A Comprehensive Survey of Grammatical Error Correction

YU WANG, YUELIN WANG, KAI DANG, JIE LIU, and ZHUO LIU, Nankai University, China

Grammatical error correction (GEC) is an important application aspect of natural language processing techniques, and GEC system is a kind of very important intelligent system that has long been explored both in academic and industrial communities. The past decade has witnessed significant progress achieved in GEC for the sake of increasing popularity of machine learning and deep learning. However, there is not a survey that untangles the large amount of research works and progress in this field. We present the first survey in GEC for a comprehensive retrospective of the literature in this area. We first give the definition of GEC task and introduce the public datasets and data annotation schema. After that, we discuss six kinds of basic approaches, six commonly applied performance boosting techniques for GEC systems, and three data augmentation methods. Since GEC is typically viewed as a sister task of Machine Translation (MT), we put more emphasis on the statistical machine translation (SMT)-based approaches and neural machine translation (NMT)-based approaches for the sake of their importance. Similarly, some performance-boosting techniques are adapted from MT and are successfully combined with GEC systems for enhancement on the final performance. More importantly, after the introduction of the evaluation in GEC, we make an in-depth analysis based on empirical results in aspects of GEC approaches and GEC systems for a clearer pattern of progress in GEC, where error type analysis and system recapitulation are clearly presented. Finally, we discuss five prospective directions for future GEC researches.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computing methodologies** → *Machine translation;*

Additional Key Words and Phrases: Grammatical error correction, machine translation, natural language processing

ACM Reference format:
Yu Wang, Yuelin Wang, Kai Dang, Jie Liu, and Zhuo Liu. 2021. A Comprehensive Survey of Grammatical Error Correction. *ACM Trans. Intell. Syst. Technol.* 12, 5, Article 65 (December 2021), 51 pages.
https://doi.org/10.1145/3474840

1 INTRODUCTION

Globalization has boosted communication between different languages, and many people have begun to learn a second language, in which grammar is a difficult part of the language acquisition process. Influenced by native language habits, their expressions may be flawed by error patterns dissimilar to those in expressions of native speakers. Besides that, misspelling errors might occur

Yu Wang and Yuelin Wang contributed equally to this research.

This research is supported by the National Natural Science Foundation of China under the grant No. 61976119 and the Natural Science Foundation of Tianjin under the grant No.18ZXZNGX00310.

Authors' address: Y. Wang, Y. Wang, K. Dang, J. Liu (corresponding author), and Z. Liu, Nankai University, Tianjin, China; emails: yuwang17@mail.nankai.edu.cn, yuelinwang17@mail.nankai.edu.cn, dangkai@mail.nankai.edu.cn, jliu@nankai.edu.cn, lzpmbw@gmail.com.

2157-6904/2021/12-ART65 $15.00
https://doi.org/10.1145/3474840

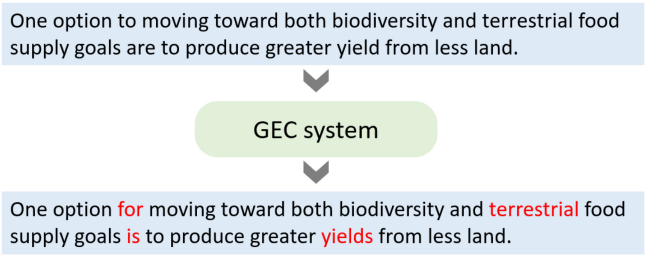

Fig. 1. A typical instance of grammatical error correction.

even among native speakers. In the process of writing articles, essays, news, and emails, typing or grammatical errors can be regarded as unprofessional and may even lead to misunderstandings. Therefore, researches on grammatical correction systems have attracted increasing attention in languages including English [12], Chinese [153], German [7], and so on.

Grammatical error correction (GEC) aims for automatically correcting various types of errors in the given text. Errors that violate rules of target language and expected usage of native speakers in morphological, lexical, syntactic, and semantic forms are all treated as target to be corrected. Early GEC systems focused on correcting grammatical errors in a sentence, i.e., accepting an ungrammatical sentence and returning a grammatically correct one. In a data-driven approach, GEC systems have also been applied to improve fluency, make sentences more native sounding, and conform to the speech habits of native speakers. Although it is ambiguous to make a full definition of grammatical errors, one recent work [13] listed 24 main categories of errors of English that should be considered as "grammatical errors" and should be corrected by GEC system, which have been accepted by GEC researchers and used to evaluate GEC systems' performances on error types. We show the 24 categories of grammatical errors in Table 1 with their annotation codes, explanations, and examples. A typical instance of grammatical error correction is shown in Figure 1.

Grammatical error detection (GED) is a related task to GEC but with some discrepancies. GED is a task of recognizing grammatically incorrect segments in a sentence, whose output are specific locations of grammatical errors, while GEC tries to give corrective suggestions for errors and return a grammatically correct sentence. Contents about GED are out of the scope of this survey, and we recommend to refer to a survey in GED [85] for more details.

The 2000s is the prior stage for GEC. In this stage, most of GEC systems are based on handcraft rules incorporating usage of parsers and linguistic characteristics, for example, the Language Tool [94] and the ESL Assistant [47]. However, the complexity of designing rules and solving conflicts among the rules requires a great magnitude of labor. Although some GEC works today still use rules as an additional source of correction, the performance of rule-based GEC systems has been superseded by data-driven approaches, which are the focus of our survey.

Remarkable progress has been achieved in GEC by data-driven approaches. In late 2000s, classification-based approaches are developed to correct preposition errors and article errors. In this approach, classifiers are trained on large magnitude of native error-free text to predict the correct target word, taking account into the linguistic features given by context. However, this approach is not suitable for correcting all types of grammatical errors and lacks the ability to correct complex sentence errors.

Inspired by **machine translation (MT)**, researchers view the GEC task as translating from the ungrammatical sentences into the correct ones and apply MT methods to GEC. In the early 2010s, **statistical machine translation (SMT)**-based GEC systems gained significant attention and

Table 1. The 24 Main Categories of Grammatical Errors with Annotation Code, Explanation, and Example

Code	Category	Description	Example
ADJ	Adjective	Wrong choice of adjective	big → wide
ADJ:FORM	Adjective Form	Wrong usage of comparative or superlative adjective	better → best, more large → larger
ADV	Adverb	Wrong choice of adverb	swiftly → efficiently
CONJ	Conjunction	Wrong choice of conjunction	but → and
CONTR	Contraction	Wrong usage of contraction	n't → not
DET	Determiner	Wrong choice of determiner	a → the
MORPH	Morphology	Tokens have the same lemma but nothing else in common	quick → quickly
NOUN	Noun	Wrong choice of noun	people → person
NOUN:INFL	Noun Inflection	Count-mass noun errors	waters → water
NOUN:NUM	Noun Number	Wrong usage of noun number	lecture → lectures
NOUN:POSS	Noun Possessive	Wrong usage of noun possessive	friend → friend's
ORTH	Orthography	Case and/or whitespace errors	bestfriend → best friend
OTHER	Other	Errors that do not fall into any other category	-
PART	Participle	Wrong choice of participle	look to → look at
PREP	Preposition	Wrong choice of preposition	below → under
PRON	Pronoun	Wrong usage of pronoun	ours → ourselves
PUNCT	Punctuation	Wrong usage of punctuation	, → .
SPELL	Spell	Misspelling	incetion → inception
VERB	Verb	Wrong choice of verb	push → pull
VERB:FORM	Verb Form	Infinitives, gerunds and participles	to do → doing, enhancing → enhanced
VERB:INFL	Verb Inflection	Wrong usage of tense morphology	writen → written
VERB:SVA	Subject-Verb Agreement	Confliction between subject and verb	he have → he has
VERB:TENSE	Verb Tense	Inflectional and periphrastic tense, modal verbs and passivization	eats → has eaten, eats → can eat, eats → was eaten
WO	Word Order	scrambled word order	better had → had better

drastically surpassed the previous results. More recently, with the increasing popularity of deep learning, **neural machine translation (NMT)**-based GEC systems applying neural sequence-to-sequence models have become dominant and achieve the state-of-the-art performance.

Although GEC and MT models share many similarities, there are still differences among them. GEC has a large overlap between input and output text, which is essentially a partial correction of the sentence, and its metrics also focus on the accuracy of the partial correction. MT has nearly no overlap between input and output and generates sentences from scratch, and its evaluations focus on the semantics and fluency of the sentences. Considering these differences, several researchers have proposed new solutions for GEC. Instead of generating sentences from scratch, they propose methods that predict editing operations from the source sentence to the target sentence, which is inherently a sequence labeling task. Depending on a powerful pre-trained language model, edits-based method reaches comparable results to NMT-based method. Meanwhile, the inference speed of this method is greatly enhanced due to the lack of auto-regressive decoding operations. We illustrate each of the data-driven approaches in Section 3 in detail.

Beyond the six basic approaches, numerous techniques have also attracted significant attention to facilitate the GEC system to achieve better overall performance, especially in SMT- and NMT-based GEC systems. Since GEC is an area that stresses the appropriate application of basic model, the techniques are important to adapt the existing models to GEC. These techniques have been explored and developed incrementally to provide assistance and could be combined

to further improve the error correction ability of GEC systems. In our survey, we describe the researches about GEC techniques and their broad application in existing works.

Data augmentation methods are also of great essence to the development of GEC. The lack of large amounts of public training sentence pairs refrains the development of more powerful MT-based GEC systems. This problem could be partly ameliorated by the proposal of various data-augmentation methods. Most data-augmentation methods generate artificial parallel data for training GEC models, where some inject noise into error-free texts for corruption while others apply back translation to error-free texts to translate them into ungrammatical counterparts. Besides, a small number of data augmentation methods pay attention to a better exploitation of existing training data.

Apart from the aforementioned components of GEC systems, we also cover other aspects of the GEC task. We introduce all the public datasets in GEC and explain some properties of the datasets and their influence and briefly describe the researches about data annotation schema. Besides, standard evaluation provides a platform where multiple GEC systems could be compared and analyzed quantitatively. The evaluation metrics, including both reference-based and reference-less, are explained and compared.

Due to the greatest number of English learners in the world, English grammar error correction has drawn the most attention from researchers, and English GEC systems are also the most refined and efficacious. Two important shared tasks—the CoNLL-2014 Shared Task on Grammatical Error Correction [100] and the BEA-2019 Shared Task on Grammatical Error Correction [12]—significantly promoted English GEC. Therefore, our survey mainly focuses on the progress made in the English GEC systems, while GEC systems for other language will get less involved. Nevertheless, since GEC tasks for different languages mutually share a common structure, the approaches, techniques, and data-augmentation methods we present could be adapted to other languages without significant variation.

Our survey makes the following contributions:

- We present the first comprehensive survey in GEC. To the best of our knowledge, no prior work focuses on the full coverage of datasets, approaches, performance boosting techniques, data-augmentation methods, and evaluation of the great magnitude of researches in GEC, especially the explorations in recent years that yield significant progress.
- We present ample analyses in aspects of approach comparison, approach development, error type, and integrated GEC systems, providing both the panorama and insights to the current GEC progress. Both our qualitative and quantitive analyses contribute to a clearer pattern of existing literature in GEC.
- We explicitly separate the elements belonging to the approaches, performance boosting techniques, and data augmentation and put them together for a more structured description of GEC works. Correspondingly, our analysis to integrated GEC systems is also divided into multiple components for a better linkage between methodologies and experimental results. Due to the nature of GEC, this is more beneficial to following works incorporating disparate approaches, techniques, and data-augmentation methods.
- We propose five prospective directions for GEC in aspects of dataset, approach, technique, and evaluation based on the existing works and analyses.

The rest of this survey is organized as follows: In Section 2, we introduce the public GEC datasets and the annotation schema. Section 3 summarizes the commonly adopted data-driven GEC approaches. We enumerate the performance boosting techniques in Section 4 and summarize data-augmentation methods in Section 5. Evaluation metrics and all the experimental analyses

Table 2. Statistics and Properties of Public GEC Datasets

Corpus	Component	# Sents	# Tokens	# Chars per sent	Sents Changed	# Ref	Edits	Error Type	Proficiency	Topic	L1
NUCLE	Train	57K	1.16M	115	38%	2	minimal	Labeled	Simplex	Simplex	Simplex
	Test	1,312	36.4K								
FCE	Train	28K	455K	74	62%	1	minimal	Labeled	Simplex	Diverse	Diverse
	Dev	2.1K	35K								
	Test	2.7K	42K								
Lang-8	-	1.04M	11.86M	56	42%	1~8	fluency	None	Diverse	Diverse	Diverse
JFLEG	Dev	754	14K	94	86%	4	fluency	None	Diverse	Diverse	Diverse
	Test	747	13K								
W&I	Train	34.3K	628.7K	60	67%	1	-	Labeled	Diverse	Diverse	Diverse
	Dev	3.4K	63.9K	94	69%	1					
	Test	3.5K	62.5K	-	-	5					
LOCNESS	Dev	1K	23.1K	123	52%	1	-	Labeled	Diverse	Diverse	Simplex
	Test	1K	23.1K	-	-	5					

are in Section 6. In Section 7, we propose the prospective directions. We conclude the survey in Section 8.

2 PROBLEM

In this section, we introduce the fundamental concepts of GEC and their notations in the survey, the public datasets, and the annotation schema of datasets.

2.1 Notation and Definition

Since GEC is an area of application of disparate theories and models, we focus only on the universal concepts that are shared among the most GEC systems.

Given a source sentence $x = \{x_1, x_2, \ldots, x_{T_x}\}$ that may contain grammatical errors, a GEC system learns to map x to its corresponding target sentence $y = \{y_1, y_2, \ldots, y_{T_y}\}$ that is error-free. The output of the GEC system is called hypothesis $\hat{y} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{T_{\hat{y}}}\}$.

2.2 Datasets

We introduce all the public datasets in GEC targeting at English, including NUCLE [34], Lang-8 [128], FCE [143], JFLEG [99], and Write&Improve+LOCNESS [12]. Statistics and properties of the datasets are listed in Table 2. Besides, we also describe the GEC datasets on languages other than English.

It is essential to investigate what kind of dataset is more beneficial to GEC. Apart from general attributes, such as sentence number and sentence length, GEC systems can also be influenced by L1 (the first language of writer), proficiency, topics, references, and so on. L1 and proficiency impact how well the system corrects errors for different writers. Specifically, a generic GEC system requires a diverse L1 and proficiency corpus, while conversely a personalized GEC system expects the dataset has more simplex and individualized L1 and proficiency. The number of topics affects the generalizability of GEC system. A higher number of topics naturally brings a more general GEC system, but it may be less obvious in a particular domain. The lower the number of topics, the better the model is in a specific domain, which means generalizability decline at the mean time. The number of references influence whether the system is overfitted. With fewer references, the system is more likely to overfit. A high number of references mitigates the overfitting problem to some extent but introduces new noise and requires more human labor.

2.2.1 NUCLE. The **NUS Corpus of Learner English (NUCLE)** is the first GEC dataset freely available for research purposes. NUCLE consists of 1,414 essays written by Asian undergraduate students at the National University of Singapore. This leads to low diversity of topic, sentence

proficiency, and L1 (the first language of writer) of the data. NUCLE is also annotated with error types. CoNLL-2014 test set, which contains 1,312 English sentences, is the test set that corresponds to NUCLE and the most widely used dataset to benchmark GEC systems.

2.2.2 FCE. The **First Certificate in English Corpus (FCE)** is a portion of proprietary **Cambridge Learner Corpus (CLC)** [31] and is a collection of 1,244 scripts written by English language learners in respond to FCE exam questions. FCE contains broader native languages of writers and more diverse sentence proficiency and topics. Similar to NUCLE, FCE is also annotated with error types. However, FCE is annotated by only one annotator, which may lead to fewer usages than NUCLE.

2.2.3 Lang-8. The Lang-8 Corpus of Learner English (Lang-8) is an English subsection of a social language learning website where essays are posted by language learners and corrected by native speakers. Although Lang-8 contains the maximal amount of original sentences, it is corrected by users with different English proficiency, weakening the quality of the data. Besides, it is not annotated with error types.

2.2.4 JFLEG. The **JHU FLuency-Extended GUG Corpus (JFLEG)** contains 1,511 sentences in GUG development and test set. Unlike NUCLE and FCE, annotation of JFLEG involves not only grammatical error correction but also sentence-level rewrite to make source sentence sound more fluent. Each sentence is annotated four times on Amazon Mechanical Turk. JFLEG provides a new perspective that GEC should make fluency edits to source sentences.

2.2.5 W&I+LOCNESS. **Write&Improve Corpus (W&I)** and LOCNESS Corpus were recently introduced by the BEA-2019 shared task. W&I consists of 3,600 annotated essay submissions from Write&Improve [36], an online web platform that assists non-native English students with their writing. The 3,600 submissions are distributed in training set, development set, and test set with 3,000, 300, and 300, respectively. LOCNESS Corpus is a collection of about 400 essays written by British and American undergraduates, thus it contains only native grammatical errors. LOCNESS contains only development set and test set. All the sentences in W&I and LOCNESS are evenly distributed at each CERF level.

2.2.6 Datasets on Other Languages. Except for the above discussed English GEC datasets, there are also several publicly available datasets targeting at other languages.

- Chinese: NLPCC 2018 Shared Task provides a benchmark dataset for Chinese GEC [154]. This dataset is collected from the Lang-8 website, where essays are written by **CSL (Chinese as a Second Language)** students and corrected by native speaker. The training set and test set contain 1.2 million and 2,000 sentence pairs, respectively.
- German: Boyd built two GEC corpora for German: Falko and MERLIN [8]. These corpora were acquired through Wikipedia edits and filtered using the automatic error annotation tool ERRANT, yielding a total of 24,077 sentence pairs. The training, development, and test set were divided in the ratio of 8:1:1.
- Russian: Rozovskaya and Roth created RULECGEC dataset for Russian GEC [113]. This dataset contains 12,480 sentence pairs from essays and papers written by students learning Russian as a foreign language and heritage speakers.
- Multilingual: Hagiwara and Mita constructed a large-scale multilingual GEC dataset from GitHub [54]. The dataset contains more than 350K edits and 64M characters in more than 15 languages, which cover both spelling errors and grammatical errors.

```
S Surrounded by such concerns , it is very
likely that we are distracted to worry about
these problems .
A 13 14|||Trans|||and|||REQUIRED|||-NONE-|||0
A 11 12|||Vt|||will be|||REQUIRED|||-NONE-|||1
A 12 12|||Wform|||too|||REQUIRED|||-NONE-|||1
```

Fig. 2. An M2 annotation format example. ThLine preceded by *S* represents the original sentence, while line preceded by *A* represents an annotation, which consists of start token offset, end token offset, error type, correction, and some additional information. Note that the last number of each annotation line denotes annotator's ID.

2.3 Annotation

In this section, we talk about the annotation of GEC data, including researches about annotation schema and inter-annotator agreement. Although annotation requires a great magnitude of labor and time, it is of great importance to the development of GEC. Data-driven approaches rely heavily on annotated data. Most importantly, it enables comparable and quantitative evaluation of different GEC systems when they are evaluated on data with standard annotation. For example, systems in the CoNLL-2014 and the BEA-2019 shared tasks are evaluated on data with official annotation, and they are ranked according to scores using identical evaluation metrics respectively. Besides, it also promotes targeted evaluation and strength of GEC systems, since error types may be annotated in data. We can examine the performances of GEC systems on specific error types to identify their strength and weakness. This is meaningful, since a robust specialized system is more desirable than a mediocre general system [13].

