A Generic Framework: Under Calibrating for Nonautoregressive Grammatical Error Correction

Anonymous EACL submission

Abstract

Grammatical error correction (GEC) can be applied in user-facing applications to help language learners improve their writing skills. However, existing GEC systems tend to propose incorrect corrections or overlook lots of grammatical errors. Confidence calibrating, which aims to estimate the true probability, is important for GEC because it can provide useful probabilistic information to the user. Whereas calibrating is well-studied for a classification task, we note that it is still challenging for GEC. We present a generic framework to observe miscalibration and for calibrate existing GEC models. We show how calibration of a GEC model is related to the performance on benchmark datasets. Our further experimental results show that re-calibrating improves the performance of GEC models on the benchmark datasets. The code to reproduce our experiments is available on GitHub at: https://github.com/****

1 Introduction

Grammatical Error Correction (GEC) is a task of automatically finding and correcting grammatical errors in a text; eg. [*Does he* ***trusts*** *it?* $\rightarrow$ *Does he trust it?*]. In the above sentence, the verb in bold "trusts" needs to be corrected to its base form "trust". Existing GEC models are prone to propose spurious corrections or overlook lots of errors in spite of their state-of-the-art (SOTA) performances on a benchmark. Lest misleading information was made to the user, it could be imperative that the model can indicate when it is likely to be correct. Despite only a few studies that address using a deep neural network model to estimate hypotheses for GEC (Liu et al., 2021; Chollampatt and Ng, 2018), our work provides a confidence that estimates the true probability.

By measuring calibrating for GEC models, our work relates the issue of calibrating and the problem that a current nonautoregressive GEC model could only detect few errors in a sentence. To improve output quality, a nonautoregressive model usually feeds the output back into itself multiple times. We observed an intrinsic bias of the iterative refinement technique. Our results achieved SOTA on benchmarks.

Secondly, multiple valid hypothesis may differ for the corrections in a single source sentence for GEC (Napoles et al., 2015). There can be more than one correct annotation for an annotator. Liu et al. (2021) find that a more accurate hypothesis can be produced by scoring and re-ranking the multiple candidate hypotheses obtained from beam search. We re-calibrate confidence for GEC and use well-calibrated confidence to further re-rank multiple top-k hypotheses. We observe improvement of performance on benchmarks for the GEC task. Our experiments show that calibrating can further increase SOTA performance of a SOTA GEC model on a benchmark, without any additional data.

2 Backgrounds

2.1 Grammatical Error Correction Task

GEC has been thought of as an structure transduction (ST) problem from the effort by Yuan and Briscoe (2016) to the effort by Rothe et al. (2021). ST refers to mapping a sequence $x = [x_1, ..., x_M]$ to a sequence $y = [y_1, ..., y_T]$, where M is the length of an input and T is the length of an output. For an ST GEC task, x is a sequence of words that may contain errors and y is a sequence of words that are corrected and provided by an annotator. GEC is also deemed as an structured prediction (SP) problem like tagging (Omelianchuk et al., 2020; Awasthi et al., 2019). SP refers to predicting an SP y for a given input x. For a tagging GEC task, $x = [x_1, ..., x_M]$ is a sequence of words that may contain errors and $y = [y_1, ..., y_M]$ is a

sequence of tags. An annotator assigns a tag y_i in y at a position i to a word x_i in x at a position i, where $i \in \{0, ..., M\}$. In a post-processing process, each word in x is transformed according to the transformation corresponding the tag assigned to the word; thus x is transformed into a sentence which we denote as x^y.

2.2 Grammatical Error Correction Models

An AR model factors a conditional probability of y into a product of conditional probabilities in Equation 1. A factored conditional distribution $p_\theta(y_t|x, y_{<t})$ over a token y_t at a time step t in y is conditioned on the input x and the partial output sequence $y_{<t} = [y_1, ..., y_{t-1}]$, where $t \in \{0, ..., T\}$.

$$p_\theta(y|x) = \prod_{t=1}^{T} p_\theta(y_t|x, y_{<t}). \quad (1)$$

An existing tagging GEC system usually employs an NA model like that by Omelianchuk et al. (2020). An NA model (Gu et al., 2018) factors a conditional probability of y into the conditional probabilities in Equation 2.

