**АВТОМАТИЧЕСКОЕ ОПРЕДЕЛЕНИЕ ОШИБОК В УЧЕБНЫХ ЭССЕ НА АНГЛИЙСКОМ ЯЗЫКЕ С ИСПОЛЬЗОВАНИЕМ BERT**  
Торубаров И. С. (istorubarov@edu.hse.ru)  
Национальный исследовательский университет «Высшая школа экономики», Москва, Российская Федерация

**AUTOMATED DETECTION OF ERRORS IN ENGLISH LEARNER ESSAYS USING BERT**  
Torubarov I. S. (istorubarov@edu.hse.ru)   
National Research University Higher School of Economics, Moscow, Russia

In this study we research the performance of a recently introduced neural network model BERT on some error detection tasks, namely, on labelling context-related errors. We explore the existing methods of higher-level language processing and highlight the features of BERT which have made us select it as our primary method. We describe the previous research on the problem and show that our model outperforms the previously suggested solutions. Next, we apply BERT to some more specific context-context error types: we elaborate on the procedure of generating the dataset, setting up and training the neural network, after which we introduce metrics to assess its performance and discuss the produced outcomes. We also discuss the performance of the model on actual texts and draw out ways to improve its output, as well as conclusions on future use of BERT as applied to error detection, its capabilities and limitations.

**Key words:** natural language processing, machine learning, error detection, learner texts, semantic tasks, BERT.

1. **Introduction**

Recent trends show that NLP methods are being successfully implemented in a growing range of various areas. The general reason for this is the fact that natural language processing is rapidly evolving and becoming capable of tackling more and more complex tasks. This growth can be illustrated by figures from GLUE benchmark [10], which is considered to be setting up a standard for assessing competence and performance of various artificial language models in complex text processing. The latest research has been steadily pushing the boundaries in such high-level text interpreting tasks as counting semantic text similarity, checking linguistic acceptability, answering to open-ended questions, et cetera.

On the other hand, the task of automatic assessment of learner texts has always been a popular topic in natural language processing. This interest naturally comes from the high demand for reliable text rating services, as they can produce results almost instantly and can also give notable insights into specific types of assignments, given that they have been trained on the latter. Nevertheless, the work in this field concentrates mainly on statistic approaches (which are being gradually abandoned in modern models) and grammar errors, whereas research using modern text processing features, let alone the one related to semantic aspects of the text, continues to be sparse [6], [13]. It has been shown that context-related errors, while being one of the most common types of errors attested in the learner corpora, are generally not covered by Automated Error Evaluation systems.

Our goal was to analyse the performance of modern NLP models on lexical errors in English learner essays. More specifically, we were interested in researching the efficacy of newly introduced Transformers, which are generally considered to be a significant milestone in researchers’ community, setting the new boundaries on all the main natural language processing tasks []. Therefore, the primary objective of our study was to create the best possible method to detect the general errors in learner essays, whereas our secondary objective was to research Transformers performance in terms of their efficiency in working with error recognition tasks.

During the development stage, we ran a series of architecture tests to find the optimal model parameters. Having uncovered those, we divided the task in two parts: first, we compared our procedure with existing state-of-the-art results from previous research, and then we obtained actual textual results by applying the model to specific error types from our research corpus. The corpus used for the latter task and in development stage is REALEC [4], an error-annotated corpus of English essays written by university students, L2 learners of English, during their examination at Higher School of Economics.

1. **Existing solutions and BERT applicability**

The task of detecting errors is designed as follows: as our input data, we have a completed plain text, for which we have to test each for the possibility of being erroneous. To maximise the precision, we should consider both left and right contexts, since either of them can influence the scope in question. Ideally these contexts should be equal to the remaining part of the text, as semantic ties can possibly stretch over the whole document. However, firstly, no models can currently cover this depth, and, secondly, distant ties are generally less frequent and less strong than the closer ones, so we will have to deal the complexity to performance.

