Grammatical Error Correction using Neural Networks

Yokila Arora
ICME
Stanford University
yarora@stanford.edu

Jaspreet Kaur
Department of Electrical Engineering
Stanford University
j031723@stanford.edu

Ayush Gupta
Department of Electrical Engineering
Stanford University
ayushg@stanford.edu

Abstract

The growing number of English Language Learners has placed increased focus on developing automated resources to have a wider teaching impact. Prominent among those are the research efforts being undertaken in the field of automated Grammatical Error Correction. While rule-based approaches and machine translation models like phrase-based statistical machine translation models have so far been the popular choice in the field, there is significant interest to develop neural network models that can address the limitations posed by these models and improve results. Inspired by the progress achieved in this direction, this work presents an improved architecture of A Neural Network Global Lexicon Model which when added to a phrase-based Statistical Machine Translation model (SMT) improves the SMT baseline $F_{0.5}$ score by 0.57.

1. Introduction

The current era of global connectivity ushered majorly by the growth of internet has made learning English indispensably important. The number of people learning English as a second language has seen a huge growth over the past decade putting greater demand for better teaching resources and facilities, especially for grammatical error correction. However, the present day scenario of providing manual writing feedback is tedious and limited which has led to increased interest in generating tools that can automate Grammatical Error Correction (GEC).

Though statistically significant improvements have been achieved by employing phrase-based Statistical Machine Translation (SMT) models for GEC, their use of discrete word representation, linear translation mapping and inability to capture a global context regresses results. To improve upon this, recent works have suggested using neural network models that can compensate for features lacking in phrase-based SMT systems. Taking motivation from those, this work adds a Neural Network Global Lexicon Model (NNGLM) to a GEC system based on phrase-based SMT approach. The NNGLM calculates individual word probabilities and uses them to score the correction predictions made by the phrase-based SMT model in response to the input sentences with incorrect grammar. The architecture specifications for the NNGLM have been decided after conducting experiments to study the impact of various network configurations on system performance. Based on those, a Feed-Forward Neural Network (FFNN) with three hidden layers has been used as the NNGLM model. Results show that the proposed model leads to significant improvements over the phrase-based SMT baseline.

The report is organized as follows. After conducting a brief review of the recent works associated with GEC in Section 2, we discuss the proposed model in Section 3. Further, the specifications about the dataset with the implementation and experimentation details are discussed in Section 4. Section 5 details the experimentation results and the following discussion along with the scope for future work are presented in Section 6. Finally, conclusions are drawn in Section 7.

2. Related Work

Research to automate GEC faces a difficult challenge because of factors like non-availability of an absolute desired result, dependence on context, existence of non-quantifiable rules, etc. Despite these, the potential benefits of success with GEC invites significant research interest. Several shared tasks like Helping Our Own (HOO) and CONLL [NWB$^+$14] have also added to this interest by bringing together a community of enthusiastic researchers.

As mentioned before, SMT systems proved to be a useful approach for achieving credible GEC application. Brockett et. al. [BDG06] was one of the early works to use SMT systems for GEC which established the potential of this technique in providing writing assistance to second language learners. Using examples of mass noun errors, they showed that SMT systems demonstrate a natural mechanism to suggest error correction. Several other works developed the use of SMT system, some even using them in combination with other approaches for superior results. Felice et. al. [FYA+14] presented a hybrid model using rule-based systems along with a SMT error correction model. Efficacy of the model at correcting mechanical errors was reported, though with instances of unnecessary deletions. Additionally, to determine the error types, Bryant et al. [BFB17] developed an automatic annotation toolkit for grammatical error correction, using a dataset-agnostic, rule-based framework. They used approximately 50 rules to discriminate between various error types and achieved 95% accuracy according to evaluation by human experts.

The use of neural networks for GEC have shown to be advantageous in the recent works starting from Bengio et. al. [BDVJ03]. A further recent work by Ha et al. [HNW15] proposes a discriminative lexicon model using a deep neural network architecture to exploit wider contexts as compared to phrase based SMT systems. The neural network acts as a multivariate classifier, and outputs probabilities for each target word given the source sentence. Further, they extend the model by using n-grams to represent source and target tokens, instead of bag of words. They claim that their translation system improves performance by 0.5 BLEU points for three different language pairs.

Using this discriminative lexicon model along with incorporating a neural network joint model [DZH+14], Chollampatt et al. [CTN16] improved the performance of a phrase- based statistical machine translation (SMT) baseline. They observed that adding neural network models help in leveraging contextual information, which is a major limitation of SMT models, offering significant improvements in accuracy. Deriving from their insight, our work considers improving the architecture of NNGLM after understanding the impact of various configurations on model performance.