$$p_\theta(y|x) = p_\theta(T|x) \prod_{t=1}^{T} p_\theta(y_t|x). \quad (2)$$

An existing GEC models is trained to minimize the following cross-entropy object function, where $(x^{(j)}, y^{(j)})$ is a tuple of an input-output training pair from a training set $D = \{(x^{(j)}, y^{(j)})\}_{j=1}^{|D|}$ ($j \in \{1, ..., |D|\}$) and θ is a set of training parameters:

$$L(\theta, D) = -\sum_{j=1}^{|D|} log(p_\theta(y^{(j)}|x^{(j)})). \quad (3)$$

2.3 Grammatical Error Correction Decoding

SOTA GEC systems usually employ an autoregressive (AR) model (Rothe et al., 2021; Stahlberg and Kumar, 2021) or a non-autoregressive (NA) model (Omelianchuk et al., 2020). At inference time, a GEC model needs to find a hypothesis y' that gives the maximum probability from the conditional distribution $p_\theta(y'|x)$ over a hypothesis space $\gamma = \{y'^1, ..., y'^{|\gamma|}\}$. Because the search space is exponentially large, an existing model inference usually requires a greedy search or a beam search. An AR model generates a token once at a time until the special end of sentence token is generated. Greedily, at a time step t, where $t \in \{1, ..., L\}$ and L is the eventually generated sequence, an AR model needs to generate a token y'_t that gives the maximum probability from the conditional distribution $p_\theta(y'_t|x, y'_{<t})$, which is conditioned on a partially generated prediction $y'_{<t} = [y'_1, ..., y'_{t-1}]$ due to lack of $y_{<t}$. Beam search maintains *top-k* partial hypotheses at each time step and then ranks the eventual hypotheses using their confidences. An NA MT model predicts the tokens at each position independently (Gu et al., 2018). Because a transformation may depend on another (Omelianchuk et al., 2020), a SOTA NA system requires a multi-round inference by feeding a transformed sentence $x^{y'}$ into the model multiple times.

3 Annotating Correctness for GEC

For an input x, an output $y = [y_1, ..., y_T]$ an AR model and a hypothesis $y' = [y'_1, ..., y'_L]$, $T = L$ at the training time but $T \neq L$ at the inference time (Ranzato et al., 2016). Thus we cannot easily adapt a generic calibrating framework by Jagannatha and Yu (2020) for SP problem, which strictly requires $L = T$. We adapt the I-measure tool (Felice and Briscoe, 2015), which is commonly used for evaluating a GEC hypothesis at the token-level by comparing edits. For y and x, the I-measure automatically produces a sequence of output edits: $e = [e_1, ..., e_K]$, where $k \in \{1, ..., K\}$. For y' and x, the I-measure automatically produces a sequence of hypothesis edits: $e' = [e'_1, ..., e'_K]$. An output token y_t at a time step t in y is corresponding to an edit e_k at a position k in e. A hypothesis token $y'_{t'}$ at a time step t' in y' is corresponding to an edit $e'_{k'}$ in e' at a position k', where $t' \in \{1, ..., K\}, k' \in \{1, ..., K\}$. We define an output y_t in y at a time step t as a corresponding output for a hypothesis $y'_{t'}$ in y' at a time step t' if the position k for e_k corresponding to y_t equals to the position k' for $e'_{k'}$ corresponding to $y'_{t'}$. We define the correctness for the token-level hypothesis $y'_{t'}$ with the corresponding output y_t as $Cor(y'_{t'}) = I(y'_{t'} = y_t), Cor(y'_{t'}) \in \{0, 1\}$, where $I()$ is the indicator function.

