Automated Essay Scoring with Attention and BERT

Kuangwei Huang and Martin Jung

School of Information, University of California, Berkeley
{kuangwei,martin.jung}@berkeley.edu

April 12, 2019

Abstract

Essays are an important form of communication involving the organization, synthesis and delivery of relevant information and ideas with regards to a domain-specific topic. Essays serve an important purpose in education to assess learning outcomes of students, but grading of essays is time consuming, subjective, leading to variation in assessment amongst human graders. Automated essay scoring (AES) has been an active area of research in recent years, and the applications of research in AES is believed to be extensible beyond education to help develop more effective search engines and question answering systems to provide better access to digital written content online. With recent advances in machine learning and neural networks in natural language processing (NLP), from dense vector representations of word embeddings[1] to the algorithms pertaining to attention models and deep bi-directional encoder representations for transformers (BERT)[2], the state-of-the-art has achieved higher levels of machine abstraction needed for complicated problems such as AES.

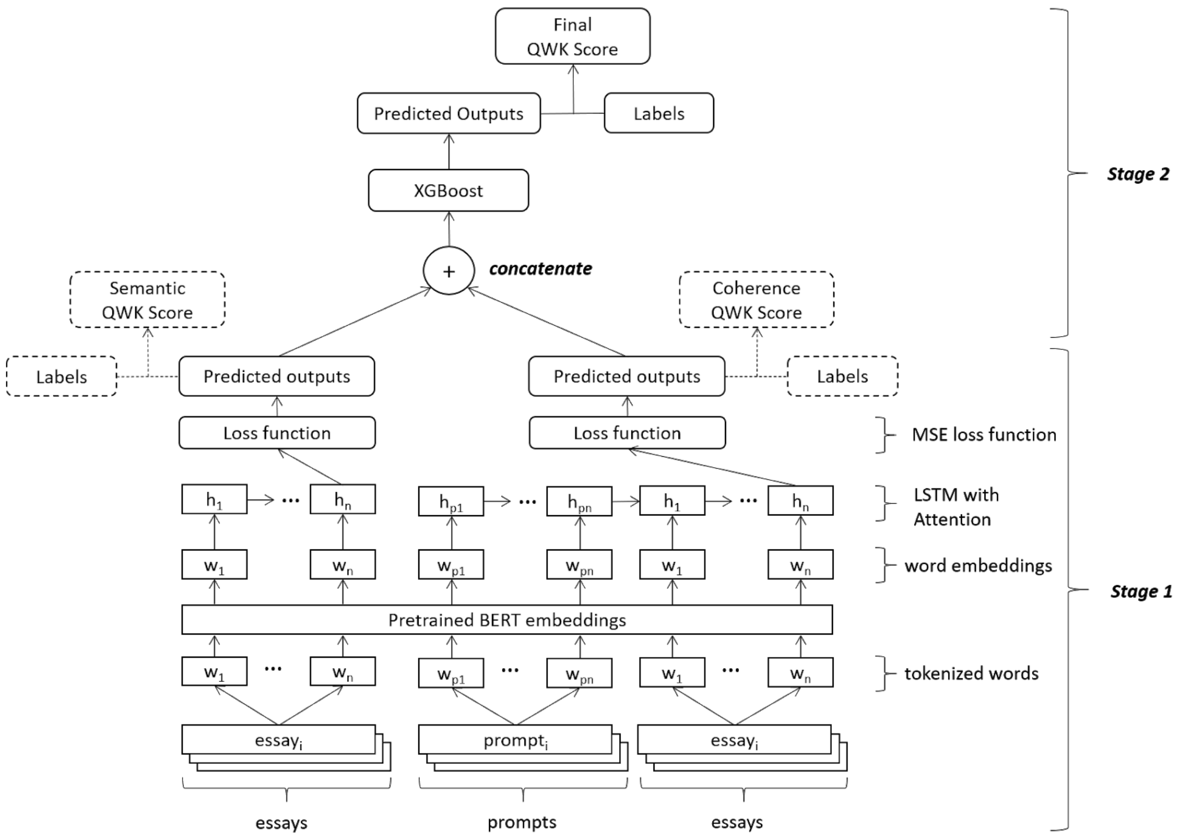

Figure 1: Model for AES TSLF

Our research project seeks to experiment using pre-trained BERT embeddings coupled with a Two-Stage Learning Framework (TSLF) combining Long Short-Term Memory (LSTM) with Attention for Semantic Score and Coherence Score at the first stage and Extreme Gradient Boosting (XGBoost) for the final score at the second stage (Figure 1).