Previous research on the matter can be divided in three epochs: one of rule-based models, followed by statistical and then neural models, with each type gradually outperforming its predecessor. For instance, rule-based models were gradually abandoned in AEE systems [6], and none of the models submitted for The SIGNLL Conference on Computational Natural Language Learning Shared tasks on error detection in 2013 and 2014 could be described as rule-based (if not accounting for simple heuristics) [2013], [2]. Then, the winning statistical model of the latter competition was outperformed by neural network in [7], while a simple statistical CRF model had ranked the lowest in the same study. As of now, all the competent models in GLUE leaderboard have outperformed CBOW [GLUE]. This is why we consider the baselines for our task as being produced by neural network-based methods.

The best existing architecture for the task comes, to our knowledge, from [7] and is a bidirectional LSTM network. This was the most logical choice for the task prior to introduction of Transformer models, as it deals with both left and right contexts, is designed to overcome the problem of rapidly falling significance of prior words and can be initialized with word embeddings, which have proven to perform better than conventional 1-of-K encoding (Huang 2015). Our interest was focused on bidirectional models, since we would have completely lost the relations bound to the right scope otherwise. The scope of a lexical error is generally broader than, for example, that of a grammatical one, and this further limited our options, as older models can't handle depth, because the significance of previous words decays too fast. Consequently, it is logical that the best model attested in [7] was based on bidirectional LSTM (or Bi-LSTM) architecture; however, being first introduced in 2014, it has accumulated some notable disadvantages. For instance, the forward-propagation of the word makes it prone to being corrupted in the process, creating the problem of vanishing gradients. The other problem arisen was that we could not find any publicly available pre-trained embeddings for Bi-LSTM (although there are some for unidirectional LSTM networks), and here is where Transformer models, namely BERT (or Bidirectional Encoder Representations from Transformers [3]), have their first advantage over the competition.

BERT is a neural network built by Google Research team, which main feature is combining the Transformer layers architecture [9] and bidirectional text processing. It has shown to perform better on text tasks than the state-of-the-art models on the time of its release [GLUE]. It is equally important that BERT is a publicly available product, released under the Apache 2.0 license.

1. **Procedure**

In our research we make advantage of the two-layered architecture of BERT, using the pre-trained uncased\_L-12\_H-768\_A-12 embedding, which is recommended by the authors as being more preferable for non-case-sensitive tasks and further proven to be so in our preliminary research, further described in (Torubarov 2019). We consider error detection as a binary classification task, providing the model with the potentially erroneous substring paired with its context and expecting the model to classify the entry as either erroneous (1) or acceptable (0). The substrings are matched to single words as produced by TweetTokenizer coming from NLTK’s punkt module: this is chosen to match human’s intuition of what a word is as close as possible (avoiding, for example, splitting *has* and *n’t* in two different tokens by word\_tokenize). To make advantage of BERT ability to capture longer arrays of text, we set the default length of our context as three sentences (parsed by NLTK’s sent\_tokenize): that containing the word being assessed, the one before and the one after it. If the sentence in question happens to be the first or the last in the document, we naturally limit our scope to 2 sentences (or 1 if it is the only sentence present). Apart from the described length, here is how some exemplary inputs and outputs of the model could look like:

|  |  |
| --- | --- |
| **Input** | **Expected output** |
| Nowadays [MASK] is becoming cosier to express <…> | 0 |
| It |
| <…> before you earn a big [MASK] of money. <…> | 1 |
| number |
| <…> For instance, I [MASK] people who have <…> | 0 |
| Know |
| <…> freedom to express [MASK] is important <…> | 1 |
| idea |
| *Fig. 1. Exemplary inputs and outputs* | |

Note the [MASK] token indicating the processed substring location: this is a special placeholder for a missing word used in pre-training of BERT (Devlin 2018). This is the closest for what our task necessities is and is proven to be more beneficial for the model than no error span indication whatsoever by our previous research in (Torubarov 2019). This is also where we have tested different combinations of model parameters and derive the best-performed specifications from, most importantly, the following:

|  |  |
| --- | --- |
| Learning rate | 2e-5 |
| Batch size | 24 |
| Checkpoint epoch increment | 1.5 |
| Maximum tokens | 128 |
| Embedding | uncased\_L-12\_H-768\_A-12 |
| Output layer | truncated\_initializer |

For fine-tuning means, we parse the presented training and test sets using the procedure described above and then randomly shuffle the dataset. In production, we simply iterate over all of the words in the given text, replacing each of them with [MASK] token and pairing with the described broader context.

