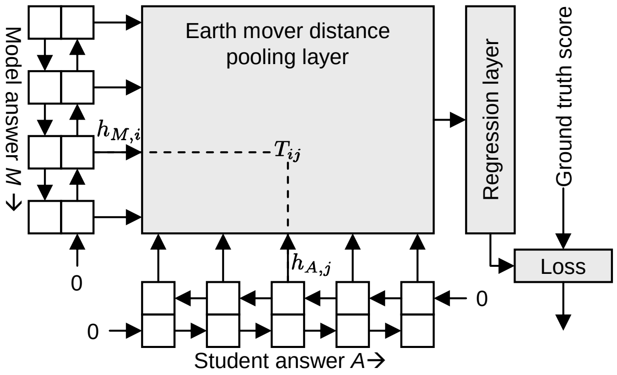

Figure 2: High-level view of our ASAG system.

as per "absolute" criteria, not relative to instructor-provided model answers [Kaggle, 2012a]. In paraphrase detection (PD), the objective is to detect if two sentences or passages have the same meaning [Socher *et al.*, 2011] can also be seen as similar to ASAG. However, unlike PD, ASAG is asymmetric and there may not be a one-to-one correspondence between concepts in model and student answers.

3 System Overview

Figure 2 gives a high-level view of our system. The components are described in detail in sections 4 and 5. During inference, the system gets a question, associated with a model answer M and a student answer A, and has to estimate a non-negative real or ordinal score y_A. During training, gold y_A are provided. M and A are modeled as sequences of tokens. Tokens are mapped to pre-trained (300-dimensional) word embeddings [Mikolov *et al.*, 2013b] and input to a conventional Siamese bi-LSTM [Hochreiter and Schmidhuber, 1997] with tied parameters, which we call LSTM$_M$ and LSTM$_A$.

Suppose there are ℓ_M tokens in model answer M and ℓ_A tokens in student answer A. Bi-LSTM LSTM$_M$ emits $\boldsymbol{h}_M = (h_{M,i} : i \in [1, \ell_M])$ where $h_{M,i} = [\overrightarrow{h}_{M,i}, \overleftarrow{h}_{M,i}]$ is a concatenation of the forward and backward states at each token position $i \in [1, \ell_M]$. Similarly, LSTM$_A$ emits $\boldsymbol{h}_A = (h_{A,j} = [\overrightarrow{h}_{A,j}, \overleftarrow{h}_{A,j}])$ at each token position $j \in [1, \ell_A]$. In our implementation each h vector is in $\mathbb{R}^{50+50}$. As Figure 2 shows, $\boldsymbol{h}_M$ and $\boldsymbol{h}_A$ are input into a pooling layer described in section 4. The output of the pooling layer goes to a flexible regression layer, discussed in section 5.

4 Earth Mover Distance (EMD) Pooling Layer

At this stage, many Siamese LSTMs would compare/combine state vectors $\overrightarrow{h}_{M,\ell_M}$ and $\overrightarrow{h}_{A,\ell_A}$ (and possibly $\overleftarrow{h}_{M,1}$ and $\overleftarrow{h}_{A,1}$) to predict a label. But, building on our intuition about ASAG thus far, we inject all of $\boldsymbol{h}_M$ and $\boldsymbol{h}_A$ into a "crossbar" type pooling layer, defined by the earth mover distance (EMD). We first define the distance between two hidden states: $d_{ij} = \|h_{M,i} - h_{A,j}\|^2$.

Next, we define earth mover distance as a minimization over an auxiliary transport matrix $\boldsymbol{T} \in \mathbb{R}^{\ell_M \times \ell_A}$:

$$\mathrm{EMD}(\boldsymbol{h}_M, \boldsymbol{h}_A) = \min_{\boldsymbol{T} \geq \boldsymbol{0}} \sum_{i=1}^{\ell_M} \sum_{j=1}^{\ell_A} T_{ij} d_{ij} \tag{1}$$

$$\text{where} \quad \textstyle\sum_i T_{ij} = 1/\ell_A \quad \text{and} \quad \sum_j T_{ij} = 1/\ell_M. \tag{2}$$

Intuitively, EMD is defined by a minimal *fractional transportation* between the state vectors in $\boldsymbol{h}_M$ and $\boldsymbol{h}_A$. [Kusner *et al.*, 2015] calculated EMD between word embedding sequences, but not to LSTM states that capture long distance semantics in the answers. In fact, in section 6, we show that EMD between words is inferior to that between LSTM states.