Introduction

The recent developments in recurrent neural networks[3], attention models and deep bi-directional encoder representations for transformers have attributed to increased interest in using these methods to have machine learning models learn features relating to semantics and coherence and develop a high generalization power for evaluation of written information. This lends itself to the complicated task of discerning good writing from bad with respect to a given context or prompt. In this paper, we report a system taking advantage of these advances and apply them to automated essay scoring.

Methods

Data

We intend to implement our model and algorithms onto the Automated Student Assessment Prize (ASAP) AES dataset (https://www.kaggle.com/c/asap-aes/data), which contains essays written by students ranging from Grade 7 to Grade 10. The dataset consists of 8 essay sets, each with a different topic or prompt, with a total of 12,978 essays with scores. The dataset will be split into train (9732 essays), development (1947) and test (1298), for scoring different models in our experiments. A summary of the essay sets is provided in Table 1.

Essay Set	*Type of Essay*	*Grade Level*	*No. of Samples*
1	persuasive / narrative / expository	8	1783
2	persuasive / narrative / expository	10	1800
3	source dependent responses	10	1726
4	source dependent responses	10	1772
5	source dependent responses	8	1805
6	source dependent responses	10	1800
7	persuasive / narrative / expository	7	1569
8	persuasive / narrative / expository	10	723
		Total	***12978***

Table 1: Summary of essay sets and training set sizes in the ASAP AES dataset

Each of the sets of essays was generated from a single prompt. Some of the essays are dependent upon source information and others are not. All responses were written by students ranging in grade levels from Grade 7 to Grade 10. All essays were hand graded and were double-scored. Each of the eight data sets has its own unique characteristics. Most of the essays have less than 200 words (Figure 2) and around 500 to 600 words cover more than 90 percent of the essays. For prompts, certain essay sets have long prompts starting with excerpts from reference texts attributing to long lengths of 800 to 1600 words (Figure 3).

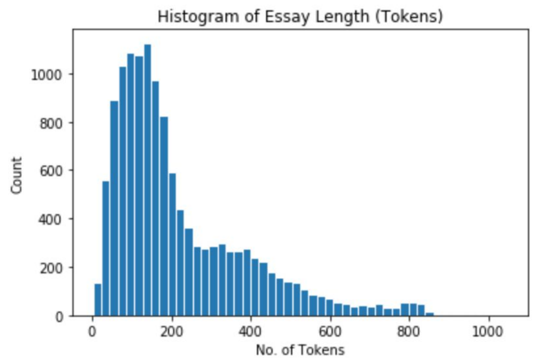

Figure 2: Histogram of Essay Length

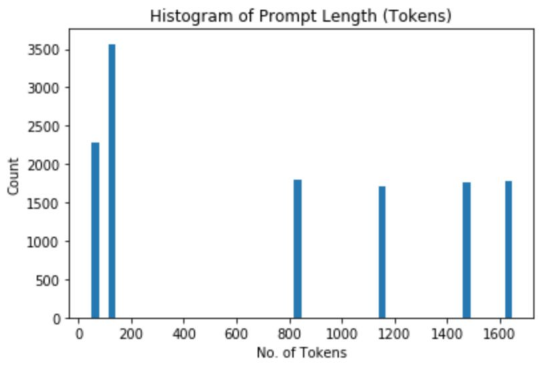

Figure 3: Histogram of Prompt Length

Evaluation Metric

Scoring for AES is compared against the human annotated labels for each essay, and the Quadratic Weighted Kappa (QWK) had been adopted by the ASAP competition as the official evaluation metric. A number of studies train and evaluate on QWK for the un-normalised scores which differ for different essay sets. We believe this would cause a model to narrowly learn features specific to the essay set scoring and would have less generalizability or opportunity for transfer learning past the existing essay sets. Hence, we chose to normalize all essay scores to a float between [0,1] to have a consistently measure of how good an essay is scored without having any relation to how the essay set scoring rules are set up. All models would be trained to minimize the mean squared error between predictions and labels, and the resultant QWK score would be converted from the model's essay score predictions into an integer score ranging between [0,100] and compared to the normalized essay score labels converted to the same [0-100] integer scale. Without extensive feature engineering and relying on neural networks, we plan to first develop a simple baseline model and then seek to achieve improved results with our proposed modeling solution. We would also compare this advanced neural network approach against the winning model (QWK of 0.81407) in the ASAP contest back in 2012 which contained more feature engineering, and other reference studies that use the same normalized score approach.