1. **Proving the adopted method plausibility**

To prove the efficacy of our method, we compare it with the notable results from the previous research: namely, with the compositional sequence labelling neural models from (Rei, Yannakoudakis 2016). We tested our method by accounting all the words deemed erroneous in the described datasets as positive examples. For the purpose of attesting the errors linked with missing words, we follow (Rei, Yannakoudakis 2016) and mark the following word as erroneous. We compare BERT outputs with CRF and Bi-LSTM performance on FCE corpus [FCE]. FCE combines multiple types of errors and deems all of the words falling into the span of a large error as incorrect, however, regrettably, it lacks proper sentence alignment, which caused us to adjust our procedure to account only for one sentence. For recomposing the tokenized sentences we used nltk.MosesDetokenizer from perluniprops modul. Our model outperformed the competitors by a big margin:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Performance on test set** | | | | |
| **Predicted** | **Correct** | **Precision** | **Recall** |  |
| CRF [7] | 914 | 516 | 56.5 | 8.2 | 25.9 |
| Bi-LSTM [7] | 3898 | 1798 | 46.1 | 28.5 | 41.1 |
| BERT, 6 epochs | 4494 | 3002 | 66.8 | 47.61 | 61.82 |
| *Fig 2. Performance of BERT compared to prior research on FCE-public dataset* | | | | | |

This proves that usage of BERT in error detection can be very advantageous, as it demonstrates a great performance in difficult concepts, such as general unspecified errors, setting the new bars on error detection. With this, we consider the applicability of our method proven: this brings us to the point where we apply the model to three specific error types and discuss the textual outputs.

1. **Applying the method to specific errors**
   1. *Method description*

As the concept of an error in general is very broad, we chose to narrow the frame of which mistakes we are looking for. This way, we could get a more limited number of results which is much easier to focus on, and also design a procedure to apply BERT error detection to specific layers of language. To make further use of BERT advantages, we could also choose context-related errors, which are generally dismissed by the conventional AEE systems.

We worked with open-source corpus REALEC for the reason that we could maintain stable contact with its annotators and administrators. Since we empirically limited the number of errors needed to successfully train a model to at least 1000, this left us with this three most-occurring contextual errors: Choice of lexical item, Deletion (not an error per se, but a case in which a specific token should be deleted to correct the sentence) and Agreement errors.

Our main challenge was to choose the way to select the incorrect entries. One may recall that we limit our potentially erroneous substring scope to exactly 1 token and process text word-by-word. With this meaning that only one word would be masked in each context in production, it makes no sense to construct entries with more than one [MASK] token. But what if an error spans more than one word? There may be two strategies: the first is to omit the n-gram errors whatsoever, leaving only single-token errors both in training and test (let us call it the single-token method), while the second way is to treat each word in an n-gram erroneous span as an independent error, creating n erroneous entries by masking the affected words iteratively (this we will call the iterative method). We actually tried both approaches in our research and will draw the outcomes for each of them.

For each type of errors, we selected 15 texts in which the type in question was attested at least once. These texts were selected to remain intact for final evaluation and no excerpts from these texts appeared in the training nor test subsets. We followed our previous research again to set the 1:14 erroneous — non-erroneous entries ratio as it has shown to be optimal for the model to efficiently distinguish the erroneous and non-erroneous entries. The latter were generated randomly by selecting a random word from a random text; however, we made sure that no resulting context would equal an existing erroneous context, in order not to run into an erroneous word by chance, therefore creating ambiguity. Furthermore, we restricted the possibility of a large non-erroneous context matching any of clean n-gram contexts just for precaution.

We measure the efficiency of our model using the same score, but for this task we construct two datasets: one with the same 1:14 error ratio as in training and another with ratio matching that what is observed in factual corpus data. We also use simple percentage metric to overcome the problem of context errors having fuzzy boundaries and potential high number of incorrect false positive detections due to corpus underannotation: for this, we asked our annotators to choose whether the model made a correct decision or not and then count the percentage of correct decisions.