4.1 Alternating Optimization

We need to backprop errors through the EMD pooling network to the LSTM weights Θ, because $\boldsymbol{h}_M, \boldsymbol{h}_A$, and, by extension, d_{ij} change as training progresses. However, in (1) and (2), $\boldsymbol{T}$'s dependence on Θ is not in a closed form. We address this problem using alternating optimization. Every half-step, we fix Θ, their hidden states $\boldsymbol{h}_M, \boldsymbol{h}_A$, and therefore d_{ij}s. Then we optimize $\boldsymbol{T}$ using LP. In the other half-step, we fix $\boldsymbol{T}$ and backprop the loss into Θ. For reasonable choices of the regression and loss layers in Figure 2 (see section 5.3), the loss becomes differentiable wrt Θ.

4.2 Sinkhorn Approximation

Unfortunately, the time complexity of solving the EMD optimization via LP is $O(d^3 \log d)$ where $d = \max\{\ell_M, \ell_A\}$ [Kusner *et al.*, 2015]. This becomes a bottleneck for training the network, because this computation needs to be performed for every single update.

To alleviate the cubic time complexity of EMD, we follow the approach of [Cuturi, 2013] to add an entropy regularization term to the transport objective (1). For any fixed $\lambda > 0$, the regularized transportation problem is defined as

$$\min_{T \geq \boldsymbol{0}} \sum_{i=1}^{\ell_M} \sum_{j=1}^{\ell_A} T_{ij} \|h_{M,i} - h_{A,j}\|^2 + \tfrac{1}{\lambda} T_{ij} \log T_{ij}, \tag{3}$$

subject to the constraints (2). The larger the λ, the closer this relaxation is to the original EMD. This modified optimization is strictly convex, and an efficient matrix scaling algorithm [Sinkhorn and Knopp, 1967; Cuturi, 2013] can be used to solve it in $O(d^2)$ time, giving the *Sinkhorn distance*, a good approximation to EMD. It may be possible to backprop through the matrix scaling steps to the LSTMs as well [Huang *et al.*, 2016], but this is left for future work.

5 Output Regression Layer and Loss Design

The EMD pooling layer outputs a real scalar, but this does not directly correspond to answer grades/score. In this section we discuss alternatives to obtain a score from the LSTM or EMD outputs. The best choice may depend on how the ASAG system will be evaluated.

5.1 Regression to Continuous Score (Baseline)

A standard Siamese LSTM system would compare $\overrightarrow{h}_{M,\ell_M}$ with $\overrightarrow{h}_{A,\ell_A}$. These can be combined into a real-valued answer score predictor. E.g., [Mueller and

Thyagarajan, 2016] use the following transformation: $\exp\left(-\left\|\overrightarrow{h}_{M,\ell_M}-\overrightarrow{h}_{A,\ell_A}\right\|_1\right)$ which is large if the LSTM states are similar, and small if they are different. The final loss can be L1 or L2 (mean square) error between the above prediction and ground truth. (More about the choice of losses in section 6.3.)

