Investigating neural architectures for short answer scoring

Brian Riordan[1], Andrea Horbach[2], Aoife Cahill[1], Torsten Zesch[2], Chong Min Lee[1]
[1]Educational Testing Service, Princeton, NJ 08541, USA
[2]Language Technology Lab, University of Duisburg-Essen, Duisburg, Germany

Abstract

Neural approaches to automated essay scoring have recently shown state-of-the-art performance. The automated essay scoring task typically involves a broad notion of writing quality that encompasses content, grammar, organization, and conventions. This differs from the *short answer content scoring* task, which focuses on content accuracy. The inputs to neural essay scoring models – ngrams and embeddings – are arguably well-suited to evaluate content in short answer scoring tasks. We investigate how several basic neural approaches similar to those used for automated essay scoring perform on short answer scoring. We show that neural architectures can outperform a strong non-neural baseline, but performance and optimal parameter settings vary across the more diverse types of prompts typical of short answer scoring.

1 Introduction

Deep neural network approaches have recently been successfully developed for several educational applications, including automated essay assessment. In several cases, neural network approaches exceeded the previous state of the art on essay scoring (Taghipour and Ng, 2016).

The task of *automated essay scoring* (AES) is generally different from the task of automated *short answer scoring* (SAS). Essay scoring generally focuses on *writing quality*, a multidimensional construct that includes ideas and elaboration, organization, style, and writing conventions such as grammar and spelling (Burstein et al., 2013). Short answer scoring, by contrast, typically focuses only on the accuracy of the content of responses (Burrows et al., 2015). Analyzing the rubrics of prompts from the Automated Student Assessment Prize shared tasks on AES and SAS, while there is some overlap across essay scoring and short answer scoring, there are three main dimensions of differences:

1. *Response length.* Responses in SAS tasks are typically shorter. For example, while the ASAP-AES data contains essays that average between about 100 and 600 tokens (Shermis, 2014), short answer scoring datasets may have average answer lengths of just several words (Basu et al., 2013) to almost 60 words (Shermis, 2015).

2. *Rubrics* focus on content only in SAS vs. broader writing quality in AES.

3. *Purpose and genre*. AES tasks cover persuasive, narrative, and source-dependent reading comprehension and English Language Arts (ELA), while SAS tasks tend to be from science, math, and ELA reading comprehension.

Given these differences, the feature sets for AES and SAS systems are often different, with AES incorporating a larger set of features to capture writing quality (Shermis and Hamner, 2013). Nevertheless, deep learning approaches to AES have thus far demonstrated strong performance with minimal inputs consisting of unigrams and word embeddings. For example, Taghipour and Ng (2016) explore simple LSTM and CNN-based architectures with regression and evaluate on the ASAP-AES data. Alikaniotis et al. (2016) train score-specific word embeddings with several LSTM architectures. Dong and Zhang (2016) demonstrate that a hierarchical CNN architecture produces

strong results on the ASAP-AES data. Recently, Zhao et al. (2017) show state-of-the-art performance on the ASAP-AES dataset with a memory network architecture.

In this work, we investigate whether deep neural network approaches with similarly minimal feature sets can produce good performance on the SAS task, including whether they can exceed a strong non-neural baseline. Unigram embedding-based neural network approaches to essay scoring capture content signals from their input features, but the extent to which they capture other aspects of writing quality rubrics has not been established. These approaches as implemented would seem to lend themselves even better to the purely content-focused rubrics in SAS, where content signals should dominate in achieving good human-machine agreement. On the other hand, recurrent neural networks may derive some of their predictive power in AES from more redundant signals in longer input sequences (as sketched by Taghipour and Ng (2016)). As a result, the shorter responses in SAS may hinder the ability of recurrent networks to achieve state-of-the-art results.

To explore the effectiveness of neural network architectures on SAS, we use the basic architecture and parameters of Taghipour and Ng (2016) on three publicly available short answer datasets: ASAP-SAS (Shermis, 2015), Powergrading (Basu et al., 2013), and SRA (Dzikovska et al., 2016, 2013). While these datasets differ with respect to the length and complexity of student responses, all prompts in the datasets focus on content accuracy. We explore how well the optimal parameters for AES from Taghipour and Ng (2016) fare on these datasets, and whether different architectures and parameters perform better on the SAS task.