The most widely applied annotation is *error-coded*. Error-coded annotation involves identifying (1) the span of grammatical erroneous context, (2) error type, and (3) corresponding correction. Among many error-coded annotation schemas, M2 format is the most commonly used error-coded annotation format today. This is also the annotation format adopted by the CoNLL-2014 shared task and the BEA-2019 shared task. M2 format can merge annotations from multiple annotators, as shown in Figure 2.

Error-coded annotation has some disadvantages. First, different corpora classify error types with significant discrepancy. As mentioned above, error types in FCE are distinguished into 80 categories, while in NUCLE are distinguished into only 27 categories. Besides, annotation may vary with different annotators due to their different linguistic background and proficiency, resulting into low **inter-annotator agreement (IAA)** [109, 129]. *Non-error-coded fluent* edits that treat GEC as whole-sentence fluency boosting rewriting may be more desirable [115].

There are also many researches on inter-annotator agreement and annotation bias. Annotation bias is a tricky problem in GEC that different annotators show bias towards specific error type and multiple corrections. However, annotation bias can be partly overcome by increasing the number of annotators [17, 132]. It is also demonstrated that by taking increasing number of annotation as golden standard, system performance also improves [14].

3 APPROACHES

In this section, we survey the basic data-driven approaches used to tackle GEC problems, including SMT-based approaches, NMT-based approaches, edit-based approaches, classification-based approaches, LM-based approaches and hybrid approaches. We divide this section in several branches to clarify the development of various GEC approaches individually, and each branch includes more details about the representative works.

Since GEC is typically viewed as Machine Translation task, SMT-based approaches and NMT-based approaches are two mainstreams and logically deserve more emphasis. As we have mentioned above, there are discrepancies between the two tasks, so we focus on how the MT approaches are adapted to GEC and the outcomes. The adaptions may occur on approach level and techniques level, and we leave the techniques to Section 4 and the empirical outcomes to Section 6.

3.1 SMT-based Approaches

Before resorting to SMT, grammatical error corrections were mainly achieved by rule-based or classification-based methods, the limitations of which are obvious. On the one hand, designing rules often requires a large amount of prior linguistic information and expert knowledge. The progress of designing rules is time-consuming and the linguistic resources do not appear to be available all the time, especially for some minority languages that are spoken by only a few people. At the same time, an one-for-all rule is almost non-existent, because the natural language itself is so flexible that it makes it difficult for rules to effectively deal with all possible exceptions. On the other hand, the flexibility of natural language also restrains classification-based methods to achieve considerable results under a more extensive scene. The preset labels limit the classification method to certain types of errors and cannot be extended to deal with more complicated error types. For GEC, generally speaking, the errors appearing often involve not only the wrong words themselves, but also the context, which may contain the surrounding tokens in a sentence or even cross-sentence information. Further, the choice of correction is also diverse and flexible, which promotes the development of generative model on GEC. In this section, we present SMT-based models and its development on grammatical error correction. Since most current researchers focus on models based on neural machine translation, we do not discuss too many particulars of models and methods in detail here.

3.1.1 Statistical Machine Translation. Machine translation [75] aims to find a translation y in target language for a given source sentence x in the source language, which probabilistically finds a sentence y that maximize $p(y|x)$. The distribution $p(y|x)$ is the translation model we want to estimate. Using Bayes' formula, we can transform the translation model into a reverse direction, which is known as noisy channel model [10]:

$$\arg\max_y p(y|x) = \arg\max_y p(x|y)p(y). \tag{1}$$

In this model, it is noted that in addition to an inverted translation model, it also includes a language model $p(y)$ on the target side. By adding a language model, the fluency of the output sentence can be scored and improved. These two models are independent, and training a language model does not require parallel corpora. A monolingual corpus with a large amount of target language can obtain a good language model. The introduction of the noise channel model allows us to first train the language model and the translation model separately and then uses the above formula to combine the two models. This idea is better reflected on the log-linear model, which generalizes it with the following equation:

$$\hat{y} = \arg\max_y p(y|x) = \arg\max_y \exp\left(\sum_{m=1}^{M} \lambda_m f_m(x, y)\right). \tag{2}$$

Compared to the noisy channel model, the log-linear model provides a more general framework. This framework contains a variable number of sub-models, namely, the feature functions $f_m(x, y)$, which could be features given by a translation model or a language model, and a group of tunable weight parameters λ_m, which are used to adjust the effect of each sub-model in the translation

process. Also, the total number of feature functions is represented as M. With the log-linear model, prior knowledge and semantic information are possible to be injected by adding feature functions.

Researches that adapt SMT-based methods treat GEC as the translation process from the "bad English" to "good English" [9]. Through training on a large number of parallel corpora, the translation model can collect the corresponding between the grammar error and its correct form, just as the translation process does in mapping bilingual sentences. A range of GEC systems have used statistical machine translation models as the basic framework and taken the advantage of the versatility and flexibility provided by log-linear models with considerable result achieved.

3.1.2 Representative SMT-based Approaches. The pilot study using SMT for grammatical error correction focused on correcting countability errors of mass noun [9]. They employed a dependency treelet-based SMT model and artificially constructed parallel corpus to alleviate the lack of parallel data required by SMT. By testing at the web-crawled CLEC dataset, their model achieved 61.52% accuracy in correcting mass noun errors, which showed the promising application of SMT. This first trial demonstrated that with the assistance of parallel GEC data, more general types of errors could be automatically corrected. Later, the potential competence of SMT-based methods was further explored on unrestricted error types [92, 149] through large-scale parallel data from Lang-8. Also, in the CoNLL-2013 and CoNLL-2014 shared tasks, several SMT-based models played significant roles in those top systems [6, 41, 65, 145] with considerable performance achieved. Among those works, task-specific features and web-scale language models were incorporated into standard SMT models [41, 65] and with appropriate tuning approach, their systems ranked first and third, respectively, on the CoNLL-2014 test set.

A work [126] after those shared tasks also demonstrated the effectiveness of systems utilizing SMT by constructing a hybrid system that combines the outputs of both the SMT components and the classification approaches. In this research, four systems (two SMT-based systems and two classification-based systems, respectively) are combined using MEMT [56], and the final system achieves a state-of-the-art result. Besides, another commonly used feature that is based on neural networks was first integrated in SMT-based GEC system in more recent studies [25], both a **neural network joint model (NNJM)** and a **neural network global lexical model (NNGLM)** are introduced into the SMT-based GEC models. The NNJM is further improved using the regularized adaptive training method described in later work [20] on a higher-quality training dataset that has a higher error per-sentence ratio.

A group of studies [58, 93, 148] paid more attention to n-best list reranking to solve the "first but not the best" problem in the outputs of previous SMT-based GEC systems. An extra rescoring post-processing also enabled more information to be incorporated into the SMT-based methods, through which improvement based on the previous systems has been acquired. Empirical study [112] concerning distinguished approaches has compared the SMT and classification-based approaches by performing error analysis of outputs, which also promoted the idea of a sound pipeline system using classification-based error type-specific components, a context sensitive spelling correction system, punctuation and casing correction systems, and SMT model. Also, system based on SMT only [66] described a baseline GEC system using task-specific features, better language models, and task-specific tuning methods of the SMT system, which outperformed previous GEC models. Later research [21] further incorporated both the NNJM and a character-level SMT for spelling check into the baseline SMT-based model and has shown the efficiency of their methods with an increase in the final performance. As the increasing development of the neural-based methods in both the MT and GEC, the more effective interaction of those methods has been researched [50] based on their comparison between SMT and NMT. Also, hybrid models were proposed in their paper with higher score in test dataset.

3.1.3 Adaption from SMT to GEC. As we have discussed in previous section, the SMT-based GEC approaches employed the log-linear model to translate an incorrect sentence to an error-free sentence and inherited amount of components that were proved efficient in SMT systems. However, compared to the typical MT task that deals with totally different sentence pairs, GEC task particularly asks for minor revision and has a strong requirement for capturing grammar structures. Thus, GEC systems often include several semantic features that could support models to better attend error patterns. In addition, the general framework used in SMT, which incorporates numbers of feature functions, endows SMT-based GEC systems with great flexibility to make adaption by appropriately selecting existing features and introducing task-specific features. In this section, we survey a range of SMT-based GEC systems to investigate the feature selections for GEC task.

Web-scale Language Models. As demonstrated in Equations (1) and (2), a target side language model $p(y)$ is already included in the noisy channel model or the more general log-linear model. However, the size of data when extracting the target side of training sets is so limited that additional web-scale language models are often trained on the large monolingual (local error-free) datasets. The monolingual corpus could be Wikipedia [20, 25, 58, 65, 112, 126], Gigaword [93, 145], and Common Crawl [21, 50, 65, 66]. Such LMs have been trained though tools such as KenLM or IRSTLM in a range of researches to further prompt the improvement of their systems' performance.

Neural Networks as Features. Although the translation model of SMT-based approaches has shown the effectiveness by estimating phrase table, the discrete phrase table and the linear mapping still limit the model to generalize more to pattern beyond the training sets. Also, the global information was always ignored by the basic SMT-based model. Although a range of language models and other sequence modeling features have been incorporated in several SMT-based GEC models, context information concerning each word was lacking. Some works tackled these limitations using complementary neural networks as additional features, namely, **neural network global lexicon model (NNGLM)** and **neural network joint model (NNJM)**, which construct continuous representation space with non-linear mapping modeled [20, 21, 25].

- **NNGLM.** A global lexicon model here is a feed forward neural network used to predict the presence of words in the corrected output by estimating the overall probability of hypothesis given the source sentence. The probability of a target hypothesis is computed using the following equation:

$$\log p(\hat{y}|x) = \sum_{i=1}^{T_{\hat{y}}} \log p(\hat{y}_i|x), \tag{3}$$

  where $p(\hat{y}_i|x)$ is the probability of the target word $\hat{y}_i$ given the source sentence x; $p(\hat{y}_i|x)$ is the output of the neural network.
- **NNJM.** Joint models in translation augment the context information in language models with words from the source sentence. Unlike the global lexicon model, NNJM uses a fixed window from the source side and takes sequence information of words into consideration to estimate the probability of the target word. The probability of the hypothesis h given the source sentence x is estimated by the following equation:

$$\log p(\hat{y}|x) = \sum_{i=1}^{T_{\hat{y}}} \log p(\hat{y}_i|c_i), \tag{4}$$

  where c_i is the context (history) for the target word $\hat{y}_i$. The context c_i consists of a set of source words centralized by word $\hat{y}_i$ and certain number of words preceding $\hat{y}_i$ from the target sentence same as the language model does.

GEC-specific Features. Although the widely used SMT tools Moses [76] had contained dense and sparse features tuning towards BLEU, which was apparently under the machine translation setting, the direct exploitation of those methods seemed to be inappropriate [66]. Early in the CoNLL-2014 shared task, several methods [41, 65] that focused on the task-specific features have been proposed tailored the particularity of grammatical error correction. Systems using character-level Levenshtein distance as dense features [41, 148] and research investigating both specific dense and sparse features [65] all have demonstrated the improvement in the performance of SMT-based approaches. Better interaction of dense features and sparse features were scrutinized in their later research [66], which cultivated a strong baseline SMT model. Here, we survey the commonly used GEC-specific features that may offer an improvement to an SMT-based model in GEC.

- **Levenshtein distance.** Both character-level Levenshtein distance and word-level Levenshtein have been used as dense features. Through the distance, the relation between target and source sentence could be modeled, especially the edit operations, which mainly reflect the correction patterns, are captured.
- **Edit operation counts.** Similar but a more refined and detailed version of Levenshtein distance feature is edit operation counts. Based on the Levenshtein distance matrix, the numbers of deletions, insertions, and substitutions that transform the source phrase into the target phrase are computed; noticeably, the sum of these counts is equal to the original Levenshtein distance.
- **Operation Sequence Model (OSM).** Operation Sequence Model is introduced into Moses for machine translation [37]. These models are Markov translation models that in GEC setting can be interpreted as Markov edition models. Translations between identical words are matches, translations that have different words on source, and target sides are substitutions; insertions and deletions are interpreted in the same way as for SMT.
- **Word-class language model (WCLM).** The injection of word-class information has shown their contribution in the machine translation task early from the IBM series of model. A more general used method used monolingual Wikipedia data to create a 9-gram word-class language model with 200 word classes produced by Word2vec [89]. These features allow to capture possible long distance dependencies and semantical aspects in the SMT-based model.
- **Sparse features.** More fine-grained features can be extracted from the Levenshtein distance matrix as specific error correction operation types with or without context by counting specific edits that are annotated with the source and target tokens that take part in the edit.

A brief summary of SMT based systems in GEC is shown in Table 3.

3.2 NMT-based Approaches

Although SMT-based approach benefits from its ability to incorporate the large amount of parallel data and monolingual data as well as the auxiliary neural network components, it still suffers from the lack of contextual information and limited generalization ability. As a solution, many researches start to research NMT-based approaches for GEC. With the increasing performances obtained by neural encoder-decoder models in machine translation, neural encoder-decoder-based models are adopted. Compared to SMT-based GEC systems, NMT-based models have two advantages. First, neural encoder-decoder model learns the mappings from source to target directly from training parallel data, rather than the required features in SMT to capture the mapping regularities. Second, NMT-based systems are able to correct unseen ungrammatical phrases and sentences more effectively than SMT-based approaches, increasing the generalization ability [146].

Table 3. A Brief Summary of SMT-based Systems in GEC Concerning a Range of Key Properties Widely Employed in Those Approaches

Source	GEC-Specific Features		Reranking		Other Attributes	Year
	Dense	Sparse	Method	Features		
[9]	None	None	None		Dependency Treelet SMT	2006
[92]	None	None	None		None	2012
[149]	None	None	None		Error Generation (EVP), POS Factored SMT, Error Selection	2013
[6]	None	None	None		Hierarchical PB-SMT	2013
[145]	None	None	None		Treelet LM, Classifier	2013
[41]	Char-Level LD	None	Averaging	LM Inside SMT, Web-Scale LM	RB System, Error Generation (EVP), POS Factored SMT, Error Selection	2014
[65]	Word-Level L1	Yes	None		Error Selection	2014
[126]	None	None	None		Classifier	2014
[20]	None	None	None		NNJM	2016
[148]	Char-Level LD	None	SVM	LM, ALM, Length, SMT, Word Lexicon	None	2016
[93]	None	None	Perceptron	POS Tag, Shallow Parse Tag, Combined	None	2016
[58]	None	None	Averaging	SMT, Lexical and POS, LM, Contexts	None	2016
[25]	None	None	None		NNGLM, NNJM	2016
[66]	LD, Edit Counts, OSM, WCLM	Yes	None		None	2016
[21]	Edit Counts, OSM, WCLM	Yes	None		Char-level SMT, NNJM	2017
[50]	Word-level LD, Edit Counts, OSM, WCLM	Yes	None		Char-level SMT, NMT Pipline	2018
[112]	None	None	None		Classifier, Speller	2016

What is important is that, although many NMT-based GEC systems apply the identical architectures with NMT systems, they could never be simply viewed as NMT systems trained with GEC data. First, many GEC systems make GEC-oriented adaptions so the NMT models are now more suitable to GEC, and we will cover the adaption in this section. Second, many GEC systems are combined with multiple performance boosting techniques, as they will be presented in Section 4. In this section, we trace the development of representative works, which are the most commonly used as backbones combining with other techniques in GEC systems today, focusing on works researching the design and training schema of neural models in GEC.

3.2.1 Neural Machine Translation. Unlike SMT consisting of different components, NMT learns a unified neural network that accepts an input sentence and outputs a translation [16, 61, 127]. A neural machine translation system is typically built on encoder-decoder architecture. The encoder encodes the source sentence $x = \{x_1, x_2, \ldots, x_{T_x}\}$ as a vector v. Then, the vector is passed to the

decoder to generate the correction y through

$$p(y) = \prod_{t=1}^{T} p(y_t | y_{1,\ldots,t-1}, v). \tag{5}$$

At each timestep, the target word is predicted with the vector and the previously generated words. Both the encoder and the decoder are RNNs composing of GRU or LSTM units.

Attention mechanism [5] is always applied to improve the output in each decoding step i by selectively focusing on the most relevant context c_i in the source. To be more specific, the hidden state in decoder is calculated by

$$s_i = f(s_{i-1}, y_{i-1}, c_i), \tag{6}$$

where s_{i-1} is the hidden state last step; y_{i-1} is the word generated last step; $f(\cdot)$ is non-linear function. The variable c_i is calculated as

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j, \tag{7}$$

where h_j is the hidden state of encoder at step j and α_{ij} is calculated as

$$\alpha_{ij} = \frac{e^{m_{ij}}}{\sum_{k=1}^{T_x} e^{m_{ik}}}. \tag{8}$$

The variable m_{ik} is a match score between s_{i-1} and h_k.

Convolutional sequence-to-sequence learning [49] is another NMT architecture. It involves an encoder-decoder architecture with multiple layers of convolutions and attention mechanisms. More specifically, in each encoder layer, the source sentence is first embedded using word embedding and position embedding as $S \in \mathbb{R}^{|S| \times h}$, where $|S|$ is the number of tokens in the source sentence, and then linearly transformed into $H^0 \in \mathbb{R}^{|S| \times h}$ before it is fed into the first encoder layer. The output of the lth encoder layer is calculated as follows:

$$H^l = \text{GLU}(\text{Conv}(H^{l-1})) + H^{l-1}, \tag{9}$$

and the output of the final encoder layer H^L is transformed into the output of encoder $E \in \mathbb{R}^{|S| \times d}$. During decoding, the embedding for the generated n target words $T \in \mathbb{R}^{n \times d}$ is calculated the same as S and is linearly transformed into $G^0 \in \mathbb{R}^{n \times h}$. The lth decoder layer computes an intermediate representation:

$$Y^l = \text{GLU}(\text{Conv}(G^{l-1})), \tag{10}$$

which is used for the calculation of attention:

$$Z^l = \text{Lin}(Y^l) + T, \tag{11}$$

$$X^l = (E + S) \times \text{softmax}(E^T \times Z^l), \tag{12}$$

$$C^l = \text{Lin}(X^l). \tag{13}$$

Then, the output of the lth decoder layer is calculated as follows:

$$G^l = Y^l + C^l + G^{l-1}. \tag{14}$$

The matrix G^L is then linearly transformed to $D \in \mathbb{R}^{n \times d}$. The final column of D is then mapped to the size of vocabulary to predict the $(n + 1)$ word.

More recently with the proposal of Transformer [134], many NMT systems replaced traditional RNN-based encoder-decoder architecture with Transformer. Transformer first encodes the source

sentence into a hidden state through a stack of several identical blocks, each consisting of a multi-head self-attention layer and a forward layer. The multi-head self-attention is calculated as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{15}$$

where Q, K, V represent the query matrix, key matrix, and value matrix, respectively, and are calculated by linear-transformation on input vectors. The variable d_k is the dimension of column vectors in Q and K. The decoder has the same architecture as encoder, but with an additional mutual attention layer over the hidden states.