5.2 Ordinal Labels with Non-convex Loss (Logits)

In a standard multi-class classification problem with categorical labels $1, \ldots, K$ (no order between them), neural networks output a multinomial distribution $(p_k : k \in [1, K])$ from a last *softmax* stage, whose cross-entropy $\mathrm{KL}(p||\cdots)$ from a one-hot gold distribution $(\ldots, 0, 1, 0, \ldots)$ is used as the loss. In case of ordinal regression [Cheng, 2007; Gutiérrez *et al.*, 2016], the gold label representation changes from one-hot to "prefix-hot": if the label is k, the first k elements of the vector are ones, and the suffix all zeros, i.e., $(1, \ldots, 1, 0, \ldots, 0)$. The goal of the regression stage is to learn a function that maps the input (EMD, say) to a vector $o = (o_1, o_2, ..., o_k, ...o_K)$, where $o_i \approx 1$ if $i \leq k$ and $o_i \approx 0$ if $i \geq k$. $\|o\|_1$ is an estimate of k itself, rather than 1.

Suppose the pooled output in Figure 2 is $z \in \mathbb{R}$. For EMD pooling, larger z implies smaller k and vice versa, so we will negate z in the following. We include in the trainable parameters $b_1 \leq b_2 \leq \cdots \leq b_K$, and predict output vector $p_k = \sigma(-z - b_k)$, where $\sigma(\bullet) = 1/(1 + e^{-\bullet})$ is the sigmoid function. As loss between p and ground truth, we can use L1 or L2 or KL divergence. During inference, to make a prediction, our method scans $p_1, p_2, \ldots, p_K$ in order. It stops when p_k drops below some predefined threshold, or k reaches K.

5.3 Ordinal Labels with Convex Loss (SVOR)

Instead of using $b_1, \ldots, b_K$ as the transition points for sigmoids, we can adopt standard convex formulations for ordinal regression [Herbrich *et al.*, 1999; Chu and Keerthi, 2007] and use them to define consecutive ranges within which the EMD output must fall to emit successive ordinal scores.

To ease notation, let $-\infty = b_0 \leq b_1 \leq \cdots \leq b_K = +\infty$, where only $b_1, \ldots, b_{K-1}$ are learnable parameters, inducing K intervals in $\mathbb{R}$. If the gold label is k, then $-z$ should lie in $[b_{k-1}, b_k]$. Tacking on a margin, we define the "tub loss" as a two-sided hinge loss around $[b_{k-1} + 1, b_k - 1]$:

$$\mathrm{tub}(z, k; \boldsymbol{b}) = \begin{cases} b_{k+1} + z, & -z \leq b_{k-1} + 1, \\ -z - (b_k - 1), & -z \geq b_k - 1, \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

The alternating training protocol of section 4.1 can now be formally specified as the **LSTM-EMD-SVOR** algorithm.

```
Initialize tied LSTM parameters $\Theta$ and regression
  parameters $\boldsymbol{b}$
**while** validation error reduces **do**
    **for** each labeled instance $\langle M, A, k \rangle$ **do**
        Input $M, A$ to LSTMs and calculate $d_{ij}$s
        Fix $\{d_{ij}\}$, find optimal $\boldsymbol{T}$ via matrix scaling
        Fix $\boldsymbol{T}$, calculate $z$, then $\mathrm{tub}(z, k; \boldsymbol{b})$ loss
        Backprop to update $\Theta$ and $\boldsymbol{b}$
```

During inference, $\Theta, \boldsymbol{b}$ are fixed. We input M, A as before, compute z, and then report k such that $-z \in [b_{k-1}, b_k]$. Depending on the optimizer, we can enforce $b_{k-1} \leq b_k$ directly, or by a suitable change of variable, like $b_{k-1} + e^{c_k} = b_k$.

6 Experiments

We evaluate our system on two grading tasks that we call CS Dataset and SemEval Dataset, described in section 6.1. In section 6.2 we describe how we augmented them to better train our networks. Subsequently we describe different evaluation metrics used, in section 6.3. The rest of this section presents accuracy numbers comparing with relevant prior art.

6.1 Datasets

We use two datasets for reporting experimental results which we describe next.