2 Datasets

The three datasets we use cover different kinds of prompts and vary considerably in the length of the answers as well as their well-formedness. Table 1 shows basic statistics for each dataset. Figures 1, 2 and 3 show examples for each of the datasets.

2.1 ASAP-SAS

The Automated Student Assessment Prize Short Answer Scoring (ASAP-SAS) dataset[1] contains 10 individual prompts, covering science, biology, and ELA. The prompts were administered to U.S. high school students in several state-level assessments. Each prompt has an average of 2,200 individual responses, typically consisting of one or a few sentences. Responses are scored by two human annotators on a scale from 0 to 2 or 0 to 3 depending on the prompt (Shermis, 2015). Following the guidelines from the Kaggle competition, we always use the score assigned by the first annotator.

[1] https://www.kaggle.com/c/asap-sas

2.2 Powergrading

The Powergrading dataset (Basu et al., 2013) contains 10 individual prompts from U.S. immigration exams with about 700 responses each. Each prompt is accompanied by one or more reference responses. As responses are very short (typically a few words – see Figure 2) and because the percentage of correct responses is very high, responses in the Powergrading dataset are to some extent repetitive. The Powergrading dataset tests models' ability to perform well on extremely short responses.

The Powergrading dataset was originally used for the task of (unsupervised) clustering (Basu et al., 2013), so that there are no state-of-the-art scoring results available for this dataset. For simplicity, we use the first out of three binary human-annotated correctness scores.

2.3 SRA

The SRA dataset (Dzikovska et al., 2012) became widely known as the dataset used in SemEval-2013 Shared Task 7 "The Joint Student Response Analysis and 8th Recognizing Textual Entailment Challenge" (Dzikovska et al., 2013). It consists of two subsets: *Beetle*, with student responses from interacting with a tutorial dialogue system, and *SciEntsBank (SEB)* with science assessment questions. We use two label sets from the shared task: the 2-way labels classify responses as correct or incorrect, while the 5-way labels provide a more fine-grained classification of responses into the categories *non_domain*, *correct*, *partially_correct_incomplete*, *contradictory* and *irrelevant*. In contrast with most SAS datasets, the SRA dataset contains a large number of prompts and with relatively few responses per prompt (see Table 1). Following the procedure from the shared task, we train models for each SRA dataset (Beetle, SEB) across all responses to all prompts.

ASAP - Prompt 1
QUESTION: After reading the groups procedure, describe what additional information you would need in order to replicate the experiment. Make sure to include at least three pieces of information.

SCORING RUBRIC FOR A 3 POINT RESPONSE: The response is an excellent answer to the question. It is correct, complete, and appropriate and contains elaboration, extension, and/or evidence of higher-order thinking and relevant prior knowledge. There is no evidence of misconceptions. Minor errors will not necessarily lower the score.

STUDENT RESPONSES:

- **3 points:** Some additional information you will need are the material. You also need to know the size of the contaneir to measure how the acid rain effected it. You need to know how much vineager is used for each sample. Another thing that would help is to know how big the sample stones are by measureing the best possible way.
- **1 point:** After reading the expirement, I realized that the additional information you need to replicate the expireiment is one, the amant of vinegar you poured in each container, two, label the containers before you start yar expirement and three, write a conclusion to make sure yar results are accurate.
- **0 points:** The student should list what rock is better and what rock is the worse in the procedure.

Figure 1: **ASAP-SAS example:** Question, partial scoring rubric, and example student responses for Prompt 1. (Spelling errors in the student responses are in the original. Source text used in the prompt is omitted here for space reasons.)

PG - PROMPT 1
QUESTION: What is one right or freedom from the First Amendment of the U.S. Constitution?

REFERENCE RESPONSES:

- speech
- religion
- assembly
- press
- petition the government

STUDENT RESPONSES:

- **correct:** freedom of speech
- **correct:** free speech
- **correct:** freedom to talk freely
- **correct:** freedome of religeon
- **incorrect:** the right to bear arms
- **incorrect:** life

Figure 2: **Powergrading example:** Question, reference responses, and example student responses for Prompt 1.

SRA - BEETLE dataset
QUESTION: What are the conditions that are required to make a bulb light up

REFERENCE RESPONSES: the bulb and the battery are in a closed path

STUDENT RESPONSES:

- **correct:** a complete circuit of electricty
- **incorrect:** connection to a battery

Figure 3: **SRA example:** Question, reference response, and example student responses from Beetle subset.