3.2.2 Representative NMT-based Approaches. Yuan and Briscoe applied RNN-based NMT-based models in GEC in 2016 for the first time [147]. Then, Xie et al. applied the first character-level neural encoder-decoder model for GEC [139] to better solve the **OOV (out-of-vocabulary)** problem. Moving forward a small step, combining word-level model and character-level model, a hybrid NMT-based GEC model with nested attention is proposed [63]. This model is designed with more consideration of GEC-specific characteristics. Before long in 2017, Sakaguchi et al. innovatively applied Reinforcement Learning on GEC to better incorporate task-specific metrics and overcome the *exposure bias* led by traditional MLE. Another novel work using the RNN-based NMT model is proposed in 2018, where the model aims to increase the fluency and soundness of the source sentence to correct grammar errors, called fluency boosting learning [48]. This work presented a new perspective to solve GEC using NMT-based approaches.

Besides the widely applied RNN-based NMT models, the first CNN-based NMT model on GEC became the first NMT-based GEC systems outperforming SMT-based GEC systems in 2018 [22]. Following this, another work in 2019 was proposed to integrate cross-sentence context in GEC via a convolutional encoder-decoder model [26]. Considering context, the model is more capable of correcting certain context-dependent errors, such as verb tense error and article error. The strength of CNN-based models over RNN-based ones is that CNN is more efficacious at capturing local context and thereby corrects a wider spectrum of grammatical errors. The long-term context information could be captured by the multi-layer architecture. Besides, only a constant number of non-linearity calculations are performed on the input regardless of the input length, whereas in RNNs, the number of non-linearities is $O(n)$, diminishing the effects of distant words.

With the proposal of Transformer [134], many NMT-based GEC models replaced traditional RNN-based encoder-decoder with Transformer and achieved promising results [19, 51, 67, 69, 83, 96, 142, 150, 152, 155]. Later in 2019, copy mechanism was innovatively applied on GEC task, allowing the model to copy the unchanged source word directly to the target side, and achieved state-of-the-art performance [152]. Compared to RNN and CNN, Transformer has better ability to build long-distance dependency in a sentence through attention, and it enables more efficient parallel computing.

More recently some NMT-based approaches are extended to a multilingual setting. Meta-learning is applied to few-shot GEC domain adaptation to better handle the problem of data scarcity for some languages other than English. The model is initialized with the parameters learned via meta-learning and can quickly adapt to new languages with few training examples, including German, Russian, French, and Mongolian [151]. Multilingual BART [80], which has demonstrated its effectiveness on machine translation task, also receives attention on GEC task on other languages, including German, Czech, and Russian [73].

3.2.3 Adaption from NMT to GEC. Many GEC approaches discussed above inherit their prototypes initially proposed for NMT task. In this section, we describe how these prototypes in NMT

are successfully adapted to GEC. We start from adaptions in model level to learning strategies and minor components in a system.

Copying mechanism. Copying mechanism is initially proposed to facilitate summarization task [52] and semantic parsing task [64]. A copy augmented model learns to copy unchanged source words into the target sentence directly. An NMT model with copying mechanism could copy some tokens (special symbol, named entity, emoji, etc.) from source sentence and enhance the robustness of the translation system [53]. Unlike for NMT, copying mechanism is especially effective for GEC, since only a few edits would be made to source sentences. The final possibility of choosing the next word is composed of two parts, the possibility of generation softmax and the possibility of copying the word from source sentence:

$$p(w) = (1 - \alpha) * p_t^{gen}(w) + \alpha * p_t^{copy}(w). \tag{16}$$

In the representative work [152], α and $p_t^{copy}(w)$ are calculated with copy attention between encoder's output H^{enc} and decoder's hidden state h_t^{dec} at step t:

$$p_t^{copy}(w|x, y_{1\ldots t-1}) = \text{softmax}(K^T q_t), \tag{17}$$

$$\alpha = \text{sigmoid}\left(W \sum (A_t \cdot V)\right). \tag{18}$$

The matrix K, V and the vector q_t are calculated as

$$K, V, q_t = W_k H^{enc}, W_v H^{enc}, W_q h_t^{dec}, \tag{19}$$

and A_t is given by

$$A_t = K^T q_t. \tag{20}$$

Character-level NMT. Generally, a character-level NMT model provides a natural way to overcome the OOV problem for both NMT [30] and GEC [139]. Although with slight difference in the way to get the representation of input sentence from character-level embeddings and in the way to generate words in target sentence, all the models share the same idea that character-level embeddings and attention are beneficial for words representation and generation.

However, unlike NMT, models for GEC should apply only a small number of local edits to the character sequence of a source word, such as correcting misspellings and copying correct words. Thus, the decoder should have direct access to the relevant source character sequence. In the hybrid attention GEC model [63], words that are in target vocabulary are generated by word-level decoder, while those who are out of target vocabulary are generated by character-level decoder. During each decoding step, the probability of each token is the product of probability of **unknown words (UNKs)** calculated by *softmax* function of word-level decoder p, and the probability of the character sequence in a token generated by the character-level decoder. To be more specific, the character-level decoder will select a source word x_{z_s} according to

$$z_s = \underset{k \in 0\ldots T-1}{\arg\max}\ \alpha_{sk}, \tag{21}$$

where α_{sk} is calculated by word-level attention. If the source word x_{z_s} is in source word vocabulary, then the character-level decoder initializes the initial hidden state using $\hat{h_t} = ReLU(\hat{W}[c_s; h_s])$, where h_s is the hidden state of word-level decoder. However, if the source word x_{z_s} is out of source word vocabulary, then a nested attention is applied to make character-level decoder attend to x_{z_s}.

Integrating context. Integrating cross-sentence context is common in the researches about neural machine translation [62, 77, 133], where different sizes of contexts are considered. Inspired by this, Chollampatt et al. designed a cross-sentence convolutional encoder-decoder model with

auxiliary encoder and gating, an extension on their earlier work [22]. In this work, two previous sentences are concatenated and encoded by the auxiliary encoder, and the representation is integrated into decoder via auxiliary attention similar to the attention between source and target.

Reinforcement learning. Reinforcement Learning is commonly adapted to overcome exposure bias led by MLE learning objective in generation tasks. In a Reinforcement Learning setting, the model (*agent*) searches the *policy* $p(h)$ directly according to:

$$\frac{\partial J(\theta)}{\partial \theta} = \alpha \mathbb{E}[\nabla \log(p(\hat{y}))\{r(\hat{y}, y) - b\}]. \tag{22}$$

The variable b is baseline used to reduce the variance. The *reward* $r(\hat{y}, y)$ is calculated by sample outputs $\hat{y}$ and references y, and is specified as different evaluation metrics according to the tasks, for example, BLEU [102] for NMT and GLEU for GEC. The parameters of the model are optimized using policy gradient algorithm.

Incorporating language model. An n-gram language model is trained on large monolingual data and integrated into the beam search decoding phase to rank the hypothesizes with consideration of fluency. This is a commonly applied trick in both NMT and GEC. To be more specific, a hypothesis's score $s(\hat{y})$ at the decoding step k is calculated as follows:

$$s(\hat{y}_{1:k}|x) = \log p_{NN}(\hat{y}_{1:k}|x) + \lambda \log p_{LM}(\hat{y}_{1:k}), \tag{23}$$

where p_{NN} is given by neural translation model and p_{LM} is given by the n-gram language model weighted by λ.

Handling misspellings. Compared to NMT, GEC faces a more serious OOV problem, since it is unrealistic to cover all the misspelled words in the vocabulary due to the limitation of vocabulary size. Several methods are applied to correct misspelling errors in NMT-based GEC systems, some of which have been previously discussed.

- **Alignment+Translation.** The misspelled words can be viewed as UNKs, and the problem could be solved by first aligning the UNKs in the target sentence to their corresponding words in the source sentence with an aligner and then training a word-level translation model to translate those words in a post-processing step [147].
- **Character-level translation model.** As discussed before, some NMT-based models are character-level, so the problem of misspelling errors is naturally solved [63, 139].
- **Spellchecker.** Many NMT-based GEC systems utilize a spellchecker[1] or open-source spell checking library[2] in preprocessing [22, 26, 48, 50, 67, 116].
- **Spell error correction system.** Zhao et al. built a statistical-based spell error correction system and used it for correction of all the misspelling errors in their training data [152].

We recapitulate the GEC systems with NMT-based approaches that we have introduced in Table 4.

3.3 Edit-based Approaches

NMT-based approaches capture the dependencies among the output tokens in an autoregressive mode and achieve optimal results. However, sequential decoding performs slowly in the inference stage with a time complexity of $O(n)$, where n denotes the length of the target sentence. Researchers reconsidered the characteristics of GEC task, the high overlap between the input and output text. In general, the transformation between the source and target sentences can be achieved with a few editing operations. Therefore, researchers proposed edit-based approaches

[1] https://azure.microsoft.com/en-us/services/cognitiveservices/spell-check/.
[2] https://github.com/AbiWord/enchant.

Table 4. A Brief Summary of NMT-based GEC Systems

Source	Framework	Input	Adaption from NMT	Handling Misspelling	Year
[147]	RNN	token level	LM	alignment, word-level translation	2016
[139]	RNN	character level	LM, Character-level model	character-level translation model	2016
[63]	RNN	token level, character level	LM, Character-level model	character-level translation model	2017
[116]	RNN	token level	Reinforcement Learning	spellchecker	2017
[22]	CNN	token level	-	spellchecker	2018
[48]	RNN	token level	LM, Fluency boosting	spellchecker	2018
[50]	RNN	token level	LM	spellchecker	2018
[67]	Transformer	token level	LM	spellchecker	2018
[152]	Transformer	token level	LM, Copying mechanism	spell error correction system	2019
[26]	CNN	token level, sentence level	Integrating context	spellchecker	2019

We split "Handling misspellings" from "Adaption from NMT" to cover more detailed information. "Fluency boosting" could be viewed as an adaption from NMT, and we leave the details to Section 4.4.

for GEC and achieved comparable performance to NMT-based methods. With only a few edits, the speed of inference of this approach is dramatically improved compared to NMT-based approaches.

Edit-based approaches discard the idea of directly predicting the target sentence y and predict a series of editing operations $e = \{e_1, e_2, \ldots, e_{T_x}\}$ based on the source sentence x, such that x can be transformed into the target sentence y after editing sequence e transformations. In general, in the training stage, the method first obtains e sequences according to x and y. Then, the model is trained to learn to predict edits by using x as model input and e as model output. In the inference stage, the model first predicts the edit sequence e based on the input x, after which edit e is applied to x to obtain the final target sequence y.

Researchers have proposed several methods for obtaining edit operations. PIE defines four edit operations, Copy (C), Delete (D), Append (A) with n-gram, Replace (R) with n-gram, and various lexical transformations [4]. The edit sequence is obtained by computing Levenshtein distance between the source and target sentences. Editing operations in LaserTagger consist of two parts, a basic operation and an append phrase P operation [88]. The basic operation is KEEP or DELETE, which indicates whether the output retains the word or not. The append phrase P operation enforces P to be added before the corresponding word, where P belongs to the phrase vocabulary V, which is obtained by combinatorial optimization. They use greedy algorithms to get the final editing sequence. GECToR is edited by manually defining editing operations whose edits include common grammatical errors such as spelling, number of nouns, subject-verb agreement, and verb forms [101]. Its tag vocabulary consists of 4,971 basic transformations (KEEP, DELETE, 1,167 APPEND operations, and 3,802 REPLACE operations) and 29 g-transformations. The edit sequence is also derived by minimizing Levenshtein distance. Building on the previous work, Seq2Edits proposes span-based edits rather than token-level edits, which make the representation easier to learn [124].

Edit-based approaches do little to improve on the model and typically use a transformer model architecture. On the basis of BERT, PIE, and GECToR use a full connection layer and a softmax layer to obtain label prediction results, respectively. LaserTagger adds shallow decoders to the BERT model and explores the respective effect of feedforward decoder and autoregressive

decoder. Inspired by pointer networks, Seq2Edits takes the attention weight over encoder state as probability to predict the source span end position based on Transformer architecture.

3.4 Classification-based Approaches

In classification-based approaches, a multi-class classifier is trained to predict the correct word in the *confusion set*, based on features of words' context from artificial feature engineering or deep learning models. Given the text to correct, for each token in the text occurring in the confusion set, the classifier predicts the most likely candidate. The classifier could be trained on monolingual corpora, thus alleviating the requirement on annotated parallel data. Although classification approaches once were popular, they are not commonly adopted today, so we survey some typical works in general.

Most of the prior classification-based approaches focus on the utilization and improvement of traditional elaborated feature engineering. During this period, approaches mainly focus on identifying or correcting article errors and preposition errors. Han and Leacock trained a maximum entropy classifier to select articles for noun phrases based on 11 features, most of which combine lexical and syntactic information [55]. Chodorow et al. similarly trained a maximum entropy classifier to detect preposition errors by predicting the most probable candidate among 34 prepositions based on 25 contextual features [18]. Based on the previous work, De Felice and Pulman selected some new contextual features and achieved a higher accuracy [35]. Besides, the features for training and a series of filters and threshold are combined with ME approach so the classifier output can be constrained [131]. Continuously, Tetreault et al. successfully added 14 parse features to baseline preposition model, and results showed statistically significant increased accuracy on native speaker test data [130]. Gamon et al. trained two separate decision tree classifiers for article errors and position errors. One for deciding whether or not a determiner/preposition should be present and the other predicting the most likely choice [46]. Gamon combined classifiers with language models and proposed meta-classifier in correcting articles and prepositions [45]. Instead treating all prepositions equally when training classifiers, the confusion set could be restricted on specific several candidates that are frequently observed in occasions of misuse in non-native data, which is called L1-dependent confusion set. Two methods that train classifiers based on L1-dependent confusion set are proposed [110].

Later, some researches extended previous methods so the classification-based approaches can be applied on correcting more types of errors, instead of article errors and preposition errors only. The University of Illinois System trained classifiers to correct noun number errors, subject-agreement errors, and verb form errors. This system ranked first in CoNLL-2013 shared task [107]. Continuously, in CoNLL-2014 shared task, the Illinois-Columbia System extended by incorporating another three error-specific classifiers: word form errors, orthography and punctuation errors, and style errors. They also applied joint inference to address inconsistent corrections suggested by multiple classifiers [108].

More recently, deep learning was combined with classification-based system without the reliance on traditional elaborated feature engineering. Sun et al. first employed CNN to represent the context of source articles and predict the correct labels [125]. Wang presented a classification-based study that trained bidirectional deep RNNs to represent context and predict the correct word directly for each error type involved (article, preposition, verb form, subject agreement) [137]. Following the research above, Kaili et al. proposed two attention mechanisms considering the context words and source word, respectively, to predict correct words better. Li et al. trained bidirectional GRU to correct errors including subject-verb agreement, article, plural or singular noun, verb form, preposition substitution, missing comma, and period comma substitution [81]. They also trained a pointer context model [136] to correct word form error [68].

Makarenkov et al. designed a bidirectional LSTM to assign a distribution over the vocabulary where correction tokens are selected and to suggest proper word substitution [87].

3.5 LM-based Approaches

The SMT-based methods and NMT-based methods are all supervised and thus require large amounts of parallel training data. However, unsupervised approaches based on LM are applied to GEC and achieve comparable performance to supervised methods. The very advantage of LM-based approaches is that they do not need large parallel annotated data, given the fact that large amounts of parallel data is not available in GEC. As a result, LM-based GEC systems are always developed for low-resource situation. Most of LM-based GEC methods use a LM to assess the hypothesis for correction, where hypothesis is generated by making changes to source ungrammatical sentence according to the designed rules or by substituting tokens in source sentence with words selected from *confusion sets*. In this section, we first describe the basic and the optimized LM-based approaches, then give more details about how to generate confusion sets.

N-gram language model can be used to compute a feature score for each hypothesis as follows:

$$score_{LM} = \frac{1}{T_{\hat{y}}} \log p_{LM}(\hat{y}). \tag{24}$$

Hypothesis with the highest score would be added to the search beam and then modified to generate another set of hypotheses. This iteration does not end until the beam is empty or the number of iteration has achieved a threshold [32]. Following this, Bryant and Briscoe re-evaluated LM-based GEC on several benchmark datasets. Language model is used to calculate the normalized log probability of the hypothesis. The process of generating confusion set and evaluating hypothesis is also iterated following previous work. Without annotated data, this work achieves comparable performance with state-of-the-art systems testing on JFLEG test set. This is because the annotation of JFLEG is fluency oriented, thus making language model-based approaches more powerful [11].

A problem of neural LM-based GEC is that the space of possible hypothesis is rather vast. To solve this issue, five finite state transducers are designed to represent large structured search space, constraining the hypothesis space while also promising that it is large enough to involve admirable corrections. The method could also be combined with SMT and NMT models when large amount of annotated data is available [122]. Based on this, Stahlberg and Byrne composed two new transducers for token deletion and insertion. Their LM-based approaches achieved higher score on CoNLL-2014 test set [123].

Confusion set is an important component in LM-based GEC approaches, since language model selects the most possible hypotheses from it. Dahlmeier and Ng generated hypothesis by (1) replacing misspelled word with correction; (2) changing observed article before Noun Phrase; (3) changing the observed preposition; (4) inserting punctuation; (5) altering noun's number form (singular of plural) [32]. Similarly, Bryant and Briscoe created confusion set by (1) applying spell checker CyHunspell on misspelled words; (2) altering morphological forms of words using automatically generated inflection database; (3) changing the observed article and preposition [11]. Besides, WikEd Error Corpus was also used to create a part of the confusion set [42].

Flachs et al. combined pre-trained language models such as BERT and GPT-2 with noisy channel model for GEC. For each word w in a given sentence, the noisy channel model estimates the probability that w is transformed from a candidate c in the confusion set. The idea behind this approach is that for any given word w, there is a genuine word c that passes through the noisy channel and transforms into w. The goal is to choose the most possible genuine word c^*:

$$c^* = \arg\max_{c \in C} P(c|w). \tag{25}$$

According to Bayes' principle, equation could be written as

$$c^* = \arg\max_{c \in C} P(w|c) * P(c), \tag{26}$$

where $P(w|c)$ is estimated by noisy channel model; $P(c)$ is given by pre-trained language model (BERT, GPT-2), and C is the generated confusion set for w [42].

3.6 Hybrid Approaches

Apart from models we have discussed, a range of researches also integrated several models into their systems to seize profit from different models to obtain better performance. Such systems with sub-models will be surveyed in this section as hybrid methods, which often contain heterogeneous sub-components. It is worth noticing that ensemble methods (Section 4.3) also involve content concerning model combination, which is similar to this section. However, in this survey, we restrict the ensemble methods to the techniques of making better combination of outputs produced by a range of models. The major difference between hybrid and ensemble in this survey is that the hybrid methods integrate several sub-systems to construct a complete correction process, whereas ensemble methods use several independent whole system, often combined at the output level, to improve the correction performance.

Yoshimoto et al. and Felice et al. employed different sub-components in their system to make corrections to different types of error and merge them to produce an error-free sentence. More specifically, Yoshimoto et al. used three systems to deal with all five error types in the CoNLL-2013 shared task, including a system based on the Treelet language model for verb form and subject verb agreement errors, a classifier trained on both learner and native corpora for noun number errors, and an SMT-based model for preposition and determiner errors [145]. Felice et al. combined both a rule-based system and an SMT-based system. Their outputs are combined to produce all possible candidates without overlapping correction and then a language model is trained to rerank those candidates [41].