CS dataset: This is one of the earliest ASAG datasets consisting of 80 questions from 12 assignments of an undergraduate "Data Structure" course leading to total $2,273$ student responses [Mohler *et al.*, 2011].[2] Each student answer was evaluated on a scale of $[0, 5]$ with 0.5 interval leading to 11 permissible scores. Student answers were independently evaluated by two annotators, with their average designated as the gold label. We train the proposed network with the ordinal-loss based final layer on this dataset for each assignment by training on the remaining 11 assignments. Before data augmentation (section 6.2), the average size of training set (in 12-fold cross validation) is about $2,100$ student responses.

SemEval dataset: This dataset is a part of the "Student Response Analysis" (SRA) in the Semantic Evaluation (SemEval) workshop in 2013 [Dzikovska *et al.*, 2013].[3] We use the SciEntsBank subset of the data which contains approximately $10,000$ answers to 197 assessment questions from 15 different science domains. The answers were graded by multiple annotators on a nominal scale viz. 'Correct', 'Partially correct/incomplete', 'Contradictory' (student answer contradicts the reference answer), 'Irrelevant' and 'non domain'.

Owing to its ordinally incomparable labels, we cannot report numbers that are directly comparable to published numbers on this task. Nevertheless, we adapt it to compare between the alternative network architectures we have proposed. We use the ordinal labels 'Correct', 'Partially correct/incomplete', and 'Irrelevant' and assign them ordinal labels 2, 1 and 0, to be able to report MAE, RMSE and correlation as in the Mohler data set.

SemEval test protocols: The test set is divided into three subsets with varying degrees of similarity with the training examples. The Unseen Answers (UA) dataset consists of responses to questions that are present in the training set. Unseen Questions (UQ) contains responses to in-domain but previously unseen questions. Three of the fifteen domains were

[2] http://web.eecs.umich.edu/~mihalcea/downloads/ShortAnswerGrading_v2.0.zip

[3] https://www.cs.york.ac.uk/semeval-2013/task7/index.php\%3Fid=data.html

held out for a final Unseen Domains (UD) test set, containing completely out-of-domain question-response pairs.

6.2 Training Data Augmentation

Public ASAG datasets are rarely large enough to effectively train a large number of model parameters in complex neural networks. Vision and speech [Cui *et al.*, 2015] labeling tasks augment training data using label-preserving transformations. [Zhang and LeCun, 2015] used a thesaurus derived from Wordnet to replace some words from text with their synonyms and thereby increasing training data size. Here we describe our data augmentation method. A substantial number of the answers in our data-set are given perfect scores as per the instructors. We hypothesize that the student answers which received perfect scores by the instructor are equivalent to instructor-provided model answers. For example, if out of n students m received perfect scores ($m < n$) then by above strategy we can generate $m \times (n-1)$ new (model answer, student answer) training pairs. Augmenting the Mohler data set results in an increase from about 2500 to 35000 training pairs. Augmenting the SemEval data set increases training pairs from about 5000 to 78000. During testing, we use only the model answer provided by the instructor.

6.3 Evaluation Metrics

ASAG systems that predict ordinal labels have been evaluated using a variety of metrics, in two broad categories: value deviation and correlation.

Mean absolute error (MAE): If y^* is the gold score and y is the system-generated score, this is simply $|y - y^*|$, averaged over instances.

Root mean square error (RMSE): If $i \in [1, I]$ indexes instances, this is $\sqrt{(1/I)\sum_i (y_i - y_i^*)^2}$.

Correlations: Here the goal is to compare the orders imposed by gold and system scores over all answers. Pearson's r is the most popular correlation coefficient in ASAG.

While both absolute and correlation measures have been used in ASAG, we argue that absolute deviation makes more sense, because we are interested in fair assessment, rather than merely ordering students. Further, MAE is believed to be superior to RMSE [Willmott and Matsuura, 2005], in part because of nonuniform scaling: errors < 1 shrink, while errors > 1 expand when squared. Nevertheless, we report all the measures for easy comparison with relevant prior art.