Dataset	# prompts	Scores / Labels	# train responses (mean)	# dev responses (mean)	# test responses (mean)	Mean length (train)
ASAP-SAS	10	0/1/2(/3)	1363	341	522	48.4
PG	10	0/1	418	140	140	3.9
SRA Beetle	47	2 or 5-way	3153 (67.1)	788 (16.8)	449 (9.4)	9.8
SRA SEB	135	2 or 5-way	2968 (29.4)	1001 (7.4)	540 (4.0)	12.5

Table 1: Overview of the datasets used in this work. Since we train prompt-specific models for ASAP-SAS and PG, we report the mean number of responses per set per prompt. For SRA, we train one model per label set across prompts and report the overall number of prompts per set as well as the mean number of responses per prompt per set (in parentheses).

3 Experiments

3.1 Method

We carried out a series of experiments across datasets to discern the effect of specific parameters in the SAS setting. We took the best parameter set from Taghipour and Ng (2016) as our reference since it performed best on the AES data. We looked at the effect of varying several important parameters to discern the effectiveness of each for SAS:

- the role of the mean-over-time layer, which was crucial for good performance in Taghipour and Ng (2016)
- the utility of pretrained embeddings
- the contribution of features derived from a convolutional layer
- the needs for network representational capacity via recurrent hidden layer size
- the role of bidirectional architectures for short response lengths
- regression versus classification
- the effect of attention

To explore the effect of specific parameters, we trained models on the training set and evaluated on the development set only. Following these experiments, we trained a model on the training and development sets and evaluated on the test set. We report prompt-level results for this model in Section 3.6.

For evaluation, we use quadratic weighted kappa (QWK) for the ASAP-SAS and Powergrading datasets. Because the class labels in the SRA dataset are unordered, we report the weighted F1 score, which was the preferred metric in the Semeval shared task (Dzikovska et al., 2016).

3.2 Baseline

As a baseline system, we use a supervised learner based on a hand-crafted feature set. This baseline is based on DkPro TC (Daxenberger et al., 2014) and relies on support vector classification using Weka (Hall et al., 2009). We preprocess the data using the ClearNlp Segmenter [2] via DKPro Core (Eckart de Castilho and Gurevych, 2014). The features used in the baseline system comprise a commonly used and effective feature set for the SAS task. We use both binary word and character uni- to trigram occurrence features, using the top 10,000 most frequent ngrams in the training data, as well as answer length, measured by the number of tokens in a response.

3.3 Neural networks

We work with the basic neural network architecture explored by Taghipour and Ng (2016) (Figure 4).[3] First, the word tokens of each response are converted to embeddings. Optionally, features are extracted from the embeddings by a convolutional network layer. This output forms the input to an LSTM layer. The hidden states of the LSTM are aggregated in either a "mean-over-time" (MoT) layer or attention layer. The MoT layer simply averages the hidden states of the LSTM across the input. We use the same attention mechanism employed in Taghipour and Ng (2016), which involves taking the dot product of each LSTM hidden state and a vector that is trained with the network. The aggregation layer output is a single vector, which is input to a fully connected layer. This layer computes a scalar (regression) or class label (classification).

[2] https://github.com/clir/clearnlp

[3] The networks are implemented in Keras and use the Theano backend.