Pipeline methods are also researched in combining sub-systems, where the output corrected by one system is passed as an input to a following correcting system. Rozovskaya and Roth first applied a classification model followed with an SMT-based system, since the SMT-system owns the ability to handle with more complex situation than classifiers [112]. A similar framework is proposed by combining the more powerful neural classifiers and SMT-based system [68]. Grundkiewicz and Junczys-Dowmunt constructed an SMT-NMT pipeline and experiments in the research have shown complementary corrections have been made. They also explore using the NMT system to rescore the n-best hypothesis obtained through SMT system to improve fluency of final outputs [50].

3.7 Comparison among Approaches

We now present the comparison of different approaches from several key properties that distinguish among GEC approaches. Those properties could help us better characterizing those models' frameworks as well as their output performances; thus, we could figure out the superiority and limitation of each approaches. We majorly consider properties as previous empirical work [112] does, but extend their conclusion to all types of models in GEC:

- **Error coverage** denotes the ability of a system to identify and correct a variety of error types.
- **Error complexity** indicates the capacity of a system to address complex mistakes such as those where multiple errors interact.

Table 5. Comparison among Different Approaches

Property	SMT	NMT	Edit-based	LM	Classification
Error Coverage	G; fully covered	G; fully covered	F; few abandoned	B; some covered	B; some covered
Error Complexity	F; error without complex dependency	G; complex error	G; complex error	B; local error	B; local error
Generalization Ability	B; restrained by phrase table	G; through continuous representation	F; restrained by edit operations	G; through large scale training data	G; through specific classifiers
Supervision/ Annotation	B; require large scale parallel data	B; require large scale parallel data	F; require small scale parallel data	G; require only unlabeled data	G; require only error labels
System Flexibility	F; could done by adding new features	B; hard to make specific modification	F; could done by adjusting edits	B; hard to inject new characteristic	G; easy by integrating new classifiers
Explainability	B; inexplicable	B; inexplicable	G; explainable	B; inexplicable	G; explainable

"G," "F," and "B" represent a relatively Good, Fair, and Bad property, respectively.

- **Generalize ability** refers to the ability of a system to identify mistakes in new unseen contexts and propose corrections beyond those observed in training data.
- **The role of supervision** or having annotated learner data for training.
- **System flexibility** is a property of the system that allows it to adapt resources specially to correct various phenomena.
- **Explainability** refers to whether the model could explain its correction operation.

The first three items reflect the property of system output while the other three items characterize more aspects of system frameworks. All conclusions of analyzing those four types of approaches have been demonstrated in Table 5, while more detailed information will be covered below. The comparison among different approaches is visualized in Figure 3.

We first employ error coverage to understand how systems differ in capturing patterns of various error types and its corresponding corrections. The classification-based methods and LM-based methods are limited in covering more error categories due to the classifiers or confusion sets that often tailor with specific error type. A typical classification system mainly deals with error types that have certain number of possible correction operations, such as article errors and preposition errors. On the contrary, almost all types of errors are well-covered by MT-based methods (both SMT and NMT), in which the error patterns are automatically captured by training on the error-complete parallel corpus. Similarly, the edit-based approaches could acquire edits for each errors after training with a well-designed edit operation set. It is also worth noticing that an MT-based system or an edit-based system may also perform poor in certain error type (hold a low recall score) even if it is trained on an error-complete dataset. Error types that appear in few samples are hard to capture for those MT-based systems and edit-based systems. Also, the operation set designed in edit-based models may abandon low frequent corrections to reduce complexity, thus may lead to a performance decline on related error types.

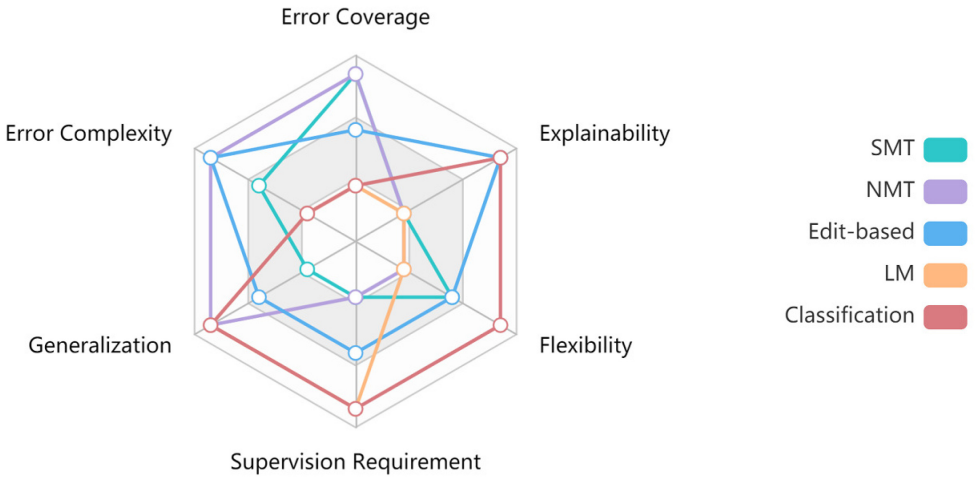

Fig. 3. Comparison among different approaches.

To handle with the situation that an error type involves complex correction as well as interaction among multiple error types, different methods employ distinct strategies. In SMT-based approaches, all the information of errors and its corresponding correction is implicated inside the phrase-table obtained from parallel datasets, which also limits their correction pattern can only be done inside a phrase while context as well as other global information beyond a single phrase is often ignored. In NMT-based approaches and edit-based approaches, context information is often considered and the links among multiple tokens are capable to build. Although long dependency problem still remains when RNN or LSTM is used, model with more flexible links such as convolutional sequence-to-sequence model and Transformer have provided better solutions. Those NMT models outperform previous SMT-based methods partially because of the benefit of shorter link path among tokens. However, LM and classification-based methods can hardly achieve an equal performance in dealing such complexity with those generative models due to the limitation of both classifiers and confusion sets.

All the correction produced by MT systems (both NMT- and SMT-based methods) are derived from patterns appeared in parallel corpus. The learned rules or grammatical information they obtained is inducted from the surface form of the error-correction pairs, which also leads to the difficulty of generalization. In SMT methods, the mined phrase-table almost places such a restriction on the correction that only correction they have met could be produced. The LM-based approach also confronted with the similar circumstance as the phrase table did in SMT-based methods due to the mapping constructed by confusion sets. The neural methods partially handled this problem by employing a continuous representation space and their approximation property when learning a pattern, nevertheless limitation still remains when we require further generalization. Although the classification cannot well cover all type of errors, it still demonstrates its competent of generalization in dealing with certain error types. For categories such as prepositions or articles, several more general rules are obtained.

Another property is whether the model allows for a flexible architecture where a range of error-specific knowledge could be incorporated by tailoring with additional resources. The fact that classification methods exploiting various classifiers suited to individual error types has demonstrated the flexibility of those methods by adjusting or augmenting with specific classifier. Although the MT-based methods appear to have a more fixed structure where a whole model is used to solve all error types, several additional techniques have been proposed to better incorporate extra resources

based on the standard MT methods. For SMT, often a range of features are integrated in log linear model, where such architecture also allows incorporation of extra information as sparse features such as syntactic patterns and correction patterns for certain error type. Whereas in NMT-based methods, extra information is more difficult to be injected, although methods such as multi-task learning could be used directly against several specific phenomena. Differing from NMT-based approaches, edit-based approaches are more flexible to make adaption to specific error patterns by adjusting preset edit operations, where modifications could impact models' preference.

Better explainability of a model could not only assistant people to understand the detailed structure of solutions, but promote fine-grained applications also. In GEC task, a model with better explainability could explain the reason for a certain correction operations. In other words, a preferred GEC system should predict the corresponding error type for each error appeared in the original sentence. The classification approaches employ a range of classifiers for different error types so we could recognize each correction as long as we check which classifier it comes from. The edit-based approaches complete the corrections while making prediction to the label of edits that hold a certain kind of correspondence with error type list. Therefore, the labeling sequence could readily formulate the error type of some corrections, where the explainability depends on the design of edit operations.

4 PERFORMANCE-BOOSTING TECHNIQUES

While various approaches are successfully applied to GEC, a wide range of performance-boosting techniques beyond basic approaches are also proposed and developed to facilitate the GEC systems to achieve better performance. In this section, we divide the typical techniques into several branches and describe their application in detail. We intentionally separate the techniques in this section and the basic approaches in Section 3 to distinguish the improvement brought by techniques, although most techniques are combined with NMT-based GEC approaches. In this way, basic GEC models could incorporate multiple techniques for more powerful GEC systems. Data augmentation strategies can be also viewed as performance-boosting techniques, however, due to the large amount of researches and the broad application of them, we describe data-augmentation methods in Section 5 individually.

It is worthy to mention that some training tricks including dropping out word embeddings of source words [121], checkpoint average [103], **byte pair encoding (BPE)** [118], and domain adaptation [67] are commonly used in developing GEC systems. However, these tricks are not initially proposed or adapted to improve the performance of GEC systems. Since numerous performance-boosting techniques are applied and combined with each other, it is unrealistic to give all of them a detailed description. So, we pay attention to techniques that are more specific to GEC, especially researches that are intended to conduct methodological exploration.

4.1 Pre-training

Sequence transfer learning has shown huge improvements on the performance of many NLP tasks, since a pre-trained model often obtained more reasonable initial parameters than using random initialization. Such pre-trained models can improve the convergence speed and training efficiency, leveraging knowledge acquired from related data while reducing the need for high-quality data for tasks where sufficient corpora are not available. Since GEC is always treated as low-resource machine translation task [67], various pre-training approaches are applied on GEC to improve the performance of models. The main discrepancies among these pre-training approaches are (1) whether the pre-training uses GEC related data and (2) what part of the parameters of the whole model is initialized with pre-trained model.

4.1.1 Pre-training with Artificial Parallel Data. Due to the lack of high-quality data, many studies on GEC focused on the efficient usage of pseudo parallel training data, most defining a pre-training task to introduce the information of error patterns comprised in GEC artificial parallel data. An NMT-based GEC system typically adopted this method to pre-train its neural seq2seq model as a **denoising autoencoder (DAE)** [135]. Given the original sentence y and its noised counterpart $\hat{y}$, the training of autoencoder aims at minimizing the distance between y and $\hat{y}$. During the pre-training, the model learns to reconstruct the input sentence and thus is capable of making corrections. The parameters of both encoder and decoder are pre-trained.

The pre-trained model could be used for the following training procedure with two strategies: *re-training* and *fine-tuning*. The main difference between re-training and fine-tuning is that the learning rate and parameters of optimizer are reset in re-training strategy, while they are maintained in fine-tuning scenario, although some work may fine-tune the pre-trained model with a smaller learning rate.

Many Transformer-based GEC systems adopted pre-training on artificial parallel GEC data [15, 19, 51, 70, 74, 141, 152]. We cover more details about artificial data generation in Section 5. Besides the synthetic parallel data, *Wikipedia* revision history could also be a source for pre-training [83].

4.1.2 Pre-training with Large Monolingual Data. Pre-training on large error-free monolingual data can also boost the final performance by leveraging large general corpora. In neural encoder-decoder models, pre-training the decoder as a language model on large monolingual could largely benefit the final performance, since the architecture of decoder of sequence learning model is the same as language model [105]. Unlike the pre-training method described in last section, only the parameters of decoder are pre-trained, and the parameters of encoder and attention mechanism are moved away during pre-training. Many NMT-based GEC systems benefit from this technique with a few discrepancies in the source of monolingual data [23, 67, 81, 152].

Besides the NMT-based systems, the parallel edit model [4] can also be pre-trained using large amounts of error-free sentences in a way much like the training of BERT, predicting the arbitrary masked token combining both forward context and backward context.

Recently, a novel work focused on the appropriate incorporation of BERT into GEC [70]. The representation of the source sentence from BERT is used as additional features in the NMT-based GEC model. Furthermore, better result was gained by fine-tuning BERT on GED task to take into account grammatical errors.

4.2 Reranking

Compared to other performance-boosting techniques that are often integrated inside the models, reranking, as a kind of post-processing, is a more independent component employed after the whole correction process. Often n-best outputs (as candidates or hypothesis) with the highest probability given by the correction model are rescored during the reranking and the optimal candidate will be selected according to the new scores as the final output. The aim of n-best list reranking, in GEC, is to rerank the correction candidates produced by the previous components using a rich set of features that are not well-covered before, so better candidates can be selected as "optimal" corrections. By appending a reranking component, (1) linguistic information and features tailored to GEC could readily be introduced into correction system; (2) outputs from different grammatical error correction systems can be incorporated; (3) some global features could be exploited without the decoding processing being modified. Empirical studies showed the deficiency of systems that the best hypothesis may not be the optimal correction [93] and a range of researches have employed reranking, demonstrating that there is considerable room for improvement brought by

the reranking component. We summarize the commonly adopted features and how to incorporate them, respectively.

4.2.1 Features.

- **Language model.** An n-gram language model is trained on large monolingual corpora to give a score $p_{LM}(\hat{y})$ to each hypothesis. The score is calculated by summing all the n-gram log possibilities together and then normalizing it by the length of hypothesis. Many systems train a 5-gram language model [22, 69, 140]. Besides, the *masked language model* probabilities computed by BERT can also be used for reranking [26]. For each token $\hat{y}_i$ in the hypothesis $\hat{y}$, they replace $\hat{y}_i$ with [MASK] token and calculate the probability of the masked token. This feature score is calculated as

$$f_{BERT}(\hat{y}) = \sum_{i=1}^{T_{\hat{y}}} \log p_{BERT}(\hat{y}_i|\hat{y}_{-i}). \tag{27}$$

- **Sentence-level correctness.** A neural sequence error detection model is trained to rerank the n-best hypotheses output by MT-based model. The model assigns a probability $p(\hat{y}_i)$ to each token, indicating the likelihood that the token is correct. Given a candidate in the n-best list, the probability of each token being correct is $\sum_{i=1}^{T_{\hat{y}}} \log p(\hat{y}_i)$. The neural error detection model can be the combination of two bidirectional LSTM-RNNs to encode both the character and the context of the token [144, 150] or trained with auxiliary context predicting tasks[144], where the loss function is modified into the following:

$$E = -\sum_{t=1}^{T} \log p(y_t|x_t \dots x_T) - \gamma \sum_{t=1}^{T-1} \log p(x_{t+1}|x_1 \dots x_t) - \gamma \sum_{t=2}^{T} \log p(x_{t-1}|x_t \dots x_T). \tag{28}$$

  The sentence-level correctness score could also be predicted by BERT, which is fine-tuned on learner corpus [69].
- **Edit operation.** Three features relating edit operation are always combined with other features, including the numbers of token-level substitutions, deletions, and insertions between source and hypothesis [22, 69]. This feature could be replaced with a similar *Levenshtein Distance* feature [144].
- **Right-to-left model.** Inspired by the application of right-to-left model in machine translation [118], some GEC systems rerank hypothesis using the scores of right-to-left model to better incorporate the context of each word [51, 70, 74].
- **Neural quality estimation.** Chollampatt and Ng proposed the first *neural quality estimation* model for GEC and used the quality estimation score as a feature in reranking, which yields statistically significant improvement on base GEC model. In this work, the neural quality estimation model is composed of a predictor and an estimator. The predictor is an attention-based multi-layer CNN model, trained on parallel source and target sentences, to predict the probability of the tokens in target sentence given the source sentence and the context of target token. The estimator is also CNN-based, trained on source sentences and system hypothesises, predicting the quality estimation score of hypothesises. The golden standard score given by GEC evaluation metric M^2 is used as label in the training [23].
- **Syntactic features.** Several global syntactic features could be injected into reranking process. A range of researches use lexical features that the words occurring in the source side and target side of an edit and their **parts-of-speech (POS)** as features. The lexical features can determine the choice and order of words, and the POS features can determine the

grammatical roles of words in the edit within a hypothesis [58, 148]. Other sequential syntactical features extract from structures like dependency parse tree were also researched [93].

- **Translation model score.** Reusing the score from translation model can partially preserve the correction information in the reranking process. Such features extracted from the correction models such as decoding scores and n-best list ranking information (represented linearly or non-linearly) could be incorporated for rescoring [148].
- **Adaptive language model.** Adaptive LM scores are calculated from the n-best list's n-gram probabilities. N-gram counts are collected using the entries in the n-best list for each source sentence. N-grams occurring more often than others in the n-best list get higher scores, ameliorating incorrect lexical choices and word order. The n-gram probability for a hypothesis word $\hat{y}_i$ given its history $\hat{y}_{i-n+1}^{i-1}$ is defined as:

$$p_{n-best}\left(\hat{y}_i|\hat{y}_{i-n+1}^{i-1}\right) = \frac{count_{n-best}\left(\hat{y}_i, \hat{y}_{i-n+1}^{i-1}\right)}{count_{n-best}\left(\hat{y}_{i-n+1}^{i-1}\right)}. \tag{29}$$

  The sentence score for the sth candidate $\hat{y}^s$ is calculated as:

$$score(\hat{y}^s) = \frac{1}{T_{\hat{y}}} \log\left(\prod p_{n-best}\left(\hat{y}_i|\hat{y}_{i-n+1}^{i-1}\right)\right), \tag{30}$$

  where it is normalized by the sentence length [148].
- **Length feature set.** Length features are often used to penalize overhaul in correction process. Unnecessary deletions or insertion will be limited by introducing those length ratios:

$$score(\hat{y}^s, s) = \frac{N(\hat{y}^s)}{N(s)}, \tag{31}$$

$$score(\hat{y}^s, \hat{y}^1) = \frac{N(\hat{y}^s)}{N(\hat{y}^1)}, \tag{32}$$

$$score(\hat{y}^s, \hat{y}^{max}) = \frac{N(\hat{y}^s)}{N(\hat{y}^{max})}, \tag{33}$$

$$score(\hat{y}^s, \hat{y}^{min}) = \frac{N(\hat{y}^s)}{N(\hat{y}^{min})}, \tag{34}$$

  where $\hat{y}^s$ is the sth candidate; $\hat{y}^1$ is the 1-best candidate (the candidate ranked 1st by the correction system); $N(\cdot)$ is the function calculating the sentence's length, thus $N(\hat{y}^{max})$ returns the maximum candidate length in the n-best list for that source sentence and $N(\hat{y}^{min})$ returns the minimum candidate length [148].

4.2.2 Incorporation. The most commonly used incorporation method is calculating the weighted sum of the scores given by multiple feature functions and selecting the hypothesis with the maximum total score, as expressed in the following equation:

$$\hat{y}^* = \arg\max_{\hat{y}} \sum_{i=1}^{m} \lambda_i f_i(\hat{y}), \tag{35}$$

where m is the number of feature functions; λ_i is the weight for the ith feature; $f_i(\cdot)$ is the feature function; $\hat{y}$ is the hypothesis being reranked.

Also, several studies trained a discriminative ranking model to score the n-best list derived from the preset features extracted from those hypotheses [93, 148]. Such incorporation of the features closely correlates their ranking performance with the ranking model they train. Weighting

feature functions differently as the parameter tuning does in the previous methods, the ranking models serve the same purpose of obtaining a learnable model to better consider different feature functions' score according to the performance.