3. Neural Network Global Lexicon Model

NNGLM calculates the probability of individual possibility of words by training a density estimation model. The model uses a binary bag of words representation for the source and target sentences. Thus, the input and output is constituted of a binary column vector. For an input sentence, the vector spans the dimension of a source vocabulary that is constituted of all possible words a sentence can contain. The vector contains value 1 at indexes corresponding to the existence of a word from the vocabulary, and 0 otherwise. Similarly, the output vector spans the dimension of a target vocabulary and correspond to the model's hypothesis for individual word probabilities given the source sentence S computed according to the following equation:

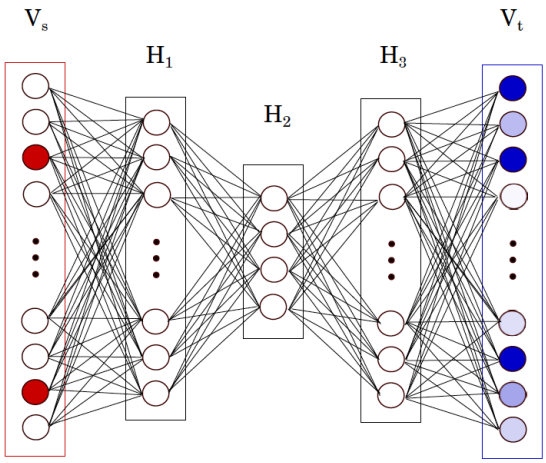

Figure 1. Employed Neural Network Global Lexicon Model

$$p(t|S) = \prod_{j=1}^{|T|} p(t_j|S)$$

where $t_j \in V_t$, i.e. the target vocabulary.

Our model uses a Feed Forward Neural Network (FFNN) with three hidden layers for NNGLM, as shown in Figure 1 [HNW15]. The network outputs $p(t_j|S)$ which is given as the result from the sigmoid activation of the output layer. Thus, the network computations are represented as below:

$$p(t_j|S) = \sigma(W^{[4]}.a^{[3]} + b^{[4]})$$

$$a^{[3]} = \sigma(W^{[3]}.a^{[2]} + b^{[3]})$$

$$a^{[2]} = \sigma(W^{[2]}.a^{[1]} + b^{[2]})$$

$$a^{[1]} = \sigma(W^{[1]}.X + b^{[1]})$$

Here, X represents the bag of words representation of the source sentence, σ is the sigmoid activation function, and $W^{[4]}$, $W^{[3]}$, $W^{[2]}$, $W^{[1]}$, $b^{[1]}$, $b^{[2]}$, $b^{[3]}$, and $b^{[4]}$ are the parameters of the NNGLM.

The parameters are determined by training the model with mini-batch gradient descent by back-propagation. The network works as a multi-variate binary classifier which gives the probability of occurrence for potential words in

the target vocabulary. The cross entropy loss function is used for optimization of parameters, given as below:

$$E = \frac{-1}{|V_t|}\sum_{j=1}^{|V_t|}[T_j \log p(t_j|S) + (1 - T_j) \log(1 - p(t_j|S)]$$

where T is the desired corrected output sentence vector obtained from the training data, T_j is the desired binary class for the j^{th} index of the target vocabulary, and $p(t_j|S)$ is the network's hypothesis for j^{th} index of the target vocabulary.

4. Setup and Experiments

To speed up the computationally expensive training of neural network's model, we have used Python's package Pytorch for flexibility and speed. The sections below give information about key data statistics and the results obtained with different network configurations.

4.1. Data

NUCLE 2014 dataset [DNW13] is used for training and is preprocessed to exclude irrelevant or unpaired data. It consists of 1,414 essays written by students who are non-native speakers of English, on a wide range of topics, such as environmental pollution, healthcare, etc. The training set contains one million words and is completely annotated with error tags and corrections. The tokenized data is extracted along with annotated corrections for wrong sentences, keeping the information necessary to establish the position of the edit. As mentioned before, two vocabularies- one for the source and the other for the target, are defined as V_s and V_t.

The training data is cleaned by removing sentence pairs that are empty or incomplete, or are too many tokens to make valuable inference. Table 1 gives statistics for the cleaned training data used for the NNGLM model.

	Source	Target
Total words	1,160,640	53,210
Unique words	33,415	6,458

Table 1. Training Dataset Statistics

Apart from the training dataset, the test data used for the evaluation of participating teams in CONLL 2013 shared task is used as the development set and the test data used for the CONLL 2014 shared task is used as the test set. Table 2 gives the statistics for the training, development and test sets used.

