

PAPER

A Method for Correcting Preposition Errors in Learner English with Feedback Messages

Ryo NAGATA^{†a)}, Member and Edward WHITTAKER^{††}, Nonmember

SUMMARY This paper presents a novel framework called *error case frames* for correcting preposition errors. They are case frames specially designed for describing and correcting preposition errors. Their most distinct advantage is that they can correct errors with feedback messages explaining why the preposition is erroneous. This paper proposes a method for automatically generating them by comparing learner and native corpora. Experiments show (i) automatically generated error case frames achieve a performance comparable to previous methods; (ii) error case frames are intuitively interpretable and manually modifiable to improve them; (iii) feedback messages provided by error case frames are effective in language learning assistance. Considering these advantages and the fact that it has been difficult to provide feedback messages using automatically generated rules, error case frames will likely be one of the major approaches for preposition error correction.

key words: grammatical error correction, preposition error, learners of English, feedback, case frames

1. Introduction

In English writing, learners of English often make grammatical errors. Typical examples include errors in article and preposition, which are most frequent ones among a wide variety of grammatical errors found in learner English. Table 1 shows that they appear in learner English very frequently. For instance, an article error occurs in every ten noun phrases (NPs) in the CoNLL-2013 test corpus [1].

One effective way of remedying grammatical errors is to give learners feedback such as error positions, correct forms, and explanatory notes. At the same time, it is time-consuming and laborious to correct them with feedback considering that they are frequent in the writing of learners of English. It would be especially hard to give feedback to every student in a normal English writing class where more than several students attend.

To overcome the difficulty, grammatical error correction has been intensively studied in recent years. Current methods mostly exploit machine learning-based classifiers to correct target errors; examples are errors in article [2]–[4], preposition [4]–[7], and tense [8], [9], to name a few. More recently, Wu and Ng [10] and Rozovskaya and Roth [11] proposed methods for simultaneously correcting multiple types of errors. Another major approach is to use a lan-

guage model (LM) for predicting correct words or phrases for a given context. Some researchers [12], [13] use statistical machine translation (SMT) for the same purpose, which can be regarded as the mixture of a classifier and an LM. With these diverse techniques, correction performance has dramatically improved against a wide variety of target errors, some of which are shown in Table 1.

Unfortunately, however, these previous methods have a crucial limitation. They are not capable of providing explanatory notes about errors they detect and correct despite the fact that appropriate explanatory notes are essential in language learning assistance [14]. Almost all previous methods are incapable of providing such explanatory notes; they are not suitable for generating open-class text by their nature. Some researchers [15], [16] made an attempt to develop hand-crafted rules for correcting errors with feedback messages. However, this approach encounters the tremendous difficulty of covering a wide variety of errors using hand-crafted rules.

In view of this background, this paper* presents a novel error correction framework called *error case frames* an example of which is shown in Fig. 1. They are case frames specially designed for describing and correcting errors in preposition attached to a verb; from Fig. 1, the reader may be able to see that it describes preposition errors such as **John often goes shopping to the market with his family.* and that the preposition *to* should be replaced with *at*. Their most significant advantage over previous methods is that they can provide learners with feedback messages, that is, explanatory notes describing why the detected preposition is erroneous and should be corrected as indicated, as shown in Fig. 2. This paper describes a method for automatically generating them by comparing learner and native corpora. Achieving a comparable correction performance, they have the following two advantages over the previous approaches: (i) they are intuitively interpretable and manually modifiable to enrich them; (ii) they are capable of providing feedback messages.

The rest of this paper is structured as follows. Section 2 introduces the definition of error case frames. Section 3 discusses the method for generating error case frames. Section 4 describes how to correct preposition errors with feedback messages by error case frames. Section 5 describes experiments conducted to evaluate error case frames. Section 6 discusses the experimental results.

Manuscript received September 6, 2016.

Manuscript revised January 19, 2017.

Manuscript publicized March 8, 2017.

[†]The author is with Konan University, Kobe-shi, 658–8501 Japan.

^{††}The author is with Inferret Limited, Northampton, U.K.

a) E-mail: rnagata@konan-u.ac.jp

DOI: 10.1587/transinf.2016EDP7380

*This paper is based on and extended from the work [17].