4.3 Model Ensemble

Although model ensemble does not improve the performance of individual model or algorithm in essence, combining different systems has the potential to improve both recall and precision. Recall could be increased when systems focusing different aspects of corrections are well-integrated so errors could be identified much more comprehensively. Precision could be increased by utilizing the fact that correction produced by multiple systems will give us more confidence about its correctness. Also, we should notice that, besides GEC, various ensemble methods have been proposed and exploited over different tasks in NLP. A range of more sophisticated methods have been successfully used in fields such as **Named Entity Recognition (NER)** and **Entity linking (EL)**. In this section, we investigate a common way to combine models, which achieves the ensemble during decoding, and several empirical studies introducing more complex ensemble methods tailored to GEC.

4.3.1 Better Search in Decoding. A widely used ensemble method utilizes different systems when searching the n-best list during decoding. Traditionally, a range of models (homogeneous or heterogeneous) are trained and fine-tuned under distinct settings separately and then combined during the decoding process, where the prediction score of each model is averaged. Model ensemble is widely used in the development of GEC systems together with beam search to obtain a more reliable n-best list. For example, an ensemble of four Transformer-based machine translation models brings higher scores on the BEA-2019 test set [123, 142]. Similar systems also made use of an ensemble of four multi-layer CNN models [22]. A more ensembled systems includes 3 neural GEC models, each consisting 4 identical multi-layer CNNs, while disparate training strategies and techniques, including label smoothing and source word dropout, are applied on training the three sets of models [23]. Since there are so many systems based on this simple ensemble pattern, and it is not restricted in GEC, we cover no more information about instances and details.

4.3.2 Recombining the Edits. Although many GEC systems regard the correction task as a special machine translation problem, a significant difference between GEC and MT is that commonly the output of an MT system is produced and evaluated as a whole, whereas an output derived from GEC systems could be seen as a combination of edits or corrections that can be evaluated separately. The pilot implementation of this idea was done by a system taking advantage of outputs from both a classification and an SMT approach [126]. In this work, pairwise alignments are first constructed using MEMT forming a confusion network and then the one-best hypothesis is produced during a beam search progress. Similar to MEMT, they employ features including language model, matches, backof,f and length features to rescore both partial and full hypothesis. Their methods allow switching among all component systems, flexibly combining their outputs.

More recently, there have been some successful GEC systems that use more sophisticated ensemble strategies to combine multiple corrections extracted from outputs produced by different systems based on the fact aforementioned. Li et al. presented an ensemble method to integrate the output of four CNN-based ensemble GEC models and eight Transformer-based GEC models and solved possible conflicts. They first built up a confidence table for each individual system, consisting of the precision and $F_{0.5}$ of each error type given by ERRANT toolkit. Then, they designed three rules to merge the corrections output by different systems using the precision as the confidence of each correction, increasing the confidence of identical corrections while discarding the conflicting

correction with lower confidence. Three types of model ensemble with different combination of CNN-based and Transformer-based models are investigated [81]. Kantor et al. proposed an ensemble method to combine different GEC systems at a higher scale, treating each individual system as a black-box. Their method merged multiple M2 format files of disparate GEC systems by splitting the M2 format files and identifying the probability that an edit of an error type is observed in a subset of edits. An optimization problem should be solved to determine the optimal probability for each error type [71].

4.4 Iterative Correction

Iterative correction in GEC aims at correcting source sentences not in a single round decoding as traditional sequence generation, but instead in multiple round inference. The generated hypothesis is not regarded as the final output, but is fed into the model to be edited in following iteration. This makes sense, because some sentences with multiple grammatical errors may not be corrected in only one decoding round, which requires higher reference ability of the model. The inspiration is from iterative beam search decoder [86], which makes correction on ungrammatical sentence through multiple rounds. In each round, the decoder searches the hypothesis space to select the best correction for the source sentence. The selected sentence is input to decoder as source sentence in the next iteration and is incrementally corrected. Some language model-based approaches rely on this algorithm [11, 32].

Fluency-oriented iterative refinement [48] corrects source sentence through multiple round inference until the fluency of hypothesis does not increase. The fluency is defined as follows:

$$f(x) = \frac{1}{1 + H(x)}, \tag{36}$$

$$H(x) = -\frac{\sum_{i=1}^{T_x} \log p(x_i|x_{<i})}{T_x}. \tag{37}$$

The variable $p(x_i|x_{<i})$ is given by a language model.

Some works also proposed an iterative decoding algorithm and successfully applied it to the model trained on out-of-domain Wikipedia data. In each iteration, an identity cost and every non-identity cost of hypothesis in the beam are calculated, and the hypothesis with minimal non-identity cost is maintained. If the minimal non-identity cost is less than the product of an identity cost and a predetermined threshold, this hypothesis is viewed as a correction and input for next iteration. The model continues to correct the input sentence until no more edit is required [82–84, 96].

Other tricks that could be viewed as iterative correction include feeding the output into the ensemble translation model for a second pass correction [81], letting the SMT-based model and classifiers take turns to make correction until no more correction is required [68], and parallel iterative edit model [4], which takes the output of the model as the input for further refinement until the output is identical to a previous hypothesis or the iteration number achieves the maximum round.

4.5 Auxiliary GED

Some GEC systems incorporate a GED task to help boost the performance, since the capability of identifying grammatically incorrect sentences is beneficial for a better error correction outcome. GED is typically incorporated into GEC systems by two methods: pipeline and multi-task learning.

4.5.1 Pipeline. In a pipeline GED-GEC system, a sentence is first input to a GED system and then with error detection results sent to a GEC system. This could further be divided into two levels: sentence level and token level.

Sentence level. A GED module first conducts a binary classification task to tell whether the input sentence contains grammatical errors, and a GEC system could reduce the number of false positives by only considering sentences classified as grammatical incorrect ones in GED module [90]. The sentence-level GED task could also be combined with sentence proficiency level prediction task [2]. A BERT is trained to predict both the binary GED label (whether correction is required) and the three proficiency-level labels simultaneously, and fine-tuned on each set of sentences with the same proficiency level.

Token level. Token-level GED is treated as a sequence labeling task. The most recent research on token-level GED-assisted GEC [15] achieves comparable performance to the state-of-the-art performance with a significant reduction in inference time. The system consists of two parts: **erroneous span detection (ESD)** and **erroneous span correction (ESC)**. The detection model is a sequence tagging model to identify the text spans that are grammatically incorrect in the source sentence, while the correction model is a sequence-to-sequence model but only outputs the corrected texts for annotated spans.

4.5.2 Multi-task Learning. Multi-task learning allows systems to use information from GED task via joint learning, leading to performance improvement on GEC. Similarly, both sentence-level and token-level GED could be trained together with GEC task, where sentence-level GED is a binary classification task and token-level GED is a sequence tagging task. Some NMT-based systems applied this technique [150, 152]

4.6 Edit-weighted Optimization

Another performance-boosting technique is to directly modify the loss function to increase the weight of edited tokens between source and target [23, 67, 96, 142, 152]. To be more specific, suppose that each target token y_j could be aligned to a corresponding source token x_i: $a_t \in \{0, 1, \ldots, T_x\}$. If y_j differs from x_i, then the loss for y_j is enhanced by a factor Λ. The modified loss is as follows:

$$L(x, y, a) = -\sum_{t=1}^{T_y} \lambda(x_{a_t}, y_t) \log p(y_t | x, y_{<t}), \tag{38}$$

$$\lambda(x_{a_t}, y_t) = \begin{cases} \Lambda, & x_{a_t} \neq y_t, \\ 1, & \text{otherwise.} \end{cases} \tag{39}$$

5 DATA AUGMENTATION

Data augmentation has always been explored in GEC, since supervised models suffer from the lack of parallel data and low quality. Unlike machine translation, a large magnitude of parallel training data in GEC is not currently available. Thus, a wide spectrum of data-augmentation methods were studied and applied to ameliorate the problem. Data could be augmented by two channels: enlarging the amount of training data, commonly known as **artificial error generation (AEG)**, and better exploiting the existing training data. Most importantly, the artificially generated data should capture commonly observed grammatical errors and imitate them in the pseudo data. In this section, we survey some typical data-augmentation methods, most of which could be divided into three categories: *noise injection*, *back translation,* and *better exploitation of existing data*.

It is worthy to note that the artificially generated data can be typically used in two approaches. First, append the generated data to existing training data and use the combined data for training. In this approach, the pseudo data is viewed *equivalent* as real training data. Second, use the generated data for *pre-training* the neural model, and then use the real training data for fine-tuning. We have discussed pre-training in Section 4.1. It is demonstrated that when used for *pre-training*, larger amounts of pseudo data yield better performance, while *equivalent* setting does not show

significant improvement in final result [74]. As a result, many neural-based GEC methods adopt pre-training to incorporate a large magnitude of artificial data.

5.1 Noise Injection

5.1.1 Deterministic Methods. Among the data-augmentation methods that corrupt error-free data by adding noise or applying noising function, some methods directly inject noise according to predefined rules, while others inject noise with consideration of more linguistic features. We first summarize direct noise injection methods, which are also called *deterministic* methods.

Izumi et al. first used predefined rules to create a corpus for grammatical error detection by replacing preposition in original sentences with alternatives [60]. Brockett et al. used predefined rules to alter quantifiers, generate plurals, and insert redundant determiners [9]. Lee and Seneff defined rules to change verb forms to create an artificial corpus [79]. Ehsan and Faili defined error templates to inject artificial errors to treebank sentences when original sentences matched the error templates [38]. More recently, similar to direct noising methods applied in low-resource machine translation [78], Zhao et al. generated artificial data using corruption, including (1) insertion of a random token with a probability, (2) deletion of a random token with a probability, (3) replacing a token with another token randomly picked from the dictionary probabilistically, (4) shuffling the tokens [152]. Lichtarge et al. applied character-level deletion, insertion, replacement, and transposition to create misspelling errors [83]. Similarly, Yang and Wang corrupted One Billion Word Benchmark corpus by deleting a token, adding a token, and replacing a token with the equal probability [142].

5.1.2 Probabilistic Methods. Although deterministic noise injection methods are effective to some extent, the generated errors such as random word order shuffling and word replacing are less realistic than errors observed in GEC datasets. Many approaches inject noise to original data with consideration of more linguistic features, thus the generated errors resemble more to real errors made by English learners. We call these approaches *probabilistic* methods.

There are some typical works in prior stage. GenRRate [44] is an error generation tool with more consideration of POS, morphology, and context. Rozovskaya and Roth proposed three methods to generate artificial article errors with more consideration of frequency information and error distribution statistics of ESL corpora: (1) injecting articles so their distribution of generated data resemble distribution of articles in ESL corpora before annotator's correction; (2) injecting articles so their distribution of generated data resemble distribution of articles in ESL corpora after annotator's correction; and (3) injecting articles according to specific condition probability. $P(token_{src}|token_{trg})$ is estimated from annotated ESL corpora, which means the probability that article $token_{src}$ should be corrected into article $token_{trg}$. During error generation, article $token_{trg}$ in error-free context is replaced with $token_{src}$ with probability $P(token_{src}|token_{trg})$ [111]. Following this, inflation method [114] is proposed to solve the problem of error sparsity and low error rate in artificial data. This noise injection approach has been widely applied by following GEC models and systems, especially classification-based [107, 108]. Yuan and Felice extracted two types of correction pattern from NUCLE corpus: context tokens and POS tags, and injected the patterns to EVP corpus with equal probability to generate artificial training data [149].

More recently, Felice and Yuan extended previous work by injecting errors with more linguistic information, and their approach generated five types of errors. Specifically, they refined the conditional probability $P(token_{src}|token_{trg})$ in the following several aspects: (1) estimating the probability of each error type and using it to change relevant instances in error-free context; (2) estimating the conditional probability of words in specific classes for different morphological context; (3) estimating the conditional probability of source words when POS of target words are assigned;

(4) estimating the conditional probability of source words when semantic classes of target words are assigned; (5) estimating the conditional probability of source words when the particular sense of target words are assigned. Experimental results validated the effectiveness of their consideration of various linguistic characteristics [40]. Based on this, the pattern extraction method [106] is an improvement and can be applied to generate errors for all types. In this method, correction patterns *(ungrammatical phrase, grammatical phrase)* are extracted from parallel sentences in training corpus, and errors are injected by looking for matches between error-free context and *grammatical phrase*. Xu et al. generated five types of errors, including concatenation, misspelling, substitution, deletion, and transposition by considering the relation between sentence length and error numbers and assigned a possibility distribution to generate each error type [141]. They designed an algorithm to create a *word tree* that was used to create more likely substitution errors. For Chinese GEC task, Zhao and Wang proposed five different noise strategies, including padding substitution, random substitution, word frequency substitution, homophone substitution, and mixed substitution [155]. By applying these strategies to the original source sentences dynamically in the training procedure, more diverse instances of error-corrected sentence pairs are generated to improve the performance of the model.

Other noise injection methods were also proposed and applied in recent GEC researches. Aspell spellchecker was applied to create confusion set to generate word replacement errors more accurately than randomly selecting a substitute from word list [51]. Choe et al. generated artificial errors basing on inspection of frequent edits and error types in annotated GEC corpora [19]. Besides, Kantor et al. generated synthetic data via applying the corrections observed in the training corpus W&I backward to native error-free data [71]. What is more, some SMT-based systems benefit from cheat sheet [104], a dictionary of error-focused phrases, which consists of two parts: (1) errors and their corresponding context that are directly extracted from existing training data and (2) mapping phrases with their translation probability over 95% from the SMT phrase table, which has a more random distribution than the prior over error location in a phrase.

The Wikipedia revision history record could also be used for data augmentation [83]. The parallel sentence pairs are extracted by aligning the texts from consecutive snapshots downsampled from individual Wikipedia pages, comparing the matching segments and filtering some pairs. Different downsampling strategies and filtering settings are combined to achieve better pre-training outcomes.

5.2 Back Translation

Back translation was initially proposed to augment training data for neural machine translation [117] and then adapted to GEC. In this kind of AEG method, a reverse machine translation model is trained, translating the error-free grammatical data into ungrammatical data.

Rei et al. first trained a phrase-based SMT model on the public FCE dataset to generate errors. Trained models are further applied on EVP dataset to extract generated parallel data. They also made attempt on other monolingual (error-free) data such as Wikipedia dataset but failed to have equal-quality data, which demonstrated by their experiments demonstrated that keeping the writing style and vocabulary close to the target domain gives better results than simply including more data [106].

Kasewa et al. first applied an attention-based neural encoder-decoder model on error generation [72]. Three different error generation methods were discussed and compared. (1) ***Argmax* (AM)** selects the most likely word at each decoding timestep according to each candidate token's generation probability p_i. (2) ***Temperature Sampling* (TS)** involves a temperature parameter τ to alter the distribution:

$$\hat{p}_i = \frac{p_i^{1/\tau}}{\sum_j p_j^{1/\tau}}, \tag{40}$$

which controls the diversity of generated tokens. (3) ***Beam Search*** **(BS)** maintains n best hypothesis each timestep according to their scores. Experimental results show that *Beam Search* error generation method improves the performance of error detection model trained on the generated parallel data most significantly.

However, the traditional beam search in decoder would result in far fewer errors in generated ungrammatical sentences than original noisy texts [140]. As a solution, the beam search noising schemes can be extended through (1) penalizing each hypothesis $\hat{y}$ in the beam by adding $k\beta_{rank}$ to their scores $s(\hat{y})$, where k is their rank in descending log-probabilistic $p(\hat{y})$ and β_{rank} is a hyperparameter (2) penalizing only the top hypothesis h_{top} of the beam by adding β_{top} to $s(h_{top})$ and (3) penalizing each hypothesis in the beam by adding $r\beta_{random}$ to their scores $s(\hat{y})$, where r is a random variable between 0 and 1. As a result, more diversity and noise were involved in synthesized sentences.

Quality control [150] was applied to guarantee the generated sentences are less grammatical. To be more specific, the generated ungrammatical sentence $\tilde{y}$ can be added to training set only when $\tilde{y}$ and the error-free original sentence y satisfy the following inequation:

$$\frac{f(y)}{f(\tilde{y})} \leq \sigma, \tag{41}$$

where

$$f(y) = \frac{\sum_{t=1}^{T_y} \log p(y_t|y_{<t})}{T_y}. \tag{42}$$

The variable σ is learned on the development set with size $|N|$:

$$\sigma = \frac{\sum_{x,y}^{N} \frac{f(y)}{f(x)}}{N}. \tag{43}$$

Htut and Tetreault compared several neural translation models that could be used on artificial error generation: multi-layer CNN, Transformer, PRPN [119], and ON-LSTM [120]. Experiment results showed that ungrammatical sentences generated by multi-layer CNN and Transformer were more beneficial to training GEC systems. However, adding too much artificial data would impact the performance [59]. So, it is important to oversample the authentic training data to maintain the balance [123]. Besides the mentioned works, many other GEC systems also incorporate monolingual data via back translation [48, 70, 74].

5.3 Better Exploitation of Existing Data

Except for the numerous researches on enlarging the amount of training data, two recent works pay attention to how to better exploit the existing annotated training datasets. We summarize them below.

Data weighted training [82] assigns a score for each training example measuring its "noise," a technique adapted from NMT [138]. The score is specified as δppl, describing the change in a model's log perplexity for an example between two checkpoints in model training:

$$\delta ppl(x, y; \theta^-, \theta^+) = \log p(y|x; \theta^-) - \log p(y|x; \theta^+), \tag{44}$$

where θ^- refers to model parameters pre-trained on a base dataset D^- (artificially generated data) and θ^+ refers to model parameters fine-tuned on a high-quality dataset. Examples from D^- that are similar to examples from D^+ can be expected to have lower δppl, while those that are different

from those of D^+ should have higher δppl. The lower an example's score, the higher weight is given to the example in the calculation of training loss.

Noise Reduction [90] is another recent work focusing on denoising training datasets. The non-negligible noise in annotated training datasets includes inappropriate corrections and uncorrected errors. It is logical to denoise GEC datasets and then obtain a better GEC model trained on the denoised datasets. Instead of simply filtering the noisy examples, a self-refinement model is trained on noisy datasets to refine noisy target sentences, and then a new GEC model could be trained on the denoised data.

6 EXPERIMENT

We have introduced public datasets, basic approaches, performance-boosting techniques, and data-augmentation methods in GEC separately. It is time to summarize the experiment results of the introduced elements when they are applied in GEC systems and present the analysis based on the experiment results. In this section, we first introduce the evaluation metrics in GEC and then summarize the experimental results of different GEC approaches in the view of their evolution. More importantly, we present error type analyses based on the BEA-2019 participants. After that, we recapitulate a wide range of GEC systems whose approaches, performance-boosting techniques, and data-augmentation methods have been introduced in this survey for a clearer pattern of existing GEC works.

6.1 Evaluation

Appropriate evaluation of GEC has long been a hot issue [28, 86, 91, 132], due to the subjectivity, complexity, and subtlety of GEC [97]. Generally, evaluation methods in GEC can be divided into two branches: reference-based and reference-less. The difference is whether reference is required during evaluation. Since reference-based methods are more commonly applied, we concentrate mainly on this type of metrics.

6.1.1 Reference-based Metrics. Reference-based evaluation metrics are computed by comparing the hypotheses with references or golden standards. Since traditional precision, recall, and F-score can be misleading [17], some evaluation metrics have been proposed, including M^2 [33], I [39], GLEU [97], and ERRANT [13]. When multiple references are available, M^2, I, and ERRANT choose the one that maximizes the metric score, while GLEU randomly selects one reference and calculates the average of 500 scores as the final score.