Modeling

We took basis from a paper which introduced TSLF [4] and simplified the model by removing seemingly redundant coherence score and feature engineering as in Figure 1. A Neural Bag of Words is used as a baseline to compare our model's performance. For our model, we use BERT embeddings and 2 LSTM models, one to predict semantic score and the other to predict coherence score which we compared across essays with the prompt to see if the model can learn features about the relationship between essay and prompt. We finally used two predicted scores to produce the final score by using boosted decision trees.

Neural Bag of Words Baseline Modeling

As shown in Figure 4, Neural Bag of Words (NBOW) takes its name from the bag-of-words assumption common to linear models, in which the weights for each input word are summed to make a prediction. Each essay word is used as an input to predict the final score of the essay.

For our baseline NBOW model, we tokenized the essays and prompts either padded or truncated the list of tokens based on a decided maximum essay and prompt length. The maximum length of the essay was set at 650, as more than 95% of the essay data was shorter than this length. We chose to set the maximum prompt length at 130, as all the longer prompts which are at more than 800 words have very long reference passages before reaching the actual essay question at the end of the prompt. We also chose to truncate the essays and prompts differently, as we expect the most relevant content in an essay located up front, whereas the most important content in a prompt is at the end with the essay question. Hence, essays which exceed the maximum essay length are truncated from the end, and prompts which exceed the maximum prompt lengths are truncated at the beginning.

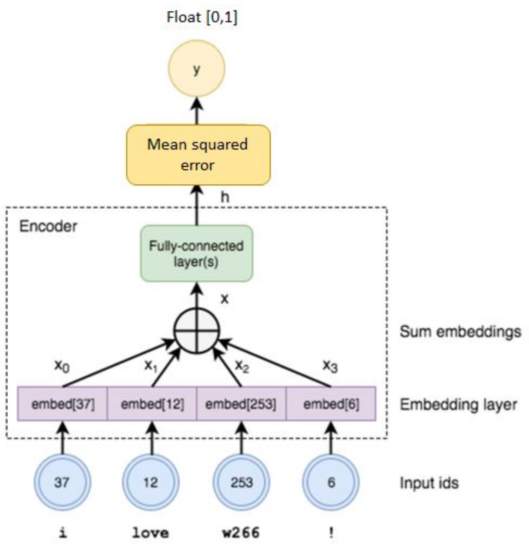

Figure 4: Neural Bag of Words

We further built the vocabulary for the baseline NBOW from all the unique words in the essays and prompts, included in the tokens for padding and unknown words, and pruned the vocabulary for words that appeared less than 3 times in the corpus.

Although the vocabulary was built with both essay and prompt, the NBOW model was feed with only the essays converted into tokenized word IDs.

Dual LSTM Attention with BERT Modeling

BERT Embeddings

We chose the latest state-of-the-art in terms of word embeddings, and utilized pre-trained BERT embeddings which is open sourced by Google. BERT makes use of an attention mechanism that learns contextual relations between words in a text. Since BERT's goal is to generate a language model, an encoder that reads the text input creates the model. Compared to directional models, which read the text input sequentially (left-to-right or right-to-left), the BERT reads the entire sequence of words at once, classified as bidirectional or non-directional method. This characteristic allows the model to learn the context of a word based on all of its surroundings instead of only on words behind or after.

We used package Kashgari[8] to establish the word embedding of essays. We utilized the uncased base-BERT embeddings with 768 dimensions. The full BERT embedding of the essays and prompts across all datasets took 7:17:56.

Stage 1: Dual LSTM with Attention

We choose to use Attention to achieve better result with LSTM model. Without Attention, translation relies on reading a complete sentence and compress all information into a fixed-length vector, an essay with a sentence with many words represented by several words will surely lead to information loss, inadequate translation, etc.