For the tagging task, for a tag-level hypothesis y'_t, we transform x into a sentence $x^{y'_t}$ by performing the transformation corresponding to y'_t. We measure the Levenshtein distance (Müller et al., 2019) between the transformed (or processed) sentence $x^{y'_t}$ and the input x, which we denote as $L_{x^{y'_t}, x}$. We measure the Levenshtein distance between the gold sentence x^y and the input x, which we denote as

L_{x,x^y}. The correctness of a sequence-level hypothesis y' is defined as $Cor(y') = I(y' = y)$. The correctness for a tag-level hypothesis y' is defined in 4.

$$Cor(y'_t) = \begin{cases} I(L_{x^{y'_t},x^y} = L_{x,x^y}) & L_{x^{y'_t},x} = 0, \\ I(L_{x^{y'_t},x^y} < L_{x,x^y}) & L_{x^{y'_t},x} > 0. \end{cases} \quad (4)$$

4 Calibration and Evaluating Calibration

In a binary classification setting, calibrating requires that a confidence p of a model b_θ for an input x and an output y equals to the true probability measure $P(y = 1|b_\theta(y|x) = q)$, where P refers to true probability measure. The perfect calibration is defined as:

$$P(\hat{y} = y|p_\theta(\hat{y}|x) = p) = p, \forall p \in [0, 1]. \quad (5)$$

5 examines the true probability P of a hypothesis $h \in H(x)$ conditioned on the model's prediction $r(h, x, p_\theta) = p$.

We approximate a difference between the true probability and the confidence p with N samples in a dataset. We separate the N samples into H bins $\{B_1, ..., B_H\}$ according to the confidence. The result is multiplied by 100. We define the expected calibration error Naeini et al. (2015) (ECE):

$$ECE = \sum_{h=1}^{H} \frac{100|B_h|}{N} |Conf(B_h) - Acc(B_h)|,$$

where $Conf(B_h)$ is the average confidence for the samples within the h-th bin B_h; $Acc(B_h)$ is the accuracy for the samples within a bin B_h.

5 Experiments and Results

5.1 Calibration of GEC Models

Firure 1 shows the calibration of a pretrained SP model. Figure 2 shows the calibration of unpretrained ST model. Firstly, they are calibrated (ie., $ECE < 3$). Subsequently, a pretrained SP GEC model is seemingly less calibrated than an unpretrained ST GEC model whereas the former has a better performance on benchmarks. We assume that the model is over-fitting. And because some gaps of the SP GEC models are larger than most of the ST models, there is a large opportunity of studying pretraining of an existing GEC model.

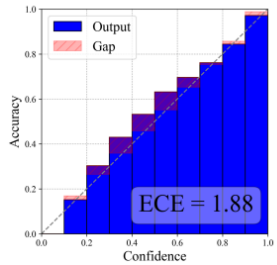

Figure 1: Miscalibration of Pretrained SP GEC

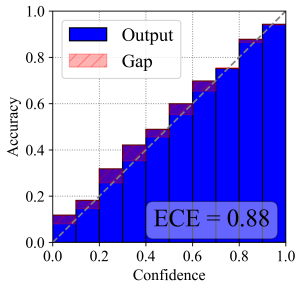

Figure 2: Miscalibration of Unpretrained ST GEC

5.2 Re-calibrating of GEC Models

5.2.1 Methods

Rerank: We acquire and rerank top-k hypotheses from a greedy search in 7:

$$\hat{y}_{tk} = \{argmax_{y_t \in V}[p_\theta(y_t|x), k]\}_k, \quad (7)$$

where $argmax[p, k]$ is a function that produces a list of top-k predictions with the highest softmax outputs; $_k$ denotes the k-th prediction in the list; V is the target vocabulary space of model.

Tweak: We can use an edit extracting tool such as ERRANT to extract edits and only take edits with top probabilities. And reject hypotheses whose confidence is lower than a threshold. The threshold is decided by a grid search.

Monte Carlo Dropout: We follow (Gal and Ghahramani, 2016) and Jagannatha and Yu (2020) to average the D ($D = 10$) MCD samples from a GEC model.

Isotonic Regression: We use Isotonic Regression (Zadrozny and Elkan, 2002) as our re-calibrating method.

LGBM: We use a Light Gradient Boosting Machine (Ke et al., 2017) (LGBM) as our control re-calibrating method.