Some works compared these metrics in relation to human judgments in both system-level and sentence-level [12, 24]. System-level correlation coefficient is calculated by comparing the ranking of the systems by human evaluation and the ranking generated by the metric scores, while sentence-level correlation coefficient is computed by examining the number of pairwise comparisons that metrics agree or disagree with humans. The first is system-level Pearson (r):

$$r = \frac{\sum_{i=1}^{q}(a_i - \overline{a})(b_i - \overline{b})}{\sqrt{\sum_{i=1}^{q}(a_i - \overline{a})^2}\sqrt{\sum_{i=1}^{q}(b_i - \overline{b})^2}}, \tag{45}$$

where a and b are system scores given by metric and human, respectively. The variable q is the number of GEC systems. The second is system-level Spearman (ρ):

$$\rho = 1 - \frac{6\sum_{i=1}^{q}(d_i)^2}{q(q^2-1)}, \tag{46}$$

Table 6. Different Evaluation Metrics

Metric	Definition	Multiple References	System		Sentence
			Pearson r	Spearman ρ	Kendall τ
M^2	Eq. 48∼50	max	0.623	0.687	0.617
I	Eq. 52∼53	max	−0.25	−0.385	0.564
GLEU	Eq. 54∼56	average	0.691	0.407	0.567
ERRANT	Eq. 48∼50	max	0.64	0.626	0.623

where d is the difference between metric rank and human rank. The third is sentence-level Kendall's Tau (τ):

$$\tau = \frac{|Concordant| - |Discordant|}{Total\ \#\ Pairwise\ Comparisons}, \tag{47}$$

where $|Concordant|$ and $|Discordant|$ are the numbers of pairwise comparisons that metric agrees and disagrees with human. Comparison of different metrics is listed in Table 6. However, following work argues that comparing metrics in relation to human judgments is costly as well as suffers from low inter-rater agreement. Instead, an automatic methodology MAEGE[27] for metric validation is proposed leveraging existing annotation in GEC.

M^2. *MaxMatch* (M^2) is the most commonly used evaluation metric in GEC today. M^2 relies on error-coded annotations of gold standard. Phrase-level edits of system hypotheses are extracted by M^2 with the maximum overlap with the gold standard. The edits are evaluated with respect to gold edits in F_β measure. Suppose the extracted set of gold edits for sentence i is $\boldsymbol{g_i}$, while the set of system hypothesis edits for sentence i is $\boldsymbol{e_i}$. Recall R, precision P, and F_β are defined as follows:

$$R = \frac{\sum_{i=1}^{n} |\boldsymbol{g_i} \cap \boldsymbol{e_i}|}{\sum_{i=1}^{n} |\boldsymbol{g_i}|}, \tag{48}$$

$$P = \frac{\sum_{i=1}^{n} |\boldsymbol{g_i} \cap \boldsymbol{e_i}|}{\sum_{i=1}^{n} |\boldsymbol{e_i}|}, \tag{49}$$

$$F_\beta = \frac{(1+\beta^2) \times R \times P}{R + \beta^2 \times P}, \tag{50}$$

where

$$\boldsymbol{g_i} \cap \boldsymbol{e_i} = \{e \in \boldsymbol{e_i} | \exists g \in \boldsymbol{g_i}, match(g, e)\}, \tag{51}$$

where $match(g, e)$ means that e and g have the same offset and correction. $F_{0.5}$ emphasizes precision twice as much as recall, and precision is more important than recall in GEC. This is because it is more desirable that edits of system hypothesis are actually grammar errors than some of the edits are erroneous correction.

I. M^2 has a series of disadvantages. First, it does not discriminate do-nothing systems and erroneous correction proposing systems, since their $F_{0.5}$ will all be 0. Second, phrase-level edits may not always reflect effective improvements, thus misleading evaluation outcomes. Besides, error detection scores are not computed. Aiming at addressing the disadvantages, *Improvement* measure, I, is proposed. Based on the alignment among the source sentence, hypothesis, and gold standard, I is computed using weighted accuracy where TPs and FPs are weighted higher than TNs and FNs. Specifically, given the aligned tokens w^{src}, w^{hyp}, and w^{std} in source, hypothesis, and gold standard, TP, FP, TN, FN are defined as follows:

- TP: $w^{src} \neq w^{std}$ while $w^{hyp} = w^{std}$;
- FP: $w^{src} \neq w^{hyp}$ while $w^{hyp} \neq w^{std}$;
- TN: $w^{src} = w^{hyp} = w^{std}$;
- FN: $w^{src} \neq w^{std}$ while $w^{hyp} \neq w^{std}$.

The weighted accuracy is calculated as follows:

$$WAcc = \frac{\lambda \cdot TP + TN}{\lambda \cdot TP + TN + \lambda \cdot (FP - \frac{FPN}{2}) + (FN - \frac{FPN}{2})}, \tag{52}$$

where FPN are cases that can be classified as both FP and FN. λ is set to 2 to give TPs and FPs higher weights. The metric I is calculated according to:

$$I = \begin{cases} \lfloor WAcc \rfloor, & if\ WAcc = WAcc_{src}, \\ \frac{WAcc - WAcc_{src}}{1 - WAcc_{src}}, & if\ WAcc > WAcc_{src}, \\ \frac{WAcc}{WAcc_{src}} - 1, & otherwise, \end{cases} \tag{53}$$

where $WAcc_{src}$ is calculated by treating source sentences as hypotheses as well.

GLEU. Both M^2 and I require explicit error annotations. Inspired by machine translation, GEC can be treated as sequence-to-sequence rewriting and propose **Generalized Language Evaluation Understanding metric (GLEU)**. GLEU calculates weighted precision of n-grams of hypothesis over references, which rewards correctly changed n-grams while penalizing n-grams that appear in the source sentence but not in the references. In the experiment, GLEU has the strongest correlation degree with human evaluation. To calculate GLEU score, n-gram precision is first calculated as follows:

$$p_k = \frac{\sum_{\hat{y}^i \in \hat{Y}} [\sum_{n \in N} \#_{\hat{y}^i, y^i}\ n - \sum_{n \in N} B(\#_{\hat{y}^i, x^i}\ n - \#_{\hat{y}^i, y^i}\ n)]}{\sum_{\hat{y}^i \in H} \sum_{n \in N} \#_{\hat{y}^i}\ n}, \tag{54}$$

where $\hat{Y}$ is a set of corrected hypotheses, $B(x)$ is a bigger number between 0 and x, $\underset{a}{\#}\ n$ is a number of n-gram sequences in a, and $\underset{a,b}{\#}\ n$ is the minimal number of n-gram sequences in a and b. $N = \{1, 2, 3, 4\}$. The brevity penalty is computed according to

$$BP = \begin{cases} 1, & if\ l_h > l_r, \\ \exp(1 - l_r/l_h), & if\ l_h \leq l_r, \end{cases} \tag{55}$$

where l_r is the token number of the references and l_h is the token number of all hypotheses. Then, GLEU is finally calculated as

$$GLEU(S, \hat{Y}, R) = BP \cdot \exp\left(\frac{1}{N} \sum_{k=1}^{N} \log p_k\right), \tag{56}$$

where S and R are sets of source sentences and references, respectively.

ERRANT. ERRANT is an improved version of M^2 scorer. It first extracts the edits, then classifies the errors into 25 categories. ERRANT F_β is calculated as same as M^2 F_β. While ERRANT and M^2 both evaluate span-based correction, ERRANT also reports the performance on span-based detection and token-based detection. Besides, it is the first metric that is capable of evaluating the performance on different error types, which is more beneficial for development of GEC systems.

6.1.2 Reference-less Metrics. Reference-less evaluation is not researched until recently. The motivation is that reference-based metrics always penalize edits that are not included in reference, resulting into unfair underestimating of GEC systems. Several reference-less evaluation metrics are researched. The first trial is the research about LFM score [57] and error count score given by **e-rater (ER)** and **Language Tool (LT)** [98]. Then, adding fluency and meaning preservation into consideration demonstrates that reference-less metrics can replace reference-based metrics in evaluating GEC systems [3]. Besides, a statistical model is trained to scale the grammaticality of a sentence using features including misspelled counts, n-gram language model scores, parser outputs, and features extracted from precision grammar parser [57]. Finally, based on UCCA scheme [1],

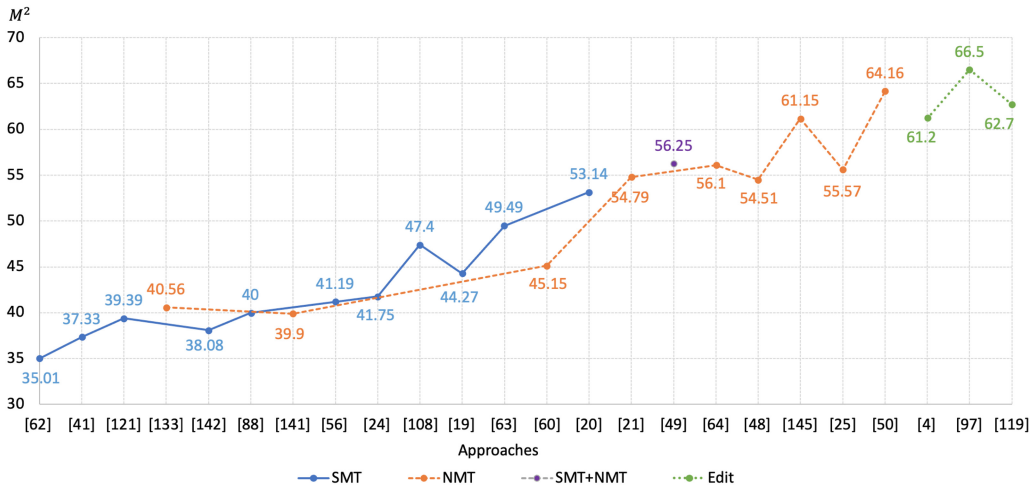

Fig. 4. Development of SMT-based approaches, NMT-based approaches, and edit-based approaches.

US_{IM} is a reference-less measure of semantic faithfulness that compares the differences between the UCCA structures of outputs and source sentences according to alignable but different tokens. Comparing to RBMs requiring many references, US_{IM} has a lower cost in GEC systems. It also shows sensitivity for changes in meaning [29].

6.2 Development of Approaches

We summarize the experimental results of SMT-based systems and NMT-based systems that we have introduced in Section 3 in Figure 4. The systems are evaluated on CoNLL-2014 test set with M^2 metric. The experiment results of the selected systems mainly reflect the performance of the approaches themselves, although some may be combined with other performance-boosting techniques. The systems are arranged in chronological order. It is obvious that incremental performance is achieved in both SMT-based approaches and NMT-based approaches.

From the increasing performance of a series of systems shown in the graph, we could draw a brief conclusion from those broken lines in Figure 4 with regard to several turning points and preference switches among different approaches. Before the CoNLL-2014 tasks, classification-based and rule-based approaches are widely used in GEC, which are not involved in Figure 4 in advance of all works we described in the lines graph. Although SMT was proposed early in 2006 [9] with considerable performance in a certain error type, further researches stagnated due to the lack of parallel data until those shared tasks together with the proposal of NUCLE and utilizing web-crawled language learner data in GEC [92, 149]. Systems exploiting SMT-based approaches [41, 65] obtained 35.01 and 37.33, respectively, in CoNLL-2014 shared task as shown in the graph, seizing the first and third place among all systems. Later work [126] that combined both SMT and classification in 2014 obtained the state-of-the-art performance with 39.39 $F_{0.5}$ score, which demonstrated that SMT-based methods had achieved the equal or even more significant position compared to classifiers in the field of GEC. More researches employing SMT in 2016 developed models that are better tailored to GEC task, and a strong baseline was created entirely with SMT [66], which reached a new state-of-the-art result, 49.49. This work also symbolized the peak of SMT-based systems in GEC then. Early in the same year, several pilot works introduced NMT-based methods, utilizing attention augmented RNN into GEC task [139, 147]. Although those NMT-based methods were never on trial in GEC before, they still showed promising results in GEC task with 40.56 and 39.9 $F_{0.5}$

scores. A later study combined both word-level and character-level attention in their NMT model, obtaining 45.15 $F_{0.5}$ score [63]. Under the influence of the empirical performance of neural methods, systems based on SMT further modified their models with additional neural features to remedy the defect of SMT methods that are unable to capture context information or make continuous representations. Indeed, a range of study [20, 21, 25] demonstrated that neural network features such as NNJM did create extra improvement to standard SMT model in GEC. More recently, NMT-based model that employed convolutional sequence-to-sequence model holding an $F_{0.5}$ score of 54.79 outperformed pre-dominated SMT approaches [22], in which dependency that correction process required could be more ready to capture than RNN by constructing shorter link path among tokens. The developed techniques, including copy mechanism and better pre-training [152], and optimized network structures better released the potential of neural methods on incorporating context and prompted the NMT-based approaches to a new state-of-the-art. Following a hybrid system [50] that combined NMT and SMT also proved the state and efficiency of NMT-based approaches in GEC. Then transformer was introduced into GEC and produced better performance [67], after which most of GEC systems preferred transformer as a baseline model. Recently, a range of researches have constructed their models based on edit operations in a grammatical error correction process. Their edit-based models benefit from smaller output space compared with generation models, thus they often call for less decoding time and parallel data to train. Although this type of model, as a brand new method for GEC, already has achieved comparative performance with NMT-based approaches, it requires further exploration to cultivate it into a well-rounded approach for GEC.

6.3 Error Type Analysis

In this section, we present analyses about system performance on different error types mentioned in Table 1. Although not so many GEC works focus on error type analysis, this is instrumental for a better understanding of the complexity of GEC task, the strengths of current systems, and further challenges in GEC. Our analysis of error type is based on submissions to the BEA-2019 shared task for two reasons. First, the BEA-2019 shared task is the most recent shared task on GEC, where the participants exploit most up-to-date approaches, techniques, data-augmentation methods, and training data, representing the research progresses achieved in GEC. Second, the BEA-2019 official evaluation includes the detailed results on error types for each participant, providing the source of error type analysis. This is especially important, given the fact that not so many GEC works present error type evaluation performance and analysis.

The average ERRANT $F_{0.5}$ scores of all the participants in BEA-2019 are shown in Figure 5. The column "All" means the overall performance on all error types. The names of other columns are error codes, each representing an error type. The painted red columns represent the six error types where GEC systems hold the worst-performing $F_{0.5}$ results, which include **adjectives (ADJ), adverbs (ADV), conjunctions (CONJ), nouns (NOUN), other (OTHER), verbs (VERB)**, and **word order (WO)** errors. These categories of errors often involve word choice errors where models should deeply understand the content of a sentence instead of fairly capture morphological patterns in grammar structure. The results indicate that most existing GEC systems are unsatisfactory to obtain semantics information to facility correction operations, especially in the case of data deficiency. Besides, the break lines in the figure represent two systems that are notable for a better score on those six types of errors, which refer to UEDIN-MS and Kakao&Brain, respectively. Both of those systems investigate more proficient ways to generate synthetic parallel data. Their ways of data augmentation not only cover the morphological errors, but also involve content-aware edits. Their models may be altered to learn to better attend the content of sentences after pre-training on those data.

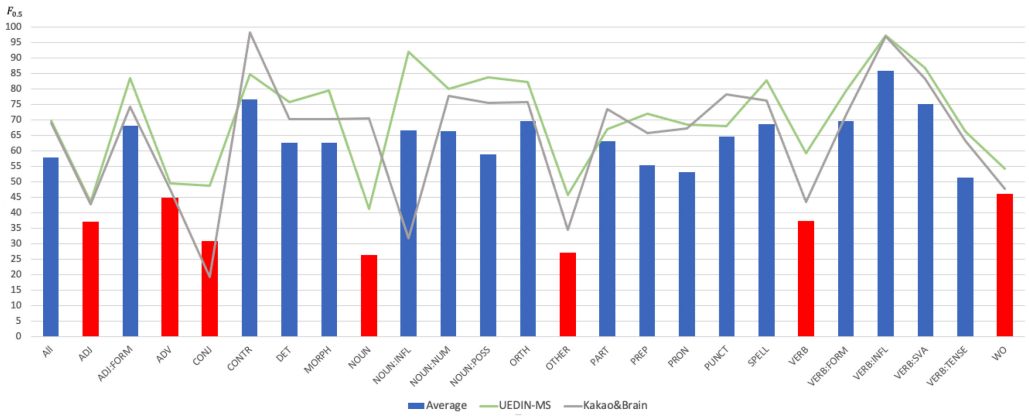

Fig. 5. The average ERRANT $F_{0.5}$ scores of all the participants in BEA-2019 shared task on overall performance and on each error type.

6.4 Analysis of Integrated GEC Systems

Now, we present empirical analysis in the perspective that a GEC system is an integration of multiple contributing components. We aim to provide a panorama of GEC, including both recent top-performing systems and a comprehensive coverage of different branches. Each system is summarized in aspects of which basic approaches they are based on, which techniques and data-augmentation methods they used, and which datasets are used for training. Each aspect corresponds to a subsection in the survey, and we also divide our analysis into several subsections accordingly.

Table 7 recapitulates typical GEC systems after CoNLL-2014 plus top three ranking systems in CoNLL-2014, including works that present initial trial on new approaches, researches that explore new beneficial techniques and data-augmentation methods, and submissions to CoNLL-2014 and BEA-2019 shared tasks comprising different components based on existing works. Note that for CoNLL-2014 shared task, we include only three top-ranking systems, since the performances of most systems in CoNLL-2014 are relatively limited compared to subsequent works, weakening their value for reference. For BEA-2019 task, a system might participate into multiple tracks, so we include the evaluation results on restricted track, where the available annotated training data is the same as other systems (i.e., the FCE, Lang-8, NUCLE, and W&I+LOCNESS) and the unlimited monolingual data is also the same as other systems, for a more controlled analysis. Systems are evaluated on at least one of the three test sets with their corresponding evaluation metric: CoNLL-2014 test set with $M^2F_{0.5}$, BEA-2019 test set with ERRANT $F_{0.5}$, and JFLEG with GLEU. We include the best score and their corresponding configuration of approaches, techniques, and data-augmentation methods. Systems before CoNLL-2014 shared task are not included, since they are not evaluated on any one of the three test sets. Systems are ordered in primary consideration of their published year. For systems in the same year, we rank them according to their performance on CoNLL-2014 test set first, then BEA-2019 test set, and at last JFLEG. This is because CoNLL-2014 test set is the most widely used benchmark in today's GEC research community, whereas BEA-2019 test set and JFLEG are not adopted as frequently. All systems are evaluated on BEA-2019 test set using ERRANT metric, except two systems [71, 74], are submissions to the BEA-2019 shared task, among which some also report M^2 on CoNLL-2014 test set while others do not. Besides, no system is evaluated on JFLEG before it was released.