With Attention model (Figure 5), the model's output now depends on a weighted combination of all the input states, not just the last state. The weights define in how much of each input state should be considered for each output. So, if h2's weight is a large number, this would mean that the model pays a lot of attention to the second state in the source sentence. The a's are typically normalized to sum to 1 (so they are a distribution over the input states). We used Attention model to predict both Semantic Score and Cohenance Score.

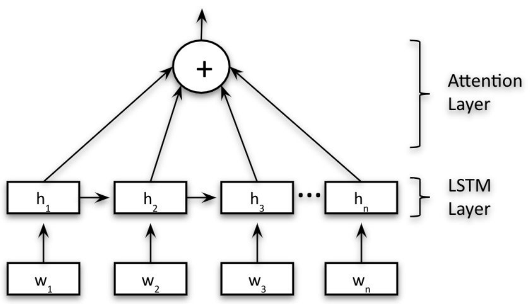

Figure 5: LSTM with Attention

Semantic model only uses BERT embedding of Essay and Coherence model uses embedding of both Prompt and Essay as below.

- h_m for last hidden state of LSTM
- h_{m+n} for last hidden state of LSTM with Prompt

For both models, we used object function to minimize Mean Squared Error as below.

Semantic Score (S_e)

- $S_e = sigmod(w_s h_m + b_s)$
- w_s for weighted matrix of the dense layer
- b_s stands for the bias
- $obj(S_e, \bar{S}_e) = 1/N \sum_{i=1}^{n} (S_i - \bar{S}_i)^2$
- S_e for predict score set of training samples
- $\bar{S}_e$ for the original hand marked score set

Coherence Score (C_e)

- $C_e = sigmod(w_c h_{m+n} + b_c)$
- w_c for weighted matrix of the dense layer
- b_c stands for the bias
- $obj(C_e, \bar{C}_e) = 1/N \sum_{i=1}^{n} (C_i - \bar{C}_i)^2$
- C_e for predict coherence score set of training samples
- $\bar{C}_e$ for the gold coherence score (hand marked scores)

Stage 2: XGBoost

We chose XGBoost for its performance and speed. In boosting, the trees are built sequentially such that each subsequent tree aims to reduce the errors of the previous tree. Each tree learns from its predecessors and updates the residual errors. Hence, the tree that grows next in the sequence will learn from an updated version of the residuals. Hence, by concatenating Semantic Score and Coherence Score as below, we expected to have better result than each predicted score.

- O_e: Estimated Output
- $O_e = XGBoost([S_e, C_e])$

[ ] means concatenating operation.

Initial Results

Baseline

The results of the baseline NBOW show that Quadratic Weighted Kappa (QWK) score for dev and test are 0.7111 and 0.7265 respectively. (Table 2)

Baseline Model		
Set	MSE	QWK
dev	3.83 %	0.7111
test	3.53 %	0.7265

Table 2: MSE and QWK result of baseline NBOW.

Stage 1: Dual LSTM with Attention

Compared to baseline QWK score, the results of the LSTM with Attention model for Semantic score shows 14.8% increase for dev set and 13.4% increase for test set with 50 hidden units and 20 epochs. (Table 3)

The LSTM with Attention model for Coherence score also shows 15.0% increase for dev set with 50 hidden unit and 20 epochs and 13.0% increase for test set with 64 hidden unit and 50 epochs when compared to the baseline.

Semantic LSTM Model				
Run	Hidden Unit	epochs	Set	QWK
1	10	5	dev	0.7427
			test	0.7674
2	50	20	dev	0.8169
			test	0.8245
3	64	50	dev	0.8111
			test	0.8033
Coherence LSTM Model				
Run	Hidden Unit	epochs	Set	QWK
1	10	5	dev	0.7249
			test	0.7472
2	50	20	dev	0.8183
			test	0.818
3	64	50	dev	0.8058
			test	0.8209

Table 3: QWK results of Semantic Model and Coherence Model.

Stage 2: XGBoost

Combining predicted scores for both Semantic model and Coherence model, XGBoost result shows 15.0% increase for dev set and 13.8% increase for test set with 50 hidden unit and 20 epochs. (Table 4)

Combined Models with XGB (pre-tuning)				
Run	Hidden Unit	epochs	Set	QWK
1	10	5	dev	0.7319
			test	0.7844
2	50	20	dev	0.8182
			test	0.8271
3	64	50	dev	0.8121
			test	0.8120

Table 4: QWK results for pre-tuned XGBoost, taking input from both LSTMs Semantic and Coherence Model.