The upper block of Table 3 gives baseline numbers. Tf-idf is the simple TFIDF similarity between M and A. Lesk is a standard WordNet-based similarity (see [Mohler *et al.*, 2011]). The numbers for [Mohler *et al.*, 2011] are from that paper. We ran code provided by [Yin *et al.*, 2015] and [He and Lin, 2016] on our ASAG data. We also tried the attention model of [Rocktäschel *et al.*, 2015], but failed to get any better baseline results than the ones shown. The second block shows ablation studies on our network architecture. The lower block shows the best design choices.

6.4 Ablation Studies

Although exhaustively covering all possible combinations in {Word2vec, ConvNet, LSTM} × {last state, all states} × {Avg, Max, EMD} × {SoftMax, L2, Logits, SVOR} would be too tedious, we compare some important points in this design space.

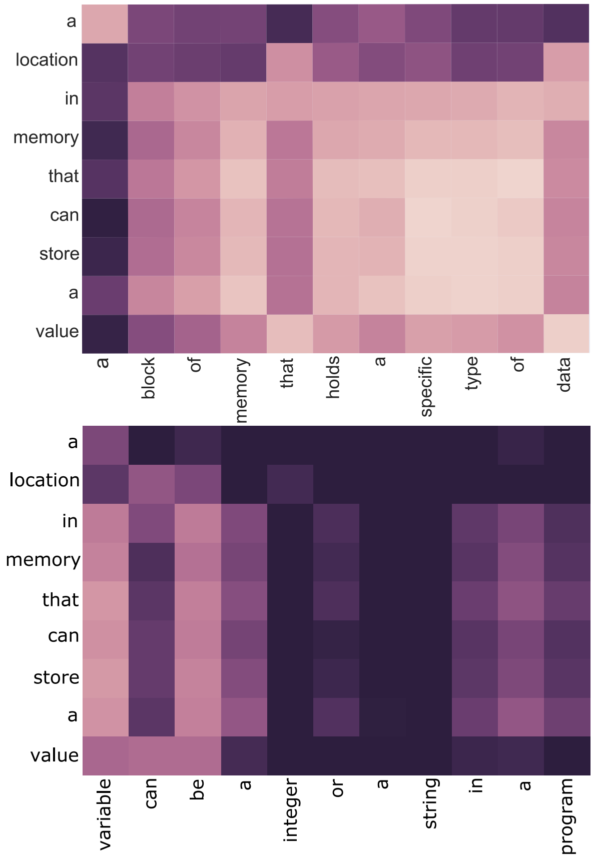

Figure 4: EMD heatmaps for good (score 5/5, above) and poor (score 2/5, below) answers. The good answer shows large values of T_{ij} (bright color) in many cells. The bad answer shows small values (dark colors). Words of M and A are shown along the margins.

Choice of Output Stage: In table 3, using L2 loss at the output generally results in poorer performance. Results are mixed among the best two choices in rows 13 and 14. SVOR is much better than Logits for MAE and RMSE, but not as good for correlation, which may be because our loss function is not ranking-oriented. Either may be used, depending on whether the assessment goal is accurate scoring or ranking.

Last-state vs. All states: LSTM-Last-L2 is the only row where the last state vector was used, and is generally inferior to most methods where all states were used (except with L2 output loss). To better understand the value of information from all states, we plotted T_{ij} from sample M, A pairs as heatmaps in Figure 4, for one good and one poor answer. The good heatmap has large bright patches representing high transportation values (i.e., LSTM state matches), where as the poor heatmap has large dark patches. Last-state regression score roughly corresponds to the brightness of the bottom right corner cell T_{ℓ_M, ℓ_A}. The heatmaps show that even a good (respectively, poor) answer may have a relatively low (respectively, high) value of this cell. Therefore, incorporating signals from all intermediate states is important. Attention-based networks also try to capture this, but they have many