5.2.2 Results1: Re-calibrating Methods

We firstly observed that corrections are not calibrated (ie., $ECE = 4.97$). Table 1 shows that

ID	Model	BEA Dev(dev)			CoNLL2014(test)			BEA2019(test)		
		prec.	rec.	$F_{0.5}$	prec.	rec.	$F_{0.5}$	prec.	rec.	$F_{0.5}$
①	RoBERTa NAGEC	55.5	38.1	50.9	67.2	44.3	60.9	70.6	57.4	67.5
②	①+MCD	59.3	35.7	52.4	70.4	41.3	61.7	74.3	55.0	69.5
③	②+Isotonic	59.5	35.9	52.6	69.9	41.3	61.4	73.8	54.7	69.0
④	②+LGBM	61.0	35.5	53.3	71.5	41.2	**62.3**	76.1	54.3	70.5
⑤	②+Isotonic(Cor)	62.1	34.2	**53.4**	72.2	40.2	**62.3**	77.4	53.1	**70.9**
⑥	②+LGBM(Cor)	61.6	34.8	**53.4**	71.8	40.5	62.2	76.6	53.7	70.6

Table 1: Using Calibrated Confidence to Re-rank Topk Predictions For an NAGEC (RoBERTa) Model

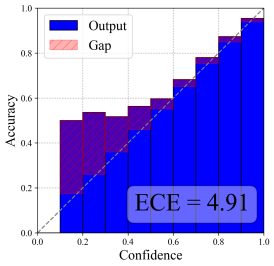

Figure 3: Calibration of Corrections

re-calibrating a GEC model improves its precision, recall and $F_{0.5}$ scores on the GEC benchmark. LGBM re-calibrating model outperforms the Isotonic method because it increase precision at the cost of recall. Since most hypotheses with high confidence are over-confident, and corrections are under-confident, re-calibrate correcting (ie., $h \neq x$) edits might improve recall. Re-calibrating corrections shows equivalent performance. Considering why Isotonic Regression performs better, it is probably that Huber loss (de Leeuw et al., 2009) makes system concentrates more on the high confidence samples to successfully alleviate miscalibration. And we assume that the non-correcting ($h = x$) edits that are annotated by an annotator contain too much new knowledge for a pre-trained model. For example, a BERT-based model, with a lower quantity of pre-training data (Liu et al., 2019), had a lower ECE than that of a RoBERTa-based model (ie., $4.88 < 5.18$ (subword level)).

Figure 3 shows the miscalibration of corrections. Figure 4 shows the miscalibration of the

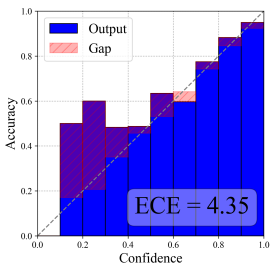

Figure 4: Calibration of re-calibrated Corrections

re-calibrated corrections using only an Isotonic method. Firsly, there might be be few samples in the interval $0 - 0.4$. Secondly, there is a higher accuracy for the confidences in the interval $0.5 - 0.6$.

5.2.3 Results2: Iterative Refinement and Strategies

	Model	BEADev(dev)			CoNLL2014(test)			BEA2019(test)		
		prec.	rec.	$F_{0.5}$	prec.	rec.	$F_{0.5}$	prec.	rec.	$F_{0.5}$
①	RoBERTa GEC×5	54.2	41.2	51.0	68.6	47.3	62.9	68.4	60.9	66.7
②	①+MCD	57.9	38.7	52.6	71.8	44.4	63.9	72.5	58.6	69.2
③	②+Isotonic(Rerank)	58.1	39.2	53.0	70.8	44.9	63.5	71.7	58.8	68.7
④	②+Isotonic(Tweak)	60.1	37.7	**53.7**	73.2	42.9	**64.2**	74.6	57.5	**70.4**

Table 2: Using Calibrated Confidence to Re-rank Topk Predictions For an NAGEC (RoBERTa) Model with Iterative Refinement

We firstly observed an intrinsic bias for the iterative refinement, that is, the $F_{0.5}$ score is 67.5 for the first round while the $F_{0.5}$ score is 66.7 of all rounds and the ECE of the first round outputs is 2.82 while the ECE is 5.4 of the latter rounds. Obviously, the model is not trained for inferences of the latter rounds. In Table 2, ④ achieves SOTA for a single RoBERTa-based NAGEC model. By reducing calibration error, we decrease mistakes of iterative refinement. Tweak method shows better performance compared to the rerank strategy. There are few correct corrections in the top-k candidates. Particularly, we achieved SOTA on the BEA-2019 test dataset ($F_{0.5} = 75.1$ ($threshold = 0.5$)) by re-calibrating and tweaking an ensemble model.