Analyses and Further Tuning

The best performing Semantic LSTM was Run 2, and this was done with at 20 training epochs. In analyzing the tensorboard epoch MSE loss plots, we found that the curve's negative gradient was not as flat as that for Run 3, which had 50 training epochs. We re-ran Run 2 with a 50 training epochs, but other than a closer fit to the train data, the results on the dev and test sets did not prove to be exceed the original Run 2 at 20 training epochs. The higher epochs may just be causing overfitting to the train data without any benefit to the model's predictive ability. (Table 5)

Semantic LSTM Model				
Run	**Hidden Unit**	**epochs**	**Set**	**QWK**
2	50	20	train	0.9073
			dev	0.8169
			test	0.8245
2a	50	50	train	0.9615
			dev	0.8198
			test	0.8212

Table 5: QWK result comparison of Semantic Model Run 2 with increased no. of training epochs.

We further took a look at different activation functions for the dense layer of the LSTMs. The models are setup with sigmoid activation, and we experimented with a rectified linear function with no improvement in QWK (Appendix Table A1).

Finally we performed a GridSearch cross-fold validation for the hyperparameters for the XGBoost Regressor, and managed to improve results with the new hyperparameters. A full listing of all the re-run results are provided in the Appendix. (Appendix Table A1)

A plot of the final scores against the variation of hidden units and training epochs show that Run 2 is still the model that produces the best results on the dev set, and this is the final selected model for the results of this study. (Figure 6)

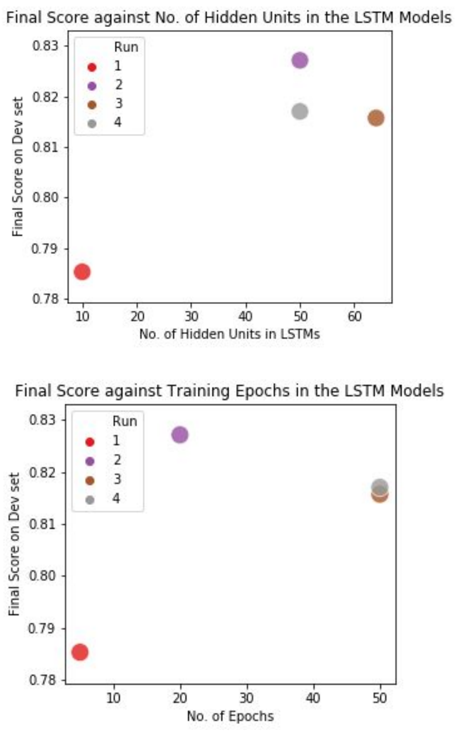

Figure 6: Plot of post-tuning 2nd stage XGB scores on the dev set data showing the variation scores against hidden units and epochs.

The final selected model results when compared to the baseline show a 14.9% improvement and a 2.5% improvement over the ASAP competition winner. However it is to be noted that the ASAP competition is tested against an un-released test set, hence this prevent direct comparisons from being made. (Table 6)

Final Result Comparison		
Model	**Set**	**QWK**
Neural BOW Baseline	dev	0.7111
	test	0.7265
Dual LSTM w Attention, BERT (Run 2, post-tuning)	dev	0.8271
	test	0.8346
Improvement		
over baseline	test	**14.9%**
over ASAP competition winner	**-***	**2.5%***
over Taghipour and Ng 2016.	**-***	**10.7%***

** ASAP competition score is based on an un-released test set, and Taghipour and Ng 2016 [6] is a research paper also utilizing normalised scores across essay sets with a different test set partition from this study.*

Table 6: Final best set of post-tuning results for the Dual LSTM w Attention Model (Run 2) compared with the NBOW baseline and other references.

When searching for other research papers for similar approaches in normalizing the scores across essay sets, we found a that another paper (Taghipour and Ng 2016) was only able to achieve a QWK score of 0.754 using a variety of neural network approaches including LSTM[6]. Our methods improve on this by 10.7%.