	Pooling input	Which state	Pooling	Output stage	System name	MAE	RMSE	Pearson's r
1					Tf-idf		1.022	0.327
2					Lesk		1.050	0.450
3					[Mohler *et al.*, 2011]		0.978	0.518
4	Convnet	All	Avg	SoftMax	ABCNN, [Yin *et al.*, 2015]	0.74	0.92	0.52
5	Convnet	All	Max	SoftMax	[He and Lin, 2016]	0.75	0.87	0.61
6	LSTM	Last		L2	LSTM-Last-L2	0.91	1.101	0.600
7	LSTM	All	EMD	L2	LSTM-EMD-L2	0.96	1.28	0.46
8	LSTM	All	Max	L2	LSTM-MaxPool-L2	1.12	1.60	0.411
9	LSTM	All	Avg	L2	LSTM-AvgPool-L2	1.16	1.58	0.393
10	Word2vec		EMD	SVOR	W2V-EMD-SVOR	0.77	1.073	0.355
11	LSTM	All	Max	SVOR	LSTM-MaxPool-SVOR	0.83	0.973	0.522
12	LSTM	All	Avg	SVOR	LSTM-AvgPool-SVOR	0.63	0.95	0.571
13	LSTM	All	EMD	SVOR	LSTM-EMD-SVOR	**0.490**	**0.830**	0.550
14	LSTM	All	EMD	Logits	LSTM-EMD-Logits	0.657	1.135	**0.649**

Table 3: Performance on Mohler CS dataset with 12-fold training (lower is better for RMSE and MAE; higher is better for Pearson's r). We assess various combinations of input stage, choice of state/s to compare, pooling logic, and regression stage.

System	MAE			RMSE			Pearson's r		
	UA	UQ	UD	UA	UQ	UD	UA	UQ	UD
W2V-EMD-SVOR	0.749	0.806	0.732	0.845	1.096	1.131	0.211	0.125	0.189
LSTM-Last-L2	0.777	0.803	0.790	1.010	1.026	1.038	0.076	0.087	0.137
LSTM-EMD-Logits	0.561	0.761	0.796	0.780	1.065	1.136	0.434	0.134	0.187
LSTM-EMD-SVOR	**0.469**	**0.738**	**0.705**	**0.758**	**0.996**	**0.958**	**0.554**	**0.157**	**0.237**

Table 5: Performance on SemEval dataset (lower is better for RMSE and MAE; higher is better for Pearson's r). For test protocols UA, UQ, and UD, see section 6.1.

more parameters which are more delicate to optimize compared to our frugal model (rows 4 and 5 in Table 3).

EMD over words vs. LSTM states: At this point one may be convinced that a whole sequence-to-sequence match is important, but it would be tempting to try a simpler alternative to LSTM, namely, the word mover distance of [Kusner *et al.*, 2015] which computes EMD across stateless sequences of word embeddings [Mikolov *et al.*, 2013a]. But, comparing row 10 against the most closely comparable rows 13 and 14 of Table 3, we see clear evidence that EMD pooling over stateless word embeddings is not nearly as good as EMD pooling over LSTM states.

6.5 Comparison with Current Systems

Comparing rows 3 and 13 in Table 3 shows that we perform better than [Mohler *et al.*, 2011]. They employ a support vector regression machine that predicts scores using a set of dependency graph alignment and lexical similarity measures while our method uses no handcrafted features.

[Ramachandran *et al.*, 2015] adopt a different evaluation setup. For each assignment/test, they use 80% of the data for training and the rest as test. This setup thus enables in-domain model training. Their system trains a random forest regressor based on features derived from automatically generated regexp patterns to capture semantic variations and syntactic structures of good answers. Results in this setup are shown in Table 6. Our model performs better on RMSE and matches on correlation (they did not report MAE).

6.6 SemEval Performance

Table 5 compares some of the system variations in Table 4.1 for the SemEval data. This data set has many tied gold and systems scores, which explains the low correlation across the board. However, for MAE and RMSE, the trends seen in the Mohler CS data are more-or-less preserved here, with LSTM-EMD-SVOR emerging superior to other combinations.

System	MAE	RMSE	r
[Ramachandran *et al.*, 2015]	-	0.86	0.61
LSTM-EMD-SVOR	0.42	**0.77**	0.61

Table 6: Performance on Mohler 2011 dataset with in-domain training (lower is better for RMSE and MAE; higher is better for Pearson's r).