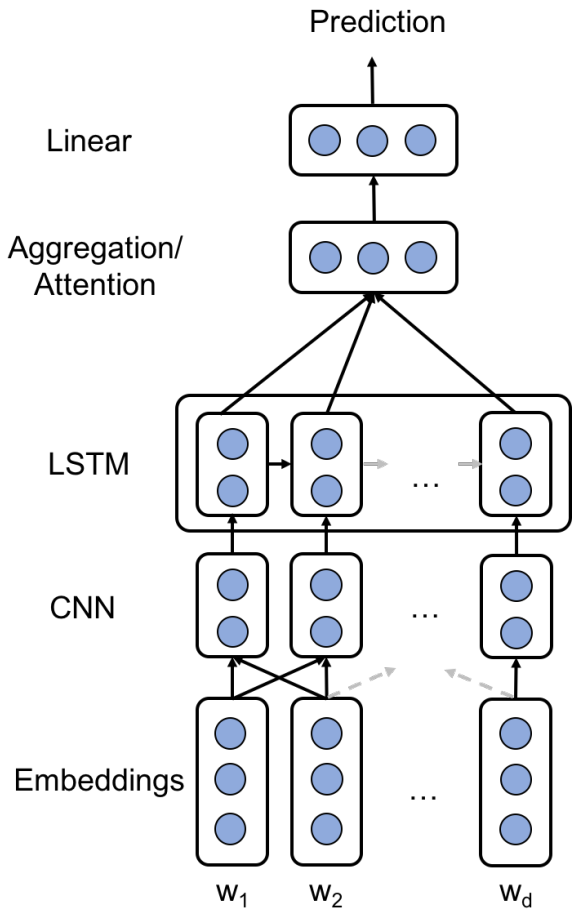

Figure 4: The basic neural architecture explored in this work for short answer scoring.

3.4 Setup, training, and evaluation

The text is lightly preprocessed as input to the neural networks following Taghipour and Ng (2016). The text is tokenized with the standard NLTK tokenizer and lowercased. All numbers are mapped to a single <num> symbol.[4] Each response is padded with a dummy token to uniform length, but these dummy tokens are masked out during model training.

For the ASAP-SAS and Powergrading datasets, prior to training, we scale all scores of responses to [0, 1] and use these scaled scores as input to the networks. For evaluation, the scaled scores are converted back to their original range. The SRA class labels are used as is.

We fix a number of neural network parameters for our experiments. For pretrained embeddings, in preliminary experiments the GloVe 100 dimension vectors (Pennington et al., 2014) performed slightly better than a selection of other off-the-shelf embeddings, and hence we use these for all conditions that involve pretrained embeddings. Embeddings for word tokens that are not found in the embeddings are randomly initialized from a uniform distribution. The convolutional layer uses a window length of 3 or 5 and 50 filters. We use a mean squared error loss for regression models and a cross-entropy loss for classification models. To train the network, we use RMSProp with ρ set to 0.9 and learning rate of 0.001. We clip the norm of the gradient to 10. The fully connected layer's bias is initialized to the mean score for the training data, and the layer is regularized with dropout of 0.5. We use a batch size of 32, which provided a good compromise between performance and runtime in preliminary experiments.

To obtain more consistent results and improve predictive performance, we evaluate the models by keeping an exponential moving average of the model's weights during training. The moving average weights w_{EMA} are updated after each batch by

$$w_{EMA} -= (1.0 - d) * (w_{EMA} - w_{current}).$$

d is a decay rate that is updated dynamically at each batch by taking into account the number of batches so far:

$$min(decay, (1 + \#batches)/(10 + \#batches))$$

where *decay* is a maximum decay rate, which we set to 0.999. This decay rate updating procedure allows the weights to be updated quickly at first while stabilizing across time.

All models are trained for 50 epochs for parameter exploration on the development set (Section 3.5) and 50 epochs for the final models on the test set (Section 3.6). Following Taghipour and Ng (2016), for our parameter exploration experiments on the development set, we report the best performance across epochs. When we train final models on the combined training and development set and evaluate on the test set, we report the results from the last epoch.

During development, we observed that even after employing best practices for ensuring repro-

[4] It may be the case that relevant content information is thus ignored. However, since many numbers occur with units of measurement, e.g. 1g, we do not have word embeddings for them either and so the embeddings would simply be random initializations. We leave a full exploration of this issue to future work.

ducibility of results[5], there was still some small variation between runs of the same parameter settings. The reasons for this variability were not clear.

3.5 Parameter exploration results

Our focus in this section is comparing different architecture and parameter choices for the neural networks with the best parameters from Taghipour and Ng (2016). Table 2 shows the results of our experiments on the *development* set for ASAP-SAS and Powergrading, and Table 3 shows the corresponding results for SRA.

Does the mean-over-time layer improve performance? Taghipour and Ng (2016) demonstrate a large performance gain with the mean-over-time layer that averages the LSTM hidden states across the response tokens. Comparing "T&N best" with "no MoT" across the datasets, we see mixed results. The mean-over-time layer performs relatively well across datasets, but achieves the best results only on the SRA-SEB dataset. We hypothesized that the mean-over-time layer is helpful when the input consists of longer responses (as was the case for the essay data in Taghipour and Ng (2016)). We computed the Pearson's correlation on the ASAP-SAS data between the difference on each prompt of the two conditions and the mean response length in the development set. However, the correlation was modest at 0.437.