Conclusion

The results clearly show that our best model provided a test score of 0.8346 on the Quadratic Weighted Kappa metric, this is a 14.9% improvement over the baseline test score of 0.7265, and also outperforms other previous LSTM methods. This verifies the high potential of recurrent neural network architectures with attention in being able to achieve higher accuracy for automated essay scoring applications. Although our model is only a 2.5% improvement over the winning model for the ASAP competition, considering that we used a different evaluation criteria with score normalization across essay sets as well as not needing to rely on feature engineering, our solution provides better generalization power to this natural language processing problem. When comparing to a Taghipour and Ng 2016 which also used normalized scores across essay set, we managed to exceed those those results by 10.7%.

LSTMs with attention represent the state-of-the-art methods to natural language processing. We believe our approach with dual LSTM has proven to be effective in the the task of automated essay scoring. A wider grid search of hyperparameters for the LSTMs could possibly provide further model optimization and improved results, however, due to time constraints we were unable to set up a larger hyperparameter space to further uncover more optimal parameters.

Future Research Opportunities

Overall, the objectives of this research has been achieved, but there exist future research opportunities which we can further improve this work. We would like to search a more extensive hyperparameter on the deep LSTM models upon getting access to more computing resources and time. We could also test how well the model stands against adversarial inputs [5]. In the course of this work, we identified a possibility to add on an additional two-headed creativity module to the model to account for essays which are have larger distance metric from the overall average essays when corrected for the essay prompt, and believe this could be an interesting addition to the model in the future.

Acknowledgements

We would also like to thank Daniel Cer and the rest of the instructors and teaching assistants for the Natural Language Processing with Deep Learning class at University of California, Berkeley.

References

Dataset:

ASAP AES dataset https://www.kaggle.com/c/asap-aes/data

Papers:

1. Mikolov, T., Corrado, G., Chen, K., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. Proceedings of the International Conference on Learning Representations (ICLR 2013), 1–12. https://arxiv.org/abs/1301.3781

2. Devlin, J., Chang, M., Lee, K., Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. https://arxiv.org/abs/1810.04805

3. Valenti, S., Neri, F., Cucchiarelli, A. (2003). An Overview of Current Research on Automated Essay Grading. Journal of Information Technology Education: Research, Volume 2, pp. 319-330. https://doi.org/10.28945/331

4. Liu, J., Xu, Y., Zhao, L. (2019). Automated Essay Scoring based on Two-Stage Learning. https://arxiv.org/abs/1901.07744

5. Farag, Y., Yannakoudakis, H., Briscoe, T. (2018). Neural Automated Essay Scoring and Coherence Modeling for Adversarially Crafted Input. NAACL 2018: Human Language Technologies, Volume 1. https://arxiv.org/abs/1804.06898

6. Taghipour K., Ng H.T., (2016). Neural Approach to Automated Essay Scoring. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 1882–1891,Austin, Texas, November 1-5, 2016 https://aclweb.org/anthology/D16-1193

Others:

7. https://github.com/BrikerMan/Kashgari

GitHub repository for this paper:

https://github.com/kuangweihuang/MIDS_w266_final_project_essay_scoring

Appendix

Run	***Scores***			***LSTM Model Parameters***				***Data Set***
	Semantic	***Coherence***	***Final***	***hidden***	***batch***	***epochs***	***activation***	
1	0.7573	0.7411	0.8193	10	32	5	sigmoid	train
	0.7427	0.7249	0.7853	10	32	5	sigmoid	dev
	0.7674	0.7472	0.8127	10	32	5	sigmoid	test
2	0.9073	0.8986	0.9307	50	32	20	sigmoid	train
	0.8169	0.8183	0.8271	50	32	20	sigmoid	dev
	0.8245	0.818	0.8346	50	32	20	sigmoid	test
3	0.9682	0.9699	0.9823	64	32	50	sigmoid	train
	0.8111	0.8058	0.8157	64	32	50	sigmoid	dev
	0.8033	0.8209	0.8229	64	32	50	sigmoid	test
4	0.9245	0.9446	0.9678	50	32	50	relu	train
	0.7864	0.809	0.817	50	32	50	relu	dev
	0.7911	0.8193	0.8297	50	32	50	relu	test

Table A1: Full set of final post tuning results for XGBoost for both LSTMs Semantic and Coherence Model.