6.7 Effect of λ

Table 7 shows how MAE and time (for one EMD or Sinkhorn calculation) varies with λ. We chose $\lambda = 10$ for all experiments as a robust compromise.

λ	MAE	EMD time	Sinkhorn time
20	0.026	0.024	0.012
10	0.0306	0.024	0.0093
1	0.1414	0.024	0.0019
0.1	2.0098	0.024	0.0004

Table 7: Trade-off between accuracy and speed via λ.

7 Conclusion

In this paper, we proposed a novel neural architecture for automatic short answer grading (ASAG). Our system combines Siamese bi-LSTMs, a novel pooling layer based on the Sinkhorn distance between LSTM state sequences, and a support vector ordinal output layer. Training is enhanced via a task-specific data augmentation strategy. Experiments on two publicly available ASAG datasets established that our system has scoring accuracy superior to recent baselines.

References

[Alikaniotis *et al.*, 2016] Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. Automatic Text Scoring Using Neural Networks. *CoRR*, abs/1606.04289, 2016.

[Bowman *et al.*, 2015] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

[Burrows *et al.*, 2015] Steven Burrows, Iryna Gurevych, and Benno Stein. The Eras and Trends of Automatic Short Answer Grading. *International Journal of Artificial Intelligence in Education*, 25(1):60–117, 2015.

[Cheng, 2007] Jianlin Cheng. A neural network approach to ordinal regression. *CoRR*, abs/0704.1028, 2007.

[Chu and Keerthi, 2007] Wei Chu and S. Sathiya Keerthi. Support Vector Ordinal Regression. *Neural Comp.*, 19:792–815, 2007.

[Conole and Warburton, 2005] Gráinne Conole and Bill Warburton. A review of computer-assisted assessment. *Research in learning technology*, 13(1), 2005.

[Cui *et al.*, 2015] Xiaodong Cui, Vaibhava Goel, and Brian Kingsbury. Data augmentation for deep neural network acoustic modeling. *Transactions on Audio, Speech and Language Processing*, 23(9):1469–1477, 2015.

[Cuturi, 2013] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, pages 2292–2300, 2013.

[Dzikovska *et al.*, 2013] Myroslava O. Dzikovska, Rodney D. Nielsen, Chris Brew, Claudia Leacock, Danilo Giampiccolo, Luisa Bentivogli, Peter Clark, Ido Dagan, and Hoa T Dang. SemEval-2013 task 7: The Joint Student Response Analysis and 8th Recognizing Textual Entailment Challenge. Technical report, DTIC Document, 2013.

[Gutiérrez *et al.*, 2016] Pedro Antonio Gutiérrez, Maria Perez-Ortiz, Javier Sanchez-Monedero, Francisco Fernandez-Navarro, and Cesar Hervas-Martinez. Ordinal regression methods: survey and experimental study. *IEEE TKDE*, 28(1):127–146, 2016.

[He and Lin, 2016] Hua He and Jimmy Lin. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *NAACL-HLT*, pages 937–948, 2016.

[Heilman and Madnani, 2013] Michael Heilman and Nitin Madnani. ETS: Domain Adaptation and Stacking for Short Answer Scoring. In *Proceedings of the 2nd joint conference on lexical and computational semantics*, volume 2, pages 275–279, 2013.

[Herbrich *et al.*, 1999] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Support vector learning for ordinal regression. In *ICANN*, pages 97–102, 1999.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[Huang *et al.*, 2016] Gao Huang, Chuan Guo, Matt J Kusner, Yu Sun, Fei Sha, and Kilian Q Weinberger. Supervised word mover's distance. In *NIPS*, pages 4862–4870, 2016.

[Kaggle, 2012a] Kaggle. The Hewlett Foundation: Automated Essay Scoring. https://www.kaggle.com/c/asap-aes, 2012.

[Kaggle, 2012b] Kaggle. The Hewlett Foundation: Short Answer Scoring. http://www.kaggle.com/c/asap-sas, 2012.