Do pretrained embeddings with tuning outperform fixed or randomly initialized embeddings? On all datasets, the pretrained embeddings with tuning (among the "T&N best" parameters) performed better than fixed pretrained or learned embeddings.[6] Tuned embeddings were especially important for the ASAP-SAS and Powergrading datasets.

Does a convolutional layer produce useful features for the SAS task? The results for convolutional features are mixed: convolutional features contribute small performance improvements on Powergrading and one of the SRA label sets (SRA SEB 2-way).

Can smaller hidden layers be used for the SAS task? Although LSTMs with smaller hidden states often outperformed the 300-dimensional LSTM in the T&N best parameter set (compare 'T&N best' performance with performance for 'LSTM dims' conditions), the improvements were all quite small.

Do bidirectional LSTMs improve performance? Bidirectional LSTM architectures produced solid gains over the T&N best parameters on ASAP-SAS, Powergrading, and two of the four SRA label sets.

Can classification improve performance? The T&N model used regression. While the labels in SRA allow only for classification, ASAP-SAS and PG work with both regression and classification. However, we found consistently better results using regression.

Can attention improve performance? The attention mechanism we considered in this paper yielded strong performance improvements over the mean-over-time layer on all datasets except SRA-SEB 5-way. The largest improvements were on Powergrading and SRA-Beetle 5-way, where increases were almost 3 points weighted F1.

We also report the results of the combinations of individual parameters that performed well on the development data at the bottom of Table 2 and Table 3. While these combinations performed better than any individual parameter variation on ASAP-SAS and Powergrading, the combination performed *worse* on three of the four label sets in the SRA data. These results underscore that these parameters do not always produce additive effects in practice.

We examined the predictions from the baseline system and the T&N system for the ASAP-SAS development set and conducted a brief error analysis. In general, across the 10 prompts, it can be observed that when the baseline system is incorrect it tends to under-predict the scores, whereas the T&N system tends to slightly over-predict scores when it is incorrect. These effects are typically small, but consistent.

3.6 Test performance

We selected the top parameter settings on the development set and trained models on the full training set (i.e. training and development sets) for each dataset:

- *ASAP-SAS:* 250-dimensional bidirectional LSTM, attention mechanism

[5]The Numpy random seed was set. Since we used Theano, in run scripts, we used `PYTHONHASHSEED=0`.

[6]We also did experiments with a much larger number of epochs for the "learned" condition, but performance did not approach that of the tuned embeddings.

								ASAP-SAS	Powergrading
Experiment	**Condition**	**Emb**	**CNN**	**Dim**	**Dir**	**MoT**	**Att**	**Mean QWK**	**Mean QWK**
Benchmark	Baseline							0.6529	**0.9049**
	T&N best	tun	no	300	uni	yes	no	0.7381	0.8724
MoT	no MoT	tun	no	300	uni	no	no	0.7197	**0.8753**
Embeddings	fixed	fix	no	300	uni	yes	no	0.7126	0.8376
	learned	lea	no	300	uni	yes	no	0.6687	0.8482
CNN	win len 5	tun	len 5	300	uni	yes	no	0.7224	**0.8748**
Directionality	bi	tun	no	300	bi	yes	no	**0.7396**	**0.8798**
LSTM dims	50	tun	no	50	uni	yes	no	0.7169	0.8514
	100	tun	no	100	uni	yes	no	0.7341	0.8567
	150	tun	no	150	uni	yes	no	0.7377	**0.8797**
	200	tun	no	200	uni	yes	no	0.7343	0.8669
	250	tun	no	250	uni	yes	no	**0.7429**	0.8547
Classification		tun	no	300	uni	yes	no	0.7164	0.8299
Attention	T&N-sum	tun	no	300	uni	no	yes	**0.7436**	**0.9005**
Best combination	ASAP-SAS	tun	no	250	bi	no	yes	**0.7439**	
Best combination	PG	tun	len 5	150	bi	no	yes		**0.9036**

Table 2: Parameter experiment results on ASAP-SAS and Powergrading on the *development* set. "Baseline" is the baseline non-neural system. "T&N best" is the best-performing parameter set in Taghipour and Ng (2016): tuned embeddings (here, GLOVE 100 dimensions), 300-dimensional LSTM, unidirectional, mean-over-time layer. Scores are bolded if they outperform the score for the "T&N best" parameter setting.