Additionally, we also obtained the Lang-8 Corpus of Learner English v1.0 and English Wikipedia to train the SMT model and used it as described in [CTN16].

	Train Data	Dev. Data	Test Data
No. of essays	1,397	50	50
No. of sentences	57,151	1,381	1,312
No. of word tokens	1,61,567	29,207	30,144

Table 2. Dataset Statistics

4.2. Experimentation

In order to design a suitable architecture for NNGLM that can achieve good accuracy for the task of GEC, we begin by training a single-layer neural network. We start training the model with number of hidden layer neurons 1000, batch size 15 and with learning rates 0.02, 0.1 and 0.5. Simulating a simple single hidden layer containing network helps in the better understanding of the impact of parameters like learning rate, number of neurons in the hidden layer, number of hidden layers, number of epochs, batch size, etc. on the performance of the model.

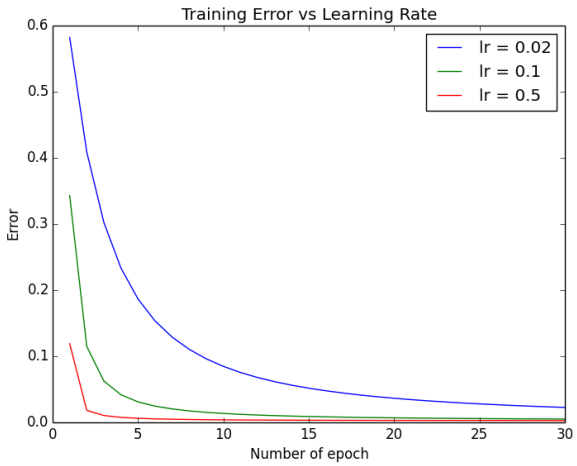

Figure 2. Plot showing Training Error v/s Learning Rate

As can be seen from Figure 2, learning rates 0.1 and 0.5 are high learning rates which though decay faster, can get stuck at worse values of loss. For this reason, we pick the optimal learning rate of 0.02.

Next, keeping the learning rate at 0.02 and number of neurons 1000 as before, we vary the batch size to find its optimum value. The plot shown in Figure 3 shows the effect of batch size on test error. As evident, the best test error was obtained for a batch size of 15.

Having found good values for the learning rate (=0.02) and batch size (=15), we explore the effect that the number of hidden layer neurons have on the error. The plot shown in Figure 4 depicts that while training error does not show a stark difference with varying number of neurons, we get better test error results when number of hidden layer neurons is 1000.

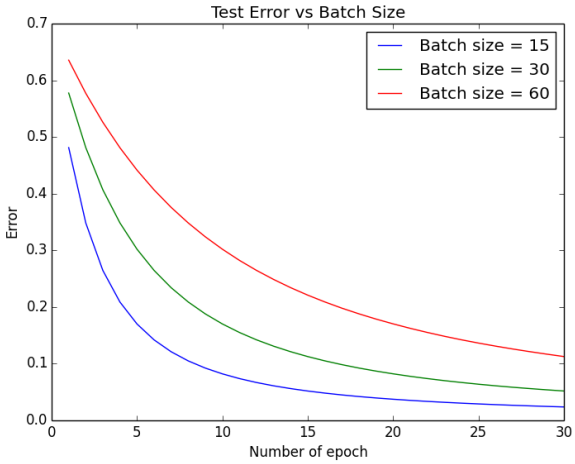

Figure 3. Plot showing Test Error v/s Batch Size

Further, obtaining optimal values of the parameters based on the experiment runs (learning rate = 0.02, batch size = 15, and no. of neurons for hidden layer 1 = 1000), we investigate the effect of increasing the number of hidden layers to the model's accuracy. We train a 2-layer FFNN with size of the second hidden layer = 500 and a 3-layer neural networks with size of the third hidden layer = 1000 (other hidden layers have the same number of neurons). As evident from the plots in Figure 5, increasing the number of layers increases the model accuracy, with the FFNN containing three hidden layers giving a test error of 0.013. All these experiments and results provide future direction to improve our model.