[Knight, 2008] Philip A Knight. The sinkhorn–knopp algorithm: convergence and applications. *SIAM Journal on Matrix Analysis and Applications*, 30(1):261–275, 2008.

[Kusner *et al.*, 2015] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. From Word Embeddings To Document Distances. In *ICML*, volume 37, pages 957–966, 2015.

[Levina and Bickel, 2001] Elizaveta Levina and Peter J. Bickel. The earth mover's distance is the mallows distance: Some insights from statistics. In *ICCV*, pages 251–256, 2001.

[Liu *et al.*, 2016] Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. Learning natural language inference using bidirectional LSTM model and inner-attention. *CoRR*, abs/1605.09090, 2016.

[MacCartney, 2009] Bill MacCartney. *Natural language inference*. PhD thesis, 2009.

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

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

[Mohler *et al.*, 2011] Michael Mohler, Razvan C. Bunescu, and Rada Mihalcea. Learning to Grade Short Answer Questions using Semantic Similarity Measures and Dependency Graph Alignments. In *ACL*, pages 752–762, 2011.

[Mueller and Thyagarajan, 2016] Jonas Mueller and Aditya Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In *AAAI*, pages 2786–2792. AAAI Press, 2016.

[Ostermann *et al.*, 2015] Simon Ostermann, Andrea Horbach, and Manfred Pinkal. Annotating Entailment Relations for Shortanswer Questions. In *ACL-IJCNLP*, page 49, 2015.

[Pulman and Sukkarieh, 2005] Stephen G. Pulman and Jana Z. Sukkarieh. Automatic Short Answer Marking. In *Proceedings of the Second Workshop on Building Educational Applications Using NLP (EdAppsNLP)*, pages 9–16, 2005.

[Ramachandran *et al.*, 2015] Lakshmi Ramachandran, Jian Cheng, and Peter Foltz. Identifying Patterns for Short Answer Scoring using Graph-based Lexico-semantic Text Matching. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 97–106, 2015.

[Rocktäschel *et al.*, 2015] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*, 2015.

[Roy *et al.*, 2015] Shourya Roy, Y. Narahari, and Om D. Deshmukh. A Perspective on Computer Assisted Assessment Techniques for Short Free-Text Answers. In *Computer Assisted Assessment*, pages 96–109. Springer, 2015.

[Sakaguchi *et al.*, 2015] Keisuke Sakaguchi, Michael Heilman, and Nitin Madnani. Effective Feature Integration for Automated Short Answer Scoring. In *NAACL-HLT*, pages 1049–1054, 2015.

[Sammons *et al.*, 2011] Mark Sammons, VG Vydiswaran, and Dan Roth. *Recognizing textual entailment. Multilingual Natural Language Applications: From Theory to Practice*. Prentice-Hall, 2011.

[Severyn and Moschitti, 2015] Aliaksei Severyn and Alessandro Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *ACM SIGIR*, pages 373–382, 2015.

[Sinkhorn and Knopp, 1967] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967.

[Socher *et al.*, 2011] Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In *NIPS*, pages 801–809, 2011.

[Sultan *et al.*, 2016] Md. Arafat Sultan, Cristobal Salazar, and Tamara Sumner. Fast and Easy Short Answer Grading with High Accuracy. In *NAACL-HLT*, pages 1070–1075, 2016.

[Wang and Jiang, 2015] Shuohang Wang and Jing Jiang. Learning natural language inference with LSTM. *arXiv preprint arXiv:1512.08849*, 2015.

[Willmott and Matsuura, 2005] Cort J. Willmott and Kenji Matsuura. Advantages of the Mean Absolute Error (MAE) over the Root Mean Square Error (RMSE) in Assessing Average Model Performance. *Climate research*, 30(1):79–82, 2005.

[Yin *et al.*, 2015] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193*, 2015.

[Zhang and LeCun, 2015] Xiang Zhang and Yann LeCun. Text understanding from scratch. *arXiv preprint arXiv:1502.01710*, 2015.