* 1. *Choice of lexical item errors*

Our best results for *Choice of lexical item* type of errors come from the 1.5th epoch of a single-token model training and 7.5th epoch of iterative model.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Type** | **Epoch** | **1:14 test** | | | **Real test** | | |
| **Precision** | **Recall** |  | **Precision** | **Recall** |  |
| Single-token | 1.5 | 53,66 | 19,79 | 39,97 | 11,11 | 20 | 12,2 |
| Iterative | 7.5 | 84,89 | 99,16 | 87,4 | 38,34 | 99,49 | 43,71 |

*Fig 3. Best results for Choice of lexical item errors*

Surprisingly, the difference here is staggering, with the iterative model beating the results of single-token one 3 to 4 times. We consider this fact to be linked with the specific architecture of BERT, which learns the importance of words in a context – when presented with identical excerpts of text, which differ only by one masked word, its context attention comes into play, allowing it to learn the underlying patterns of the error.

Choice of lexical item errors are the most interesting, albeit the most complicated sphere of automated error detection. This is due to the fact that semantic errors

To prove our claim on fuzziness of lexical errors, here we list some cases of detection for which we cannot be exactly sure whether it is a correct detection or not:

|  |  |
| --- | --- |
| Text id: esl\_00583 | |
| Corpus annotation | Single-token model |
| <…> situation is not as prosperous as <…> | <…> situation is not as prosperous as <…> |
| <…> are to be noticed and seriously considered. | <…> are to be noticed and seriously considered. |
| Text id: OBy\_69\_2 | |
| Corpus annotation | Iterative model |
| <…> number of these people raise sagnificantly. | <…> number of these people raise sagnificantly. |
| Modern displays should be straight belove <…> | Modern displays should be straight belove <…> |
| *Fig. . Complex cases of detection* | |

This paradox led us to confide in numerical statistics of Agreement errors rather than that of Choice of lexical item errors beforehand. Nevertheless, there are still clearly accurate corrections, as well as obviously false detections, as one could see:

|  |  |
| --- | --- |
| Text id: esl\_00382 | |
| Corpus annotation | Single-token model |
| <…> the amount of people and cars increases <…> | <…> the amount of people and cars increases <…> |
| Text id: esl\_00583 | |
| Corpus annotation | Iterative model |
| <…> the percentage of thick old people <…> | <…> the percentage of thick old people <…> |
| *Fig. . Correct models output* | |

|  |  |
| --- | --- |
| Text id: esl\_00583 | |
| Corpus annotation | Single-token model |
| <…> situation is not as prosperous as <…> | <…> situation is not as prosperous as <…> |
| <…> are to be noticed and seriously considered. | <…> are to be noticed and seriously considered. |
| Text id: OBy\_69\_2 | |
| Corpus annotation | Iterative model |
| <…> number of these people raise sagnificantly. | <…> number of these people raise sagnificantly. |
| Modern displays should be straight belove <…> | Modern displays should be straight belove <…> |
| *Fig. . Incorrect models output* | |

One immediate proof of the usefulness of our results is the effective detection of previously unattested errors by multiple models of ours. Here are some examples illustrating this fact:

|  |  |
| --- | --- |
| Text id: esl\_00583 | |
| Corpus annotation | Single-token model |
| <…> situation is not as prosperous as <…> | <…> situation is not as prosperous as <…> |
| <…> are to be noticed and seriously considered. | <…> are to be noticed and seriously considered. |
| Text id: AAl\_9\_2 | |
| Corpus annotation | Iterative model |
| <…> no matter in acceptance of such kind <…> | <…> no matter in acceptance of such kind <…> |
| <…> educate you withough such separation. | <…> educate you withough such separation. |
| *Fig. . Previously unattested errors detection* | |

That being said, we do suffer from an exceptionally high number of false positive detections. This can be illustrated with the %true metric, for the purpose of calculating of which, we asked two expert REALEC annotators to assess whether our models have assigned the tokens correctly or not, putting together the following result:

Judging by these data, the Uncased model emerges as the winner, although with a narrow margin. To sum up, our best models, albeit uncovering some insightful unattested errors, do clearly tend to assign more erroneous tokens than our human baseline mean, which should not be the case. Somewhat fortunate, this is better suited for a possible fix than the opposite, as this is the overly conservative models.