Table 7. A Panorama of Integrated GEC Systems

Source	Year	Approach			Performance Boosting Technique				Data Augmentation			Training Data				Evaluation		
		S	N	Others	P	R	E	Others	N	B	E	N	F	L	W	CoNLL-2014 M^2	BEA-2019 ERRANT	JFLEG GLEU
[82]	2020		√		√		√	I			√		√	√	√	66.8	73	64.9
[101]	2020			E	√		√		√			√	√	√	√	66.5	73.6	-
[70]	2020		√		√	√				√		√	√	√	√	65.2	69.8	62
[90]	2020		√		√	√	√	G			√			√	√	63.1	67.8	63.7
[124]	2020			E	√	√	√						√	√	√	62.7	70.5	64.3
[15]	2020		√		√			G				√	√	√	√	61	66.9	-
[74]	2019		√		√	√	√		√	√					√	65.0	70.2	61.4
[51]	2019		√		√	√	√		√			√	√	√	√	64.16	69.47	61.16
[141]	2019		√		√		√		√				√		√	63.2	66.61	62.6
[4]	2019			E	√		√	I				√	√	√		61.2	-	61
[152]	2019		√		√	√	√	I, G	√			√	√	√		61.15	-	61
[84]	2019		√				√	I						√		60.4	-	63.3
[19]	2019		√		√	√	√		√			√	√	√	√	60.33	69	-
[123]	2019		√			√				√					√	58.9	63.72	-
[122]	2019	√	√	L		√	√					√		√		58.4	-	55.6
[26]	2019		√		√	√	√					√		√		55.57	-	-
[71]	2019		√		√		√		√			√	√	√	√	-	73.18	-
[81]	2019		√		√		√	I				√	√	√	√	-	66.78	-
[150]	2019		√			√		G		√		√	√	√	√	-	66.75	-
[2]	2019		√					G		√		√	√	√	√	-	60.97	-
[96]	2019		√					I, W				√	√	√	√	-	59.39	-
[142]	2019		√				√	W	√			√	√	√	√	-	58.62	-
[69]	2019		√		√	√	√					√	√	√	√	-	53.45	-
[42]	2019			L												-	40.17	-
[83]	2018		√		√		√	I	√					√		58.3	-	62.4
[23]	2018		√		√	√	√	W				√	√	√		56.52	-	-
[50]	2018	√	√	H		√						√		√		56.25	-	61.5
[67]	2018		√		√		√	W				√	√	√		55.8	-	59.9
[22]	2018		√		√	√	√					√		√		54.79	-	57.47
[48]	2018		√				√	I		√		√		√		54.51	-	57.74
[68]	2018	√		C, H				I								50.16	-	-
[11]	2018			L												34.09	-	48.75
[116]	2018		√									√	√	√		-	-	53.98
[21]	2017	√										√		√		53.14	-	56.78
[144]	2017	√				√						√	√			51.68	-	43.26
[63]	2017		√				√					√		√		45.15	-	-
[66]	2016	√										√		√		49.49	-	-
[112]	2016	√		C, H					√					√		47.4	-	-
[20]	2016	√										√		√		44.27	-	-
[25]	2016	√										√		√		41.75	-	-
[58]	2016	√				√						√		√		41.19	-	-
[93]	2016	√				√								√		40	-	-
[148]	2016	√				√							√			38.08	-	-
[139]	2016		√				√							√		40.56	-	-
[147]	2016		√										√			39.9	-	-
[126]	2014	√		C								√		√		39.39	-	-
[108]	2014			C								√				36.79	-	-
[65]	2014	√										√		√		35.01	-	-

For the limitation of space, we use some abbreviations: In column "Approach," "S," "N," "E," "C," "L," and "H" refer to SMT, NMT, edit, classification, language model, and hybrid-based approaches; in column "Performance Boosting Technique," "P," "R," "E," "I," "G," and "W" refer to pre-training, reranking, ensemble, iterative decoding, auxiliary GED, and edit-weighted optimization, respectively; in column "Data Augmentation," "N," "B," "E" refer to noising, back translation, and better exploitation of training data; and in column "Training Data," "N," "F," "L," and "W" refer to NUCLE, FCE, Lang-8, and W&I+LOCNESS.

6.4.1 General Trend. Great progress has been achieved in the past six years after CoNLL-2014 shared task. Although submissions to CoNLL-2014 shared task we included in Table 7 rank top 3 [41, 65, 108], they are largely superseded by more powerful systems. Incremental improvement on performance is observed when it comes closer to 2019. Several systems that participated in

Table 8. Improvements Brought by Pre-training

Source	Approach	Test Set	Evaluation Metric	Improvement
[152]	NMT	CoNLL-2014 test	M^2	54.67 → 58.8
[141]	NMT	BEA-2019 dev	ERRANT	47.01 → 55.37
[141]	NMT	BEA-2019 test	ERRANT	53.24 → 67.21
[67]	NMT	CoNLL-2014 test	M^2	51.0 → 54.1

BEA-2019 are also evaluated on CoNLL-2014 test set using M^2 evaluation metric [19, 51, 74, 123, 141], and their scores have significantly surpassed the scores of top-ranking participants in CoNLL-2014. As we can see from the table, several factors contribute to this significant progress. First, the selection of approaches gradually changes from SMT, classification, LM, and hybrid to more powerful NMT, and more recently appealing Edit. Second, more and more task-specific performance-boosting techniques are researched and combined with approaches. Third, data augmentation is researched and is increasingly welcomed to train better GEC systems. Fourth, the recently proposed dataset W&I+LOCNESS provides more training data for GEC systems after 2019.

6.4.2 Approaches. It is obvious that NMT-based approach has become dominant since 2017. All the submissions to BEA-2019 shared task are NMT-based, except for Reference [42], participating in low-resource track only and adopting language model-based approach. Nearly two-thirds of systems in BEA-2019 shared task are based on Transformer, which also leads to similar scores in some NMT-based systems [81, 141, 150]. Actually, many participants in BEA-2019 share the same architecture and could only be differentiated via other non-approach level properties. It is obvious that compared to SMT, classification, and LM, NMT-based approaches are much more commonly combined with the performance-boosting techniques. Only with NMT-based approaches can a system be combined with pre-training, ensemble, auxiliary GED, and edit-weighted optimization. Although classification-based and SMT-based approaches made great contributions to GEC at early stages, they are almost abandoned in recent top-performing systems. LM-based approaches are valuable for low-resource GEC, but their performances are not comparable to NMT-based systems. Edit-based approaches, however, have the potential to outperform the NMT-based approaches.

6.4.3 Performance-boosting Techniques. In this section, we discuss the pattern of how the techniques are applied in GEC systems and the effect of them. Beyond Table 7, we further summarize ablation studies in the works, applying the techniques for a more quantified analysis. It is worthy to note that although some techniques always exist in top-ranking systems and have higher expectation on performance-boosting, integrating a technique does not always guarantee an improvement on baseline.

Pre-training. Pre-training has become the requisite for powerful GEC systems no matter what other techniques are involved, for the sake of its incomparable ability to incorporate artificial training data or large monolingual data, as we discussed in Section 4.1. In other words, a well-performing GEC system must benefit partly from pre-training; for example, systems with the top five scores on CoNLL-2014 test set [4, 51, 74, 141, 152] and systems ranking the top four on BEA-2019 test set [19, 51, 71, 74]. Table 8 shows the huge improvements gained by pre-training. It is highly recommended that both the two types of pre-training could be used to boost a GEC system's performance. Actually, pre-training has become an integral part for current GEC systems.

Ensemble. Ensemble is the most widely adopted technique and always yields better performance than single model, as shown in Table 9. More than half of the systems in 2019 used the combination of NMT-based GEC approach, pre-training, and ensemble of multiple basic model with identical architecture. It is worthy to mention that the system with the highest ERRANT

Table 9. Improvements Brought by Ensemble

Source	Approach	Test Set	Evaluation Metric	Improvement
[152]	NMT	CoNLL-2014 test	M^2	59.75 → 61.15
[83]	NMT	CoNLL-2014 test	M^2	54.9 → 58.3
[83]	NMT	JFLEG	GLEU	59.3 → 62.4
[141]	NMT	CoNLL-2014 test	M^2	60.9 → 63.2
[141]	NMT	JFLEG	GLEU	60.8 → 62.6
[19]	NMT	CoNLL-2014 test	M^2	63.05 → 69.06
[123]	NMT	CoNLL-2014 test	M^2	55.72 → 58.9
[122]	SMT, NMT, LM	CoNLL-2014 test	M^2	56.33 → 58.4
[122]	SMT, NMT, LM	JFLEG	GLEU	55.39 → 55.6
[26]	NMT	CoNLL-2014 test	M^2	53.06 → 54.25
[22]	NMT	CoNLL-2014 test	M^2	46.38 → 49.33

Table 10. Improvements Brought by Reranking

Source	Approach	Test Set	Evaluation Metric	Improvement
[26]	NMT	CoNLL-2014 test	M^2	54.25 → 55.57
[150]	NMT	BEA-2019 dev	ERRANT	49.68 → 50.39
[51]	NMT	BEA-2019 dev	ERRANT	52.3 → 53
[69]	NMT	BEA-2019	ERRANT	32.6 → 35.35
[23]	NMT	CoNLL-2014 test	M^2	55.72 → 55.97
[50]	SMT, NMT, Hybrid	CoNLL-2014 test	M^2	53.51 → 54.95
[22]	NMT	CoNLL-2014 test	M^2	46.38 → 49.33
[144]	SMT	CoNLL-2014 test	M^2	49.34 → 51.08

score (73.18) [71] and system with the fourth highest score (66.78) [81] on BEA-2019 test set are based on sophisticated ensemble strategies combining the output of different GEC models, highlighting the benefit brought by combining different GEC systems. We have discussed them in detail in Section 4.3.

Reranking. While pre-training and ensemble are restricted in NMT-based systems, reranking is more universal, since it is more approach-independent. Reranking has been researched as early as 2014 and combined with SMT-based systems [41, 58, 93, 148]. However, although many top-performing systems adopt reranking (systems performing outstanding on BEA-2019 test set), it does not yield significant improvement on baselines, as shown in Table 10.

Iterative correction, auxiliary GED, and edit-weighted optimization. Ablation results are collected in Table 11. Iterative correction is not as widely applied and effective, but can be also used for systems-based on various approaches [4, 68]. Compared to reranking, iterative correction yields slightly more improvement on baselines [83]. Auxiliary GED and edit-weighted training object are less extensively used and restricted in NMT-based systems. For all the three systems boosted by auxiliary GED, the improvements are slight, so it is logical that auxiliary GED had better to be used with other techniques. Edit-weighted optimization does not always promise a better performance, especially when it is not combined with pre-training and reranking [96, 142].

6.4.4 Data Augmentation. An important discrepancy between the two data-augmentation methods is that data generated via noising is always used for pre-training [19, 51, 71, 74, 141, 152], while data generated by back translation is viewed as equivalent to authentic training data. This is

Table 11. Improvements Brought by Auxiliary GED, Iterative Correction, and Edit-weighted Optimization

Technique	Source	Approach	Test Set	Evaluation Metric	Improvement
Auxiliary GED	[152]	NMT	CoNLL-2014 test	M^2	58.8 → 59.76
	[2]	NMT	BEA-2019 test	ERRANT	59.88 → 60.97
	[74]	NMT	BEA-2019 test	ERRANT	69.8 → 70.2
Iterative Correction	[83]	NMT	CoNLL-2014 test	M^2	55.2 → 58.3
	[83]	NMT	JFLEG	GLEU	57.0 → 62.4
	[81]	NMT	BEA-2019 dev	ERRANT	53.05 → 53.72
	[96]	NMT	BEA-2019 dev	ERRANT	43.29 → 44.27
Edit-weighted Optimization	[67]	NMT	CoNLL-2014 test	M^2	48.0 → 51.0

because data generated via noising may not precisely capture the distribution of error pattern in realistic data, although many explorations are conducted to improve the quality of the noisy data, as we discussed in Section 5.1. Besides, it can be concluded that as the noising strategy becomes more sophisticated, better performance is gained; for example, the system pre-trained on the augmented data generated by sophisticated process achieved 63.2 M^2 $F_{0.5}$ score on CoNLL-2014 test set [141], about 5 points higher than the system pre-trained on the data generated by deterministic approach, which achieved 58.3 M^2 $F_{0.5}$ on CoNLL-2014 test set [83].

6.4.5 Training Data and Evaluation. MT-based systems rely heavily on parallel training data. Almost all the NMT-based participants in BEA-2019 exploit all the training corpora available, and NUCLE and Lang-8 are used by almost all the MT-based systems. The newly available high-quality dataset W&I+LOCNESS also contributes to the better results of participants in BEA-2019 and perhaps has the highest quality among all the datasets. FCE is not used as extensively, possibly due to the lack of multiple references. Although the performances of classification-based and LM-based systems are relatively limited compared to MT-based systems, the nature of their low reliance on parallel training data adds many advantages to them for the sake of better application in low-resource scenario; for example, grammar error correction for other languages instead of English. In aspect of evaluation, no system obtains the highest score on all test sets according to their corresponding metric. This is suggestive that evaluation results may vary with different test sets, about which we discuss more in Section 7.

7 PROSPECTIVE DIRECTION

In this section, we discuss several prospective directions for future work.

- **Dataset.** Most of existing datasets for GEC focus on ESL problem, where the source of samples often holds limited sentence proficiency, topics, and L1s, compared to a more generalized GEC task. GEC systems obtained from such parallel data could not be directly employed in more complex but realistic scenarios where the incorrect sentences may be created by people diverse in proficiencies and domains. As exposed by some empirical studies [43], systems selected by testing scores on traditional GEC datasets are less likely to be as effective at correcting errors when applying them on the realistic web-crawled data, even with several domain adaption strategies. The results may come from the difference between existing corpus and more generalized data concerning with domains and error distributions. Therefore, GEC datasets with more diverse samples should be explored for more generalized GEC applications.
- **Edit-based approaches.** Recently, researchers have proposed edit-based approaches for GEC task, as we described in Section 3.3, which decodes faster than NMT-based approaches and achieves comparable results to SOTA performances. We expect that the edit-based

approaches would become mainstream in the future. There are still many issues to consider with edit-based approaches. First, although researchers have proposed different definitions for edit and methods to obtain edit sequences, there is a lack of research on which edit definition method is optimal on representing the edit space. Second, pre-training models are commonly used as the body architecture in this method. It is worthy to explore which pre-training model is the most helpful for GEC.

- **Adaptation to L1.** Most of the existing works are L1 agnostic, treating text written by writers with various first languages as equivalent. However, for better application of GEC, prospective GEC systems should provide more individualized feedback to English learners, taking consideration of their L1s. Some works have already focused on several specific first languages, such as incorporating L1-specific text to SMT-based approaches on Chinese, Russian, and Spanish [20], and adapting NMT-based models to 12 L1s, including Chinese, French, German, Greek, Italian, and so on [95]. However, there is still much room to be explored; especially in terms of datasets and evaluation metrics, it is significant to know how to measure the impact of data for personalized learners.
- **Cross-sentence level GEC.** As the analysis in Section 6.3 shows, current GEC systems struggle most with content word errors. This is because the systems make sentence-level grammatical error correction, considering only the semantic information within the sentence. In fact, semantic information across sentences is conducive to grammatical error correction at the paragraph level, such as the information about pronoun, verb tense, noun number, and so on. One NMT-based system [26] considers wider contexts and demonstrates that cross-sentence grammatical error correction is helpful. Although the considered context is limited within only two sentences, this work is illuminating for prospective GEC researches, such as modeling broader context, even in document-level. However, this presents challenges on model design to fully utilize the context-aware information and the datasets to include wider context where the erroneous sentence occurs.
- **Better evaluation.** Systems have always been evaluated on a single test set, such as in the BEA-2019 shared task. However, different test sets lead to inconsistent evaluation performance, as shown in Table 7. Cross-corpora evaluation has been preliminary proposed [91] and is worthy to receive more attention and become as important as uni-corpora evaluation in the future. Another issue in the evaluation is that, although existing evaluation metrics capture grammatical correction and fluency, no one measures the preserving of meaning or adequacy, which is also necessary to consider when evaluating a GEC system. So, a more admirable metric should explain the grammar, fluency, and meaning loyalty of system output.

8 CONCLUSION

We present the first survey in **grammar error correction (GEC)** for a comprehensive retrospect of existing progress. We first give definition of the task and introduction of public datasets and annotation schema. Then, six dominant basic approaches and their comparison are explained in detail with emphasis on MT-based approaches. After that, we organize the numerous performance-boosting techniques into six branches and describe their application and development in GEC, and three data-augmentation methods are separately introduced due to their importance. More importantly, in Section 6, after the introduction of the standard evaluation, we give an in-depth analysis based on empirical results in aspects of approaches, error types, and GEC systems for a clearer pattern of existing works. Finally, we present five prospective directions based on existing progress in GEC. We hope our effort could make contributions for future researches in the community.

REFERENCES

[1] Omri Abend and Ari Rappoport. 2013. Universal conceptual cognitive annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics 228–238.

[2] Hiroki Asano, Masato Mita, Tomoya Mizumoto, and Jun Suzuki. 2019. The AIP-Tohoku system at the BEA-2019 shared task. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. 176–182.

[3] Hiroki Asano, Tomoya Mizumoto, and Kentaro Inui. 2017. Reference-based metrics can be replaced with reference-less metrics in evaluating grammatical error correction systems. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*. Asian Federation of Natural Language Processing, 343–348.

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

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*.

[6] Bibek Behera and Pushpak Bhattacharyya. 2013. Automated grammar correction using hierarchical phrase-based statistical machine translation. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*. Asian Federation of Natural Language Processing, 937–941.

[7] Adriane Boyd. 2018. Using Wikipedia edits in low resource grammatical error correction. In *Proceedings of the 4th Workshop on Noisy User-generated Text*. 79–84.

[8] Adriane Boyd. 2018. Using Wikipedia edits in low resource grammatical error correction. In *Proceedings of the 4th Workshop on Noisy User-generated Text*. Association for Computational Linguistics, 79–84. DOI: https://doi.org/10.18653/v1/w18-6111

[9] Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting ESL errors using phrasal SMT techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 249–256.

[10] Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Comput. Ling.* 19, 2 (1993), 263–311.

[11] Christopher Bryant and Ted Briscoe. 2018. Language model-based grammatical error correction without annotated training data. In *Proceedings of the 13th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 247–253.

[12] Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 52–75.

[13] Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 793–805.

[14] Christopher Bryant and Hwee Tou Ng. 2015. How far are we from fully automatic high quality grammatical error correction? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, 697–707.

[15] Mengyun Chen, Tao Ge, Xingxing Zhang, Furu Wei, and Ming Zhou. 2020. Improving the efficiency of grammatical error correction with erroneous span detection and correction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 7162–7169.

[16] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 1724–1734.

[17] Martin Chodorow, Markus Dickinson, Ross Israel, and Joel R. Tetreault. 2012. Problems in evaluating grammatical error detection systems. In *Proceedings of the 24th International Conference on Computational Linguistics*. 611–628.

[18] Martin Chodorow, Joel Tetreault, and Na-Rae Han. 2007. Detection of grammatical errors involving prepositions. In *Proceedings of the 4th ACL-SIGSEM Workshop on Prepositions*. Association for Computational Linguistics, 25–30.

[19] Yo Joong Choe, Jiyeon Ham, Kyubyong Park, and Yeoil Yoon. 2019. A neural grammatical error correction system built on better pre-training and sequential transfer learning. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 213–227.

[20] Shamil Chollampatt, Duc Tam Hoang, and Hwee Tou Ng. 2016. Adapting grammatical error correction based on the native language of writers with neural network joint models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1901–1911.

[21] Shamil Chollampatt and Hwee Tou Ng. 2017. Connecting the dots: Towards human-level grammatical error correction. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 327–333.

[22] Shamil Chollampatt and Hwee Tou Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI'18), the 30th Innovative Applications of Artificial Intelligence (IAAI'18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI'18)*. 5755–5762.

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

[24] Shamil Chollampatt and Hwee Tou Ng. 2018. A reassessment of reference-based grammatical error correction metrics. In *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 2730–2741.