6 Conclusion and Future Work

We propose a framework for calibrating of Grammatical Error Correction. We examine several methods to alleviate the miscalibration of the GEC models. Still, correcting words are not calibrated. We present two general schemes to use calibrated probabilities. We achieve SOTA for a single RoBERTa-based NAGEC model. Our future works may consider using an AR language model to calibrate an NA model especially for the correcting edits.

7 Limitations

Insufficient samples for statistical measurements are used; we have not yet tested on a larger size dataset, where annotation rules and text quality may vary. We are limited to accessing to real-world data. Evaluating on real-world data and exploring decision-critic systems are important directions for future work. Lastly, sample bias or selection bias were not considered carefully by us.

References

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel iterative edit models for local sequence transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4260–4270, Hong Kong, China. Association for Computational Linguistics.

Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75, Florence, Italy. Association for Computational Linguistics.

Shamil Chollampatt and Hwee Tou Ng. 2018. Neural quality estimation of grammatical error correction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2528–2539, Brussels, Belgium. Association for Computational Linguistics.

Mariano Felice and Ted Briscoe. 2015. Towards a standard evaluation method for grammatical error detection and correction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 578–587, Denver, Colorado. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML 2016*.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation.

Abhyuday Jagannatha and Hong Yu. 2020. Calibrating structured output predictors for natural language processing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2078–2092, Online. Association for Computational Linguistics.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1236–1242, Hong Kong, China. Association for Computational Linguistics.

Jan de Leeuw, Kurt Hornik, and Patrick Mair. 2009. Isotone optimization in r: Pool-adjacent-violators algorithm (pava) and active set methods. *Journal of Statistical Software*, 32(5):1–24.

V. I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Zhenghao Liu, Xiaoyuan Yi, Maosong Sun, Liner Yang, and Tat-Seng Chua. 2021. Neural quality estimation with multiple hypotheses for grammatical error correction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5441–5452, Online. Association for Computational Linguistics.

Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help? *CoRR*, abs/1906.02629.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. 2015. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 588–593, Beijing, China. Association for Computational Linguistics.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14, Baltimore, Maryland. Association for Computational Linguistics.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, Seattle, WA, USA → Online. Association for Computational Linguistics.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *ICLR 2016*.

Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 702–707, Online. Association for Computational Linguistics.

Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 37–47, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS 2017*.

Shuo Wang, Zhaopeng Tu, Shuming Shi, and Yang Liu. 2020. On the inference calibration of neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3070–3079, Online. Association for Computational Linguistics.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 380–386, San Diego, California. Association for Computational Linguistics.

Bianca Zadrozny and Charles Elkan. 2002. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, page 694–699, New York, NY, USA. Association for Computing Machinery.

A Defining Confidence Calibration for Re-calibrating

We define confidence calibration of re-calibrating as follow, where P refers to the ground truth probability for a confidence p:

$$P(Cor(h)|r(h, x, p_\theta) = p) = p, \forall p \in [0, 1]. \quad (10)$$

B Constructing Entity and Event Set

We extend the framework by (Jagannatha and Yu, 2020) who query the sentence-level and the span-level entity s and construct the event of interest set $I(x)$.

In GEC, the users may be more interested in a local probability as a detailed hint. And a token-level probability can offer our study fined-grained information. So we focus on the marginal probability such as $p_\theta(\hat{y}_i|\hat{y}_{<i}, x_i)$ or $p_\theta(\hat{e}_i|x)$. The calibration problem is reduced to that of calibrating a multi-class classifier.

Uniformly, we query the token-level entity $\hat{y}_i$ and the edit-level entity $\hat{e}_i$ as the $\hat{s}_i$ from the sequence-level entity $\hat{s}$ and define the event of interest that is corresponded to the entity $\hat{s}_i$ as $E = \{\hat{s} : \hat{s}_i = s_i\}$. Eg. "The output entity $\hat{y}_i$ in $\hat{y}$ matches the ground truth y_i and is necessarily correct,"; "The output entity $\hat{e}_i$ in $\hat{e}$ matches the ground truth e_i and is necessarily correct,".