* 1. *Deletion errors*
  2. *Agreement errors*

1. **Deploying REALEC AutoAnnotator service**

We deployed our selected trained models on a server along with the trained feed-forward network. The user in encouraged to input a text, which is then processed by the models (this process may take up to a dozen of seconds). The result is then presented in a user-friendly way. There is no limit on the text content. Please refer to [AA] for the online version of the service, while the following screenshot illustrates a produced text.

1. **Results discussion and conclusion**

Given the outcomes of our research, we believe that BERT has proven to be a valuable tool for error detection in learner texts. Having said that, developing the optimal ways for its implementation is still a topic for further research. We expect that one could experiment with all the variables we used in our procedure and find a better combination. We also suppose that a better model stacking combination could be proposed.

One perspective field of research would be to train BERT on other types of errors and other datasets, a method which has shown to be beneficial in [7], multiplying the knowledge of the model. Given that we follow the same error structure as REALEC, it is a possibility that REALEC AutoAnnotator service will see its extension in this way.

One could also further research in the field of trying to feed some morphological information or provide some other sort of linguistic analysis, yet this seems less probable to succeed considering that BERT infers its predictions based on the combinations of embedding vectors, which are very sensible to the integrity of a sentence. We also do not count off the possible enhancement of results via designing a neural model more efficient than BERT or empowering BERT with a better embedding (a way that we could not possibly try due to computational restrictions).

Our results have shown BERT to be a valuable tool for corpus annotators, able to provide a valuable prediction, upon which one could base its annotation. It is also a good measure for the general level of corpus annotation and a way to uncover some insightful corpus entries. However, REALEC AutoAnnotator cannot be currently considered a flawless standalone tool for student self-evaluation. One may get a grasp on where the possible errors might be in his text but has to give the service a benefit of a doubt and proceed with caution.

All our proceedings are made publicly available and can be accessed at [8]. We are currently planning to maintain the development of this project, considering some of the aforementioned possible improvements, with our ultimate goal being to provide a new perspective on how we organise our corpora and how powerful automated essay assessment can be.

1. **Acknowledgements**

We would like to thank Ekaterina Gerasimenko (bachelor, NRU HSE), Elizaveta Ershova, Darya Overnikova (both – undergraduate students, NRU HSE), and, above all, Olga Vinogradova (Associate Professor, NRU HSE) for their invaluable help across various stages of the development of this research.

## Bibliography

1. bak. (2018) Исправляем опечатки с учётом контекста // Habr.com website. 24th of January. (<https://habr.com/ru/post/346618/>)
2. Bryant C., Ng H. T. How Far are We from Fully Automatic High Quality Grammatical Error Correction? //Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). – 2015. – Т. 1. – С. 697-707.
3. Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding //arXiv preprint arXiv:1810.04805. – 2018.
4. Kuzmenko E., Kutuzov A. Russian Error-Annotated Learner English Corpus.
5. L. Yang. Fine-tune BERT for Extractive Summarization //arXiv preprint arXiv:1903.10318. – 2019.
6. Leacock C. et al. Automated grammatical error detection for language learners //Synthesis lectures on human language technologies. – 2014. – Т. 7. – №. 1. – С. 1-170.
7. Rei M., Yannakoudakis H. Compositional sequence labeling models for error detection in learner writing //arXiv preprint arXiv:1607.06153. – 2016.
8. Torubarov I. (2019) Research for automated annotation of errors in REALEC, a corpus of English learner essays // Github.com website. 30th of April (<https://github.com/isikus/realec-autoannotation>)
9. Vaswanwe A. et al. Attention is all you need //Advances in neural information processing systems. – 2017. – С. 5998-6008.
10. Wang A. et al. Glue: A multi-task benchmark and analysis platform for natural language understanding //arXiv preprint arXiv:1804.07461. – 2018.
11. Wu Y. et al. Google's neural machine translation system: Bridging the gap between human and machine translation //arXiv preprint arXiv:1609.08144. – 2016.
12. Yannakoudakis H., Briscoe T., Medlock B. A new dataset and method for automatically grading ESOL texts //Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. – Association for Computational Linguistics, 2011. – С. 180-189.
13. Zupanc K., Bosnić Z. Automated essay evaluation with semantic analysis //Knowledge-Based Systems. – 2017. – Т. 120. – С. 118-132.