Based on these experiments, the parameters for the NNGLM were chosen as follows. The batch size was chosen to be 15, the learning rate was kept 0.02 and the number of neurons for the three hidden layers were decided to be 1000, 500, and 1000, respectively. This NNGLM architecture was integrated with a phrase-based SMT model trained according to the SMT baseline implemented in [CTN16]. The SMT model produces five target hypothesis each of which are scored by the NNGLM by accumulating the probabilities of individual words present in the hypothesis as given by the NNGLM model. The translation model for the SMT system was trained on a combination of NUCLE and Land-8 Corpus of Learner English v1.0. Using the default features from Moses, like forward and inverse lexical weights, forward and inverse phrase translation probabilities, word penalty and phrase penalty. The language model for the SMT is also composed using Moses and consists of two 5-gram model features, one trained using the target sentences of NUCLE and other using the English Wikipedia that consists of around 1.78n billion tokens. Using a big and varied dataset for the training of the constituent models of SMT helps in making appropriate correction predictions. This is especially relevant for the language model which ensures the fluency of the translation prediction. This SMT model alone, without addition of the NNGLM, is our considered baseline for comparison of results.

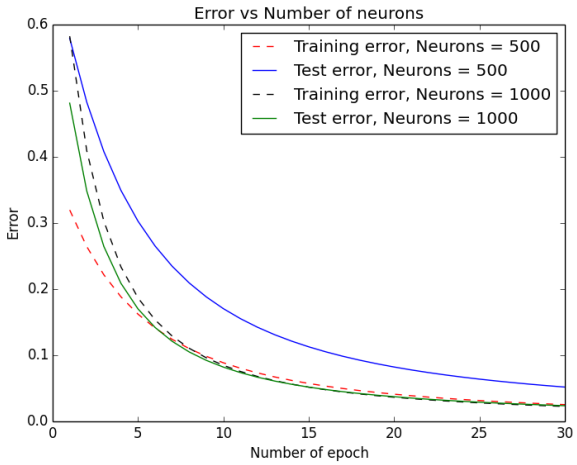

Figure 4. Plot showing Error v/s No. of Neurons

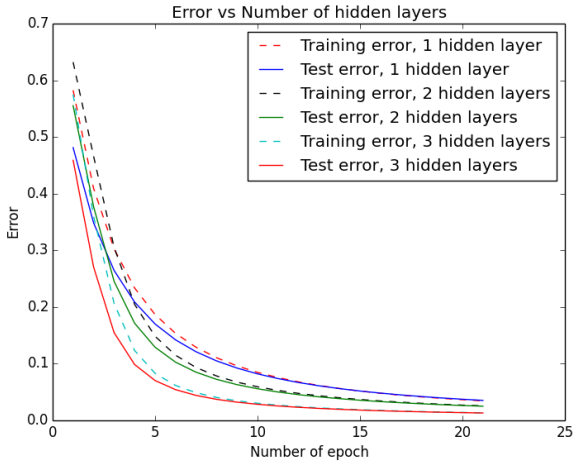

Figure 5. Plot showing Error v/s No. of Hidden layers

5. Results

The performance of a GEC model is determined by calculating a $F_{0.5}$ score which places higher emphasis on the precision of correction than just identifying a possibility for correction. Given a gold-standard edits, which are corrections suggested by a human, the $F_{0.5}$ score is given in terms of precision of correction P and recall or identification of potential for correction R as follows.

$$R = \frac{\sum_{i=1}^{n} |g_i \cap e_i|}{\sum_{i=1}^{n} |g_i|} \qquad P = \frac{\sum_{i=1}^{n} |g_i \cap e_i|}{R + 0.5^2 \times P}$$

$$F_{0.5} = \frac{(1+0.5^2)\times R\times P}{\sum_{i=1}^{n} |g_i|}$$

Here, n are the number of sentences, g_i is a set of gold-standard edits for sentence i and e_i is a set of predicts edits for the sentence i.

A system giving a better $F_{0.5}$ score is considered to be offering a better error correction. It is worth mentioning here that the performance measure can sometimes be faulty as it relies heavily on the gold-standard edits, i.e., the corrections suggested by a human annotator. In the absence of an absolute unique correction to a wrong sentence, a correct translation offered by a system may still be marked wrong if it does not match with the referenced gold edit. Therefore, two annotator references were used in [NWB$^+$14] to achieve a more accurate scorer. However, in our work, we have chosen a single gold-standard edit.

Table 3 presents the results obtained for our proposed model with the training, development and test data.

	Training data	Dev. data	Test data
Precision	52.26	50.84	50.76
Recall	23.67	23.39	23.43
$F_{0.5}$	42.09	41.17	41.15

Table 3. System results on the used dataset

To get a relative measure, the proposed system's performance is compared with the SMT baseline and the baseline added with NNGLM as used in [CTN16] (Ref. 1) in Table 4. As can be clearly seen, the proposed architecture for the NNGLM benefits its application for GEC and improves the SMT baseline by a $F_{0.5}$ score of 0.57.