- *Powergrading:* CNN features with window length 5, 150-dimensional bidirectional LSTM, attention mechanism
- *SRA:* Because of the decreased performance of the combined best individual parameters on the development data, we use a 300-dimensional *uni*directional LSTM with attention mechanism.

These models are "T&N tuned" in Table 4, which appear along with the non-neural baseline system. On ASAP-SAS, the "T&N tuned" parameter configuration outperformed the baseline system and the "T&N best" parameters. The tuned system does not reach the state-of-the-art Fisher-transformed mean score on the ASAP-SAS dataset (Ramachandran et al., 2015)[7], which, like the winner of the ASAP-SAS competition (Tandalla, 2012), employed prompt-specific regular expressions. Other top performing systems used prompt-specific preprocessing and ensemble-based approaches over rich feature spaces (Higgins et al., 2014).

[7]Ramachandran et al. (2015) state that their mean QWK is 0.0053 higher than the Tandalla result, so in Table 4 we report that score truncated to 3 decimal places rather than the rounded result reported in Ramachandran et al. (2015).

On the Powergrading dataset, the "T&N tuned" system did not match the performance of the baseline system, consistent with the results on the development set (Table 2). It appears that on the very short and redundant data in this dataset, the character- and n-gram based system can learn somewhat more efficiently than the neural systems.

On the SRA datasets, the "T&N tuned" model outperformed the baseline and the "T&N best" settings on average across prompts, by a larger margin than the other datasets. On the SRA data, as on the ASAP-SAS data, a gap remains between the tuned model's performance and the state of the art. On SRA, this may be partly due to the use of "question indicator" features by the top performing systems (Heilman and Madnani, 2013; Ott et al., 2013).

The performance improvement over the baseline system was larger on the development sets than on the test sets. Part of the reason for this is that the test set evaluation procedure likely did not choose the best-performing epoch for the neural models.

								SRA Beetle 2-way	SRA Beetle 5-way	SRA SEB 2-way	SRA SEB 5-way
Experiment	**Condition**	**Emb**	**CNN**	**Dim**	**Dir**	**MoT**	**Att**	**Mean wF1**	**Mean wF1**	**Mean wF1**	**Mean wF1**
Benchmark	Baseline							0.7438	0.5815	0.7011	0.5415
	T&N best	tun	no	300	uni	yes	no	0.7805	0.6184	0.7386	0.6175
MoT	no MoT	tun	no	300	uni	no	no	0.7803	0.6163	0.7384	0.6159
Embeddings	fixed	fix	no	300	uni	yes	no	0.7803	0.6119	0.7112	0.5730
	learned	lea	no	300	uni	yes	no	0.7396	0.5929	0.7285	0.5855
CNN	win len 3	tun	no	300	uni	yes	no	0.7786	0.6048	**0.7431**	0.5874
Directionality	bi	tun	no	300	bi	yes	no	0.7699	**0.6461**	**0.7511**	0.6171
LSTM dims	50	tun	no	50	uni	yes	no	0.7603	0.5954	**0.7395**	0.5992
	100	tun	no	100	uni	yes	no	0.7679	**0.6192**	0.7341	0.5925
	150	tun	no	150	uni	yes	no	**0.7816**	0.6168	**0.7389**	0.6039
	200	tun	no	200	uni	yes	no	0.7768	**0.6186**	0.7336	0.6080
	250	tun	no	250	uni	yes	no	0.7663	0.6106	0.7334	0.6160
Attention	T&N-sum	tun	no	300	uni	no	yes	**0.7915**	**0.6469**	**0.7454**	0.5941
Combination		tun				no	yes	0.7691	**0.6246**	0.7308	0.6109

Table 3: Parameter experiment results on SRA datasets on the *development* set. "wF1" is the weighted F1 score. "Baseline" is the baseline non-neural system. "T&N best" is the best-performing parameter set in Taghipour & Ng (2016): tuned embeddings (here, GLOVE 100 dimensions), 300-dimensional LSTM, unidirectional, mean-over-time layer. Scores are bolded if they outperform the score for the "T&N best" parameter setting.