[25] Shamil Chollampatt, Kaveh Taghipour, and Hwee Tou Ng. 2016. Neural network translation models for grammatical error correction. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 2768–2774.

[26] Shamil Chollampatt, Weiqi Wang, and Hwee Tou Ng. 2019. Cross-sentence grammatical error correction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 435–445.

[27] Leshem Choshen and Omri Abend. 2018. Automatic metric validation for grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 1372–1382.

[28] Leshem Choshen and Omri Abend. 2018. Inherent biases in reference-based evaluation for grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 632–642.

[29] Leshem Choshen and Omri Abend. 2018. Reference-less measure of faithfulness for grammatical error correction. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 124–129.

[30] Marta R. CostaJussà and José A. R. Fonollosa. 2016. Character-based neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

[31] Nicholls D. 2003. The Cambridge learner corpus: Error coding and analysis for lexicography and ELT. In *Proceedings of the Corpus Linguistics Conference*. 572–581.

[32] Daniel Dahlmeier and Hwee Tou Ng. 2012. A beam-search decoder for grammatical error correction. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 568–578.

[33] Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 568–572.

[34] Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner English: The NUS corpus of learner English. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 22–31.

[35] Rachele De Felice and Stephen G. Pulman. 2008. A classifier-based approach to preposition and determiner error correction in L2 English. In *Proceedings of the 22nd International Conference on Computational Linguistics*. COLING 2008 Organizing Committee, 169–176.

[36] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 4171–4186.

[37] Nadir Durrani, Alexander Fraser, Helmut Schmid, Hieu Hoang, and Philipp Koehn. 2013. Can Markov models over minimal translation units help phrase-based SMT? In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 399–405.

[38] Nava Ehsan and Heshaam Faili. 2013. Grammatical and context-sensitive error correction using a statistical machine translation framework. *Software: Practice and Experience* 43, 2 (2013), 187–206.

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

[40] Mariano Felice and Zheng Yuan. 2014. Generating artificial errors for grammatical error correction. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 116–126.

[41] Mariano Felice, Zheng Yuan, Øistein E. Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. 2014. Grammatical error correction using hybrid systems and type filtering. In *Proceedings of the 18th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 15–24.

[42] Simon Flachs, Ophélie Lacroix, and Anders Søgaard. 2019. Noisy channel for low resource grammatical error correction. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 191–196.

[43] Simon Flachs, Ophélie Lacroix, Helen Yannakoudakis, Marek Rei, and Anders Søgaard. 2020. Grammatical error correction in low error density domains: A new benchmark and analyses. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 8467–8478.

[44] Jennifer Foster and Oistein Andersen. 2009. GenERRate: Generating errors for use in grammatical error detection. In *Proceedings of the 4th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 82–90.

[45] Michael Gamon. 2010. Using mostly native data to correct errors in learners' writing. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 163–171.

[46] Michael Gamon, Jianfeng Gao, Chris Brockett, Alexandre Klementiev, William B. Dolan, Dmitriy Belenko, and Lucy Vanderwende. 2008. Using contextual speller techniques and language modeling for ESL error correction. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing*.

[47] Michael Gamon, Claudia Leacock, Chris Brockett, William Dolan, Jianfeng Gao, Dmitriy Belenko, and Alexandre Klementiev. 2013. Using statistical techniques and web search to correct ESL errors. *CALICO J.* 26 (01 2013), 491–511.

[48] Tao Ge, Furu Wei, and Ming Zhou. 2018. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 1055–1065.

[49] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning*. 1243–1252.

[50] Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2018. Near human-level performance in grammatical error correction with hybrid machine translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 284–290.

[51] Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 252–263.

[52] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 1631–1640.

[53] Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 140–149.

[54] Masato Hagiwara and Masato Mita. 2020. GitHub typo corpus: A large-scale multilingual dataset of misspellings and grammatical errors. In *Proceedings of the 12th Language Resources and Evaluation Conference*. European Language Resources Association, 6761–6768. Retrieved from https://www.aclweb.org/anthology/2020.lrec-1.835/.

[55] Na-Rae HAN, Martin Chodorow, and Claudia Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering* 12, 2 (2006), 115–129.

[56] Kenneth Heafield and Alon Lavie. 2010. Combining machine translation output with open source: The Carnegie Mellon multi-engine machine translation scheme. *Prague Bull. Math. Ling.* 93 (Jan. 2010), 27–36.

[57] Michael Heilman, Aoife Cahill, Nitin Madnani, Melissa Lopez, Matthew Mulholland, and Joel Tetreault. 2014. Predicting grammaticality on an ordinal scale. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 174–180.

[58] Duc Tam Hoang, Shamil Chollampatt, and Hwee Tou Ng. 2016. Exploiting N-best hypotheses to improve an SMT approach to grammatical error correction. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 2803–2809. Retrieved from http://www.ijcai.org/Abstract/16/398.

[59] Phu Mon Htut and Joel Tetreault. 2019. The unbearable weight of generating artificial errors for grammatical error correction. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 478–483.

[60] Emi Izumi, Kiyotaka Uchimoto, Toyomi Saiga, Thepchai Supnithi, and Hitoshi Isahara. 2003. Automatic error detection in the Japanese learners' English spoken data. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 145–148.

[61] Sébastien Jean, KyungHyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*. 1–10.

[62] Sebastien Jean, Stanislas Lauly, Orhan Firat, and Kyunghyun Cho. 2017. Does Neural Machine Translation Benefit from Larger Context? arXiv preprint arXiv:1704.05135.

[63] Jianshu Ji, Qinlong Wang, Kristina Toutanova, Yongen Gong, Steven Truong, and Jianfeng Gao. 2017. A nested attention neural hybrid model for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 753–762.

[64] Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

[65] Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2014. The AMU system in the CoNLL-2014 shared task: Grammatical error correction by data-intensive and feature-rich statistical machine translation. In *Proceedings of the 18th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 25–33.

[66] Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1546–1556.

[67] Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 595–606.

[68] Zhu Kaili, Chuan Wang, Ruobing Li, Yang Liu, Tianlei Hu, and Hui Lin. 2018. A simple but effective classification model for grammatical error correction. *CoRR* abs/1807.00488 (2018). arXiv:1807.00488

[69] Masahiro Kaneko, Kengo Hotate, Satoru Katsumata, and Mamoru Komachi. 2019. TMU transformer system using BERT for re-ranking at BEA 2019 grammatical error correction on restricted track. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 207–212.

[70] Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 4248–4254.

[71] Yoav Kantor, Yoav Katz, Leshem Choshen, Edo Cohen-Karlik, Naftali Liberman, Assaf Toledo, Amir Menczel, and Noam Slonim. 2019. Learning to combine grammatical error corrections. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 139–148.

[72] Sudhanshu Kasewa, Pontus Stenetorp, and Sebastian Riedel. 2018. Wronging a right: Generating better errors to improve grammatical error detection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 4977–4983.

[73] Satoru Katsumata and Mamoru Komachi. 2020. Stronger baselines for grammatical error correction using a pretrained encoder-decoder model. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*. 827–832.

[74] 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 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. 1236–1242.

[75] Philipp Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press. Retrieved from http://www.statmt.org/book/.

[76] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 177–180.

[77] Shaohui Kuang, Deyi Xiong, Weihua Luo, and Guodong Zhou. 2018. Modeling coherence for neural machine translation with dynamic and topic caches. In *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 596–606.

[78] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *Proceedings of the 6th International Conference on Learning Representations*.

[79] John Lee and Stephanie Seneff. 2008. Correcting misuse of verb forms. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 174–182.

[80] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language

generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7871–7880.

[81] Ruobing Li, Chuan Wang, Yefei Zha, Yonghong Yu, Shiman Guo, Qiang Wang, Yang Liu, and Hui Lin. 2019. The LAIX systems in the BEA-2019 GEC shared task. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 159–167.

[82] Jared Lichtarge, Chris Alberti, and Shankar Kumar. 2020. Data weighted training strategies for grammatical error correction. *Trans. Assoc. Comput. Lingu.* 8 (2020), 634–646.

[83] Jared Lichtarge, Christopher Alberti, Shankar Kumar, Noam Shazeer, and Niki Parmar. 2018. Weakly supervised grammatical error correction using iterative decoding. *CoRR* abs/1811.01710 (2018).

[84] Jared Lichtarge, Chris Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. 2019. Corpora generation for grammatical error correction. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 3291–3301.

[85] Xiaofei Lu. 2015. Automated grammatical error detection for language learners, second edition claudia leacock[1], Martin Chodorow[2], Michael Gamon[3], and Joel Tetreault[4] ([1]CTB McGraw-Hill; [2]hunter college and the graduate center, city university of New York; [3]Microsoft Research; [4]Yahoo! Labs) Morgan & Claypool (synthesis lectures on human language technologies, edited by Graeme Hirst, volume 25), 2014, xv+154pp; paperbound, ISBN 978-1-62705-013-5. *Comput. Linguistics* 41, 1 (2015), 149–151.

[86] Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2014. Automated grammatical error detection for language learners. *Synthesis Lectures on Human Language Technologies* 7, 1 (2014), 1–170.

[87] Victor Makarenkov, Lior Rokach, and Bracha Shapira. 2019. Choosing the right word: Using bidirectional LSTM tagger for writing support systems. *Eng. Appl. Artif. Intell.* 84 (2019), 1–10.

[88] Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 5054–5065.

[89] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space.

[90] Masato Mita, Shun Kiyono, Masahiro Kaneko, Jun Suzuki, and Kentaro Inui. 2020. A self-refinement strategy for noise reduction in grammatical error correction.

[91] Masato Mita, Tomoya Mizumoto, Masahiro Kaneko, Ryo Nagata, and Kentaro Inui. 2019. Cross-corpora evaluation and analysis of grammatical error correction models—Is single-corpus evaluation enough? In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 1309–1314.

[92] Tomoya Mizumoto, Yuta Hayashibe, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2012. The effect of learner corpus size in grammatical error correction of ESL writings. In *Proceedings of COLING 2012: Posters*. The COLING 2012 Organizing Committee, 863–872.

[93] Tomoya Mizumoto and Yuji Matsumoto. 2016. Discriminative reranking for grammatical error correction with statistical machine translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 1133–1138.

[94] Daniel Naber. 2003. A rule-based style and grammar checker. (01 2003).

[95] Maria Nadejde and Joel Tetreault. 2019. Personalizing grammatical error correction: Adaptation to proficiency level and L1. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT'19)*. Association for Computational Linguistics, 27–33.

[96] Jakub Náplava and Milan Straka. 2019. CUNI system for the building educational applications 2019 shared task: Grammatical error correction. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 183–190.

[97] 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*. Association for Computational Linguistics, 588–593.

[98] Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2016. There's no comparison: Reference-less evaluation metrics in grammatical error correction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2109–2115.

[99] Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 229–234.

[100] 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 18th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 1–14.

[101] Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem N. Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR - grammatical error correction: Tag, not rewrite. In *Proceedings of the 15th Workshop on Innovative Use of NLP for Building Educational Applications*. 163–170.

[102] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 311–318.

[103] Martin Popel and Ondrej Bojar. 2018. Training tips for the transformer model. *Prague Bull. Math. Ling.* 110 (2018), 43–70.

[104] Mengyang Qiu, Xuejiao Chen, Maggie Liu, Krishna Parvathala, Apurva Patil, and Jungyeul Park. 2019. Improving precision of grammatical error correction with a cheat sheet. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 240–245.

[105] Prajit Ramachandran, Peter Liu, and Quoc Le. 2017. Unsupervised pretraining for sequence to sequence learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 383–391.

[106] Marek Rei, Mariano Felice, Zheng Yuan, and Ted Briscoe. 2017. Artificial error generation with machine translation and syntactic patterns. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 287–292.

[107] Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, and Dan Roth. 2013. The University of Illinois system in the CoNLL-2013 shared task. In *Proceedings of the 17th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 13–19.

[108] Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, Dan Roth, and Nizar Habash. 2014. The Illinois-Columbia system in the CoNLL-2014 shared task. In *Proceedings of the 18th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 34–42.

[109] Alla Rozovskaya and Dan Roth. 2010. Annotating ESL errors: Challenges and rewards. In *Proceedings of the NAACL HLT 5th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 28–36.

[110] Alla Rozovskaya and Dan Roth. 2010. Generating confusion sets for context-sensitive error correction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 961–970.

[111] Alla Rozovskaya and Dan Roth. 2010. Training paradigms for correcting errors in grammar and usage. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 154–162.

[112] Alla Rozovskaya and Dan Roth. 2016. Grammatical error correction: Machine translation and classifiers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2205–2215.

[113] Alla Rozovskaya and Dan Roth. 2019. Grammar error correction in morphologically-rich languages: The case of Russian. *Trans. Assoc. Comput. Ling.* 7 (2019), 1–17. Retrieved from https://transacl.org/ojs/index.php/tacl/article/view/1454.

[114] Alla Rozovskaya, Mark Sammons, and Dan Roth. 2012. The UI system in the HOO 2012 shared task on error correction. In *Proceedings of the 7th Workshop on Building Educational Applications Using NLP*. Association for Computational Linguistics, 272–280.

[115] Keisuke Sakaguchi, Courtney Napoles, Matt Post, and Joel Tetreault. 2016. Reassessing the goals of grammatical error correction: Fluency instead of grammaticality. *Trans. Assoc. Comput. Ling.* 4 (2016), 169–182.

[116] Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2017. Grammatical error correction with neural reinforcement learning. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*. Asian Federation of Natural Language Processing, 366–372.

[117] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 86–96.

[118] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 1715–1725.

[119] Yikang Shen, Zhouhan Lin, Chin-Wei Huang, and Aaron Courville. 2017. Neural Language Modeling by Jointly Learning Syntax and Lexicon. *arXiv preprint arXiv:1711.02013*.

[120] Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron C. Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks. In *Proceedings of the 7th International Conference on Learning Representations*.

[121] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15, 1 (2014), 1929–1958. Retrieved from http://dl.acm.org/citation.cfm?id=2670313.

[122] Felix Stahlberg, Christopher Bryant, and Bill Byrne. 2019. Neural grammatical error correction with finite state transducers. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 4033–4039.

[123] Felix Stahlberg and Bill Byrne. 2019. The CUED's grammatical error correction systems for BEA-2019. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 168–175.

[124] Felix Stahlberg and Shankar Kumar. 2020. Seq2Edits: Sequence transduction using span-level edit operations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 5147–5159.

[125] Chengjie Sun, Xiaoqiang Jin, Lin Lei, Yuming Zhao, and Xiaolong Wang. 2015. *Convolutional Neural Networks for Correcting English Article Errors*. Springer International Publishing.

[126] Raymond Hendy Susanto, Peter Phandi, and Hwee Tou Ng. 2014. System combination for grammatical error correction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 951–962.

[127] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 3104–3112.

[128] Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for ESL learners using global context. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 198–202.

[129] Joel Tetreault and Martin Chodorow. 2008. Native judgments of non-native usage: Experiments in preposition error detection. In *Proceedings of the Workshop on Human Judgements in Computational Linguistics*. COLING 2008 Organizing Committee, 24–32.

[130] Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010. Using parse features for preposition selection and error detection. In *Proceedings of the Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 353–358.

[131] Joel R. Tetreault and Martin Chodorow. 2008. The ups and downs of preposition error detection in ESL writing. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING'08)*. COLING 2008 Organizing Committee, 865–872.

[132] Joel R. Tetreault, Martin Chodorow, and Nitin Madnani. 2014. Bucking the trend: Improved evaluation and annotation practices for ESL error detection systems. *Lang. Resour. Eval* 48, 1 (2014), 5–31.

[133] Jörg Tiedemann and Yves Scherrer. 2017. Neural machine translation with extended context. In *Proceedings of the 3rd Workshop on Discourse in Machine Translation*. Association for Computational Linguistics, 82–92.

[134] 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 *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 5998–6008.

[135] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning (ICML'08)*. Association for Computing Machinery, 1096–1103.

[136] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 2692–2700.

[137] Chuan Wang, Ruobing Li, and Hui Lin. 2017. Deep context model for grammatical error correction. In *Proceedings of the 7th ISCA International Workshop on Speech and Language Technology in Education*. 167–171.

[138] Wei Wang, Taro Watanabe, Macduff Hughes, Tetsuji Nakagawa, and Ciprian Chelba. 2018. Denoising neural machine translation training with trusted data and online data selection. In *Proceedings of the 3rd Conference on Machine Translation*. Association for Computational Linguistics, 133–143.

[139] Ziang Xie, Anand Avati, Naveen Arivazhagan, Dan Jurafsky, and Andrew Y. Ng. 2016. Neural language correction with character-based attention. *CoRR* (2016).

[140] Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Ng, and Dan Jurafsky. 2018. Noising and denoising natural language: Diverse backtranslation for grammar correction. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 619–628.

[141] Shuyao Xu, Jiehao Zhang, Jin Chen, and Long Qin. 2019. Erroneous data generation for grammatical error correction. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 149–158.

[142] Liner Yang and Chencheng Wang. 2019. The BLCU system in the BEA 2019 shared task. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 197–206.

[143] Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 180–189.

[144] Helen Yannakoudakis, Marek Rei, Øistein E. Andersen, and Zheng Yuan. 2017. Neural sequence-labelling models for grammatical error correction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2795–2806.

[145] Ippei Yoshimoto, Tomoya Kose, Kensuke Mitsuzawa, Keisuke Sakaguchi, Tomoya Mizumoto, Yuta Hayashibe, Mamoru Komachi, and Yuji Matsumoto. 2013. NAIST at 2013 CoNLL grammatical error correction shared task. In *Proceedings of the 17th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 26–33.

[146] Zheng Yuan. 2017. *Grammatical Error Correction in Non-native English*. Technical Report. University of Cambridge, Computer Laboratory.

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

[148] Zheng Yuan, Ted Briscoe, and Mariano Felice. 2016. Candidate re-ranking for SMT-based grammatical error correction. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 256–266.

[149] Zheng Yuan and Mariano Felice. 2013. Constrained grammatical error correction using statistical machine translation. In *Proceedings of the 17th Conference on Computational Natural Language Learning: Shared Task*. Association for Computational Linguistics, 52–61.

[150] Zheng Yuan, Felix Stahlberg, Marek Rei, Bill Byrne, and Helen Yannakoudakis. 2019. Neural and FST-based approaches to grammatical error correction. In *Proceedings of the 14th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 228–239.

[151] Shengsheng Zhang, Yaping Huang, Yun Chen, Liner Yang, Chencheng Wang, and Erhong Yang. 2021. Few-shot domain adaptation for grammatical error correction via meta-learning. *CoRR* abs/2101.12409 (2021).

[152] Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 156–165.

[153] Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. Overview of the NLPCC 2018 shared task: Grammatical error correction. In *Proceedings of the 7th CCF International Conference on Natural Language Processing and Chinese Computing*. 439–445.

[154] Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. Overview of the NLPCC 2018 shared task: Grammatical error correction. In *Proceedings of the 7th CCF International Conference on Natural Language Processing and Chinese Computing*. Springer, 439–445. Retrieved from https://doi.org/10.1007/978-3-319-99501-4_41

[155] Zewei Zhao and Houfeng Wang. 2020. MaskGEC: Improving neural grammatical error correction via dynamic masking. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence, the 32nd Innovative Applications of Artificial Intelligence Conference, the 10th AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 1226–1233.

Received December 2020; revised May 2021; accepted July 2021