	SMT baseline	Ref. 1	Proposed
Precision	50.56	50.73	50.76
Recall	22.68	23.21	23.43
$F_{0.5}$	40.58	41.01	41.15

Table 4. Relative performance measure

Also, an output example generated by the system is presented in Table 5 to offer a qualitative analysis. In the first example, the proposed GEC system predicts a correct edit for the incorrect source sentence while the SMT baseline, though detects the error, makes an incorrect suggestion.

Considering the second example, the proposed system correctly detects and suggests the correction while the SMT baseline misses the error.

6. Discussion and Scope

Though NNGLM improves the GEC accuracy of the SMT baseline, there are certain inadequacies in their use. Firstly, the NNGLM does not consider the word sequencing, focusing mainly on the choice of words that should appear in the output. However, encoding the complete lexical information from the source maybe unnecessary sometimes as many parts of the source context don't add any valuable correction quality to the prediction. Secondly, the probabilities of the individual words will be skewed towards zero because of the imbalance in the training data. Words like 'the' or 'of' will definitely be present more number of times in the training data than other non-generic words. Thus, rescaling becomes important as done in [CTN16]. Also, sometimes silly errors like spelling mistakes may not be present in the vocabularies for the NNGLM, which will lead to non-detection of error.

Using a neural network joint model that keeps track of sentence sequencing can help mitigate these failures. Moreover, using other neural network schemes like Recurrent Neural Networks (RNNs) or Convolutional Neural Networks (CNNs) also give good results for GEC, like used in [XAA$^+$16]. The research in applying neural networks to GEC is developing at a fast rate, and improved results can be achieved by developing hybrid models, preferably employing advanced tools from natural language processing. Also, training significantly impacts the performance of such neural network based systems, and use of varied, clean and effective training data shall surely help.

7. Conclusion

This work focuses on improving the architecture of a Neural Network Global Lexicon Model (NNGLM) and integrate it with a phrase-based Statistical Machine Translation (SMT) model to make an effective system that can automate the task of grammatical error correction with enhanced results. The experiments conducted show that using a feed forward neural network with three hidden layers, with appropriately selected parameters, helps improve the $F_{0.5}$ score compared to the SMT baseline by 0.57.

Source	At the same time, we are prepared to know when there are other members got this disease.
Baseline System	At the same time, we are prepared to know when there are other members *that* got this disease.
Proposed System	At the same time, we are prepared to know when there are other members *who have* got this disease.
Source	Above all, life is more important than secret.
Baseline System	Above all, life is more important than *secret*.
Proposed System	Above all, life is more important than *secrets*.

Table 5. Output example for qualitative analysis

8. Acknowledgement

This work was a part of the CS229 Machine Learning course at Stanford University conducted by Prof. Andrew NG and Prof. Dan Boneh. We would like to thank them and the TAs, especially Ziang Xie, for their guidance and valuable insights that helped accomplishing this project.

9. Contributions

Ayush Gupta: Training, Testing, Results, Report
Jaspreet Kaur: Preprocessing, Report
Yokila Arora: Preprocessing, Training, Results, Report

References

[BDG06] Chris Brockett, William B Dolan, and Michael Gamon. Correcting esl errors using phrasal smt techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 249–256. Association for Computational Linguistics, 2006.

[BDVJ03] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[BFB17] Christopher Bryant, Mariano Felice, and E Briscoe. Automatic annotation and evaluation of error types for grammatical error correction. Association for Computational Linguistics, 2017.

[CTN16] Shamil Chollampatt, Kaveh Taghipour, and Hwee Tou Ng. Neural network translation models for grammatical error correction. *arXiv preprint arXiv:1606.00189*, 2016.

[DNW13] Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. Building a large annotated corpus of learner english: The nus corpus of learner english. In *BEA@ NAACL-HLT*, pages 22–31, 2013.

[DZH+14] Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard M Schwartz, and John Makhoul. Fast and robust neural network joint models for statistical machine translation. In *ACL (1)*, pages 1370–1380, 2014.

[FYA+14] Mariano Felice, Zheng Yuan, Øistein E Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. Grammatical error correction using hybrid systems and type filtering. In *CoNLL Shared Task*, pages 15–24, 2014.

[HNW15] Thanh-Le Ha, Jan Niehues, and Alex Waibel. Lexical translation model using a deep neural network architecture. *arXiv preprint arXiv:1504.07395*, 2015.

[NWB+14] Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. The conll-2014 shared task on grammatical error correction. In *CoNLL Shared Task*, pages 1–14, 2014.

[XAA+16] Ziang Xie, Anand Avati, Naveen Arivazhagan, Dan Jurafsky, and Andrew Y Ng. Neural language correction with character-based attention. *arXiv preprint arXiv:1603.09727*, 2016.