4 Discussion

Our results establish that the basic neural architecture of pretrained embeddings with tuning across model training and LSTMs is a reasonably effective architecture for the short answer content scoring task. The architecture performs well enough to exceed a non-neural content scoring baseline system in most cases.

Given the diversity of prompts in SAS, there was a good deal of variation in the effectiveness of parameter choices in this neural architecture. Still, some basic trends emerged. First, pretrained embeddings tuned across model training were crucial for competitive performance on most datasets. Second, neural models for SAS generally benefit from similar size hidden dimensions as models for AES. Only the Powergrading dataset, with very short answers and a small vocabulary for each prompt, benefitted from a significantly smaller LSTM dimensionality. The relationship between task, rubrics, vocabulary size, and the representational capacity of neural models for SAS need further exploration.

Third, a mean-over-time aggregation mechanism on top of the LSTM generally performed well, but notably this mechanism was not nearly as important as in the AES task. Mean-over-time produced competitive results on many prompts, but contrary to Taghipour and Ng (2016), *bidirectional* LSTMS and *attention* produced some of the best results, which is consistent with results for neural models on other text classification tasks (e.g., Longpre et al. (2016)).

Research is needed to explain these emerging differences in effective neural architectures for AES vs. SAS, including model-specific factors such as the interaction of an LSTM's integration of features over time and the redundancy of predictive signals in essays vs. short answers, along with data-specific factors such as the consistency of human scoring, the demands of different rubrics, and the homogeneity or diversity of prompts in each setting. At the same time, different from the AES task, the family of neural architectures explored here needs further augmenting to achieve state-of-the-art results on the SAS task. Moreover, more experiments are needed to document how well neural systems perform relative to highly optimized non-neural systems. While further parameter optimizations and different architectures may yield better results, it may be the case that the

Dataset	Prompt	Baseline	T&N best	T&N tuned	State of the art
ASAP-SAS	1	0.719	0.784	0.**795**	
	2	0.719	**0.742**	0.718	
	3	0.592	**0.702**	0.684	
	4	0.688	0.697	**0.700**	
	5	0.752	0.821	**0.830**	
	6	0.775	0.774	**0.790**	
	7	0.606	0.638	**0.648**	
	8	**0.571**	0.566	0.554	
	9	0.760	**0.791**	0.777	
	10	0.691	0.681	**0.735**	
	Mean	0.687	0.720	**0.723**	
	$\text{Mean}_{\text{Fisher}}$	0.693	0.728	0.732	**0.776**
PG	1	**1.000**	**1.000**	**1.000**	-
	2	0.866	**0.897**	0.844	-
	3	**0.743**	0.597	0.614	-
	4	**0.926**	0.903	0.887	-
	5	**0.930**	0.759	0.759	-
	6	**0.930**	0.880	0.906	-
	7	0.831	0.831	**0.881**	-
	8	0.985	0.970	**1.000**	-
	13	**0.576**	0.553	0.554	-
	20	**0.949**	**0.949**	**0.949**	-
	Mean	**0.873**	0.834	0.839	-
SRA	Beetle 2-way	0.742	0.776	0.790	**0.845**
	Beetle 5-way	0.583	0.630	0.633	**0.715**
	SEB 2-way	0.661	0.670	0.712	**0.773**
	SEB 5-way	0.503	0.521	0.533	**0.643**
	Mean	0.622	0.649	0.667	**0.744**

Table 4: *Test* set results for all datasets across prompts. Scores for ASAP-SAS and PG are QWK. $\text{Mean}_{\text{Fisher}}$ is the Fisher-transformed mean QWK used in the ASAP-SAS competition. Scores for SRA are weighted F1 scores.

SAS task of content scoring with relatively short response sequences requires neural approaches to employ a larger set of features (Pado, 2016) or a greater level of prompt-specific tuning, or pairing with methods from active learning (Horbach and Palmer, 2016).

Acknowledgements

We thank Nitin Madnani, Swapna Somasundaran, Beata Beigman Klebanov and the anonymous reviewers for their detailed comments. Part of this work was funded by the German Federal Ministry of Education and Research under grant no. FKZ 01PL16075.

References

Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. 2016. Automatic text scoring using neural networks. In *Proceedings of the 54th Annual Meeting of the Association of Computational Linguistics*.

Sumit Basu, Chuck Jacobs, and Lucy Vanderwende. 2013. Powergrading: a Clustering Approach to Amplify Human Effort for Short Answer Grading. *Transactions of the Association for Computational Linguistics (TACL)* 1:391–402.