We pool together E to construct the event of interest set $I(x)$. Each event E is an elementary event in $I(x)$. We calibrate the local probability $p_\theta(\hat{s} \in E|x)$ to each elementary event E in the event of interest set $I(x)$ instead of their joint probability.

C Annotating Example

Ori.	Hyp.	Gold	Label	Conf.	Cal.
Bodies	Bodies	Bodies	TN	0.982	0.961
can	can	can	TN	0.975	0.974
start	start	start	TN	0.986	0.985
their	their	their	TN	0.983	0.984
days	days	days	TN	0.929	0.951
in	in	in	TN	0.582	0.644
a	a	a	TN	0.475	0.571
test	test	test	TN	0.963	0.957
tubes	tubes	tube	FN	0.634	0.667
.	.	.	TN	0.998	0.998

Table 3: An example of using I-measure to obtain the three-way alignment of the an original sequence, a hypothesis sequence and a gold standard sequence. Labels, local confidences and calibrated confidences for the hypothesis tokens generated at each time step $p_\theta(\hat{y}_t|\hat{y}_{<t}, x)$ are listed.

In a generic framework, the GEC models should be calibrated in the context where they are formulated, therefore we design different procedures for annotating. As for the structured prediction problems, we design an minimum edit distance based procedure (Levenshtein, 1966) to measure the correctness of an edit entity e_i. We denote $D(a, b)$ as edit distance between the sequence a and the sequence b. The insertion cost is 1; the deletion cost is 1; the substitution cost is 2. Going through each edit e_i in the sequence e, we delete e_i for a sequence d, perform e_i on x to get a sequence h.

And we have the gold standard sequence g. We annotate the each entity e_i following the procedure below:

$$\begin{cases} D(x,h) = 0 \begin{cases} D(h,g) \geq D(x',g) : FN, \\ D(h,g) < D(x',g) : TN, \end{cases} \\ D(x,h) \neq 0 \begin{cases} D(h,g) > D(x,g) : FP, \\ D(h,g) < D(x,g) : TP, \\ D(h,h) = D(x,g) : FPN. \end{cases} \end{cases} \tag{9}$$

For a structured transduction problem, (Wang et al., 2020) use the TER metric and tool to annotate the output entity of the sequence transduction models. TER is commonly used in the MT task. TER is the minimum number of edits between the hypothesis and the reference to the average length of the references. Given a hypothesis and a reference, TER should be unique, however, the alignment is not unique (Levenshtein, 1966). (Felice and Briscoe, 2015) propose a three-way dynamic programming algorithm for the alignment problem in the GEC task. We use the I-measure tool by (Felice and Briscoe, 2015) to obtain a token-level annotation with the evaluation labels defined by (Felice and Briscoe, 2015): true negative (TN), true positive (TP), false negative (FN), false positive (FP) or false positive negative (FPN). Table 3 shows an example of using I-measure to annotate the tokens. Please refer to (Felice and Briscoe, 2015) for pseudo code.

D Experimental Details

We study the miscalibration of the two state-of-the-art architectures: the GECToR single models (Omelianchuk et al., 2020) [1] , Transformer-large model (Vaswani et al., 2017). We observe miscalibration of the models provided by the authors on the development dataset (Bryant et al., 2019). We use and train our GEC models following the experimental settings by (Kiyono et al., 2019) and (Omelianchuk et al., 2020). The data for training our models are the BEA-2019 shared task data assigned by (Bryant et al., 2019). We use the officially assigned validation dataset by (Bryant et al., 2019) as the development dataset to select GEC models and as the re-calibration dataset. Our GEC models are evaluated on the BEA-2019 benchmark test dataset (Bryant et al., 2019). We evaluate our calibration methods and report the results on the GEC benchmark dataset CoNLL2014 (Ng et al., 2014). We study the multi-round outputs of the SP models with the best number of rounds suggested by the authors.

[1] https://github.com/grammarly/gector