Steven Burrows, Iryna Gurevych, and Benno Stein. 2015. The eras and trends of automatic short answer grading. *International Journal of Artificial Intelligence in Education* 25(1):60–117.

Jill Burstein, Joel Tetreault, and Nitin Madnani. 2013. The e-rater automated essay scoring system. *Handbook of automated essay evaluation: Current applications and new directions* pages 55–67.

Johannes Daxenberger, Oliver Ferschke, Iryna Gurevych, and Torsten Zesch. 2014. Dkpro TC: A java-based framework for supervised learning experiments on textual data. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*.

Fei Dong and Yue Zhang. 2016. Automatic features for essay scoring - an empirical study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Myroslava O. Dzikovska, Rodney Nielsen, Chris Brew, Claudia Leacock, Danilo Giampiccolo, Luisa Bentivogli, Peter Clark, Ido Dagan, and Hoa Trang Dang. 2013. SemEval-2013 Task 7: The Joint Student Response Analysis and 8th Recognizing Textual Entailment Challenge. **SEM 2013: The First Joint Conference on Lexical and Computational Semantics* .

Myroslava O Dzikovska, Rodney D Nielsen, and Chris Brew. 2012. Towards effective tutorial feedback for explanation questions: A dataset and baselines. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Myroslava O Dzikovska, Rodney D Nielsen, and Claudia Leacock. 2016. The joint student response analysis and recognizing textual entailment challenge: making sense of student responses in educational applications. *Language Resources and Evaluation* 50(1):67–93.

Richard Eckart de Castilho and Iryna Gurevych. 2014. A broad-coverage collection of portable nlp components for building shareable analysis pipelines. In *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT*. Association for Computational Linguistics and Dublin City University, Dublin, Ireland, pages 1–11.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. *SIGKDD Explorer Newsletter* 11(1):10–18.

Michael Heilman and Nitin Madnani. 2013. ETS: Domain adaptation and stacking for short answer scoring. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval)*. pages 275–279.

Derrick Higgins, Chris Brew, Michael Heilman, Ramon Ziai, Lei Chen, Aoife Cahill, Michael Flor, Nitin Madnani, Joel R Tetreault, Daniel Blanchard, Diane Napolitano, Chong Min Lee, and John Blackmore. 2014. Is getting the right answer just about choosing the right words? The role of syntactically-informed features in short answer scoring http://arxiv.org/abs/1403.0801.

Andrea Horbach and Alexis Palmer. 2016. Investigating active learning for short-answer scoring. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*.

Shayne Longpre, Sabeek Pradhan, Caiming Xiong, and Richard Socher. 2016. A way out of the odyssey: Analyzing and combining recent insights for lstms. *arXiv preprint arXiv:1611.05104* .

Niels Ott, Ramon Ziai, Michael Hahn, and Walt Detmar Meurers. 2013. CoMeT: Integrating different levels of linguistic modeling for meaning assessment. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh InternationalWorkshop on Semantic Evaluation (SemEval 2013)*. pages 608–616.

Ulrike Pado. 2016. Get semantic with me! the usefulness of different feature types for short-answer grading. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING)*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

Lakshmi Ramachandran, Jian Cheng, and Peter W Foltz. 2015. Identifying patterns for short answer scoring using graph-based lexico-semantic text matching. In *Proceedings of the 10th Workshop on Innovative Use of NLP for Building Educational Applications*.

Mark D Shermis. 2014. State-of-the-art automated essay scoring: Competition, results, and future directions from a united states demonstration. *Assessing Writing* 20:53–76.

Mark D Shermis. 2015. Contrasting state-of-the-art in the machine scoring of short-form constructed responses. *Educational Assessment* 20(1):46–65.

Mark D. Shermis and Ben Hamner. 2013. Contrasting state-of-the-art automated scoring of essays. In *Handbook of Automated Essay Evaluation*, Taylor and Francis, New York.

Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Luis Tandalla. 2012. Scoring short answer essays.

Siyuan Zhao, Yaqiong Zhang, Xiaolu Xiong, Anthony Botelho, and Neil Heffernan. 2017. A memory-augmented neural model for automated grading. In *Proceedings of the Fourth ACM Conference on Learning @ Scale*.