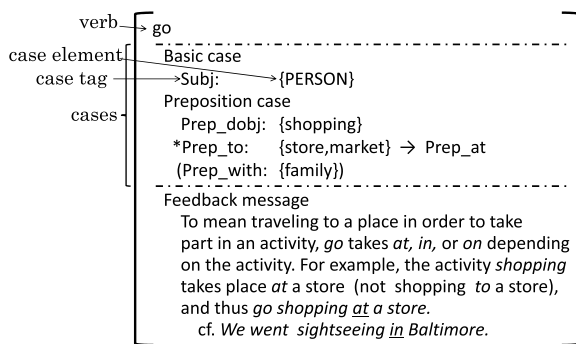
Table 1 Error rates and detection/correction performance on article and preposition errors.

Target task	Error rate	Method ¹	Performance ²	Target corpus ³
Article error correction	10% (errors/NP)	AP [18]	$F_1 = 0.335$	CoNLL2013
Article error detection	2.3% (errors/word)	DL [19]	$R=0.717, P=0.654, F_1 = 0.684$	Private corpus
Preposition error correction	10.7% (errors/prep.)	SMT [13]	$R = 0.176, P=0.346, F_1 = 0.233$	CoNLL2013
	1.6% (errors/token)	SMT [13]	$R = 0.115, P=0.385, F_1 = 0.176$	KJ
	1.6% (errors/token)	MEM [20]	$R = 0.167, P=0.310, F_1 = 0.217$	KJ

¹ AP (Averaged Perceptron); CRF (Conditional Random Field); DL (Decision List); SMT (Statistical Machine Translation); MEM (Maximum Entropy Model).

² The performances were basically cited from their original papers. For those that are not shown in their paper, the authors calculated them.

³ CoNLL2013 (CoNLL-2013 test corpus); KJ (Konan-JIEM learner corpus).

**Fig. 1** Example of an error case frame.

Target sentence: In the univarsity, I studied English in the morning.

Error: correct preposition *at*

Feedback message:

Though both *at* and *in* are prepositions of place, *at* is used to denote the place (university) to which the person belongs and where the learning activities take place.

Target sentence: When the day is holiday, I go to shopping, singing in Karaoke and talking in cafe.

Error: remove *to*

Feedback message:

Go directly takes the activity without a preposition when it means traveling to a place in order to take part in an activity by *go* and *~ing*, e.g., *I went shopping*.
Similar expressions: *go swimming, go fishing, go sightseeing*

Fig. 2 Error correction and feedback messages provided by the proposed method.

2. Error Case Frame

An error case frame consists of a verb, cases, and a feedback message as shown in Fig. 1[†]. The following explains error case frames in detail based on this example; occasionally consulting it may help understanding the following sections.

An error case frame always has a verb. In Fig. 1, the verb is *go*.

Cases are arguments the verb takes in an error case frame. A case consists of a case tag and case elements. A case tag and case elements describe, respectively, the role that the case plays in the error case frame and a set of words

that are allowed to appear as the argument. For instance, in Fig. 1, “Subj: {PERSON}” is a case where its case tag and element are “Subj:” and “{PERSON},” respectively, denoting that a person such as *John* plays a role of the subject of the verb. Note that tokens in all upper case such as “PERSON” refer to a group of words such as {john,he,...} in this paper.

Cases are classified into two categories: basic and preposition cases. Basic cases are either a subject or a particle, whose case tags are “Subj:” and “Ptr:”, respectively. The “Subj:” case is obligatory while the “Ptr:” is optional. Preposition cases correspond to the prepositions the verb takes as its arguments. Its case tag has the form of “Prep_x” where x ranges over the target prepositions. It should be emphasized that direct and indirect objects are included in the preposition cases for efficiency; their case tags are denoted as “Prep_dobj” and “Prep_iobj”, respectively. Preposition cases are classified into those obligatory and optional. *Optional* here means that the verb can constitute a sentence with or without the preposition. Optional prepositions are written in parentheses as in “(Prep_with:{family})”, which denotes the prepositional phrase *with family* may be omitted in the case frame.

Preposition cases describe the information about an error. An error case frame is constrained to contain only one erroneous preposition case. It is marked with the symbol “*”; the preposition case “*Prep_to:{store,market}” is erroneous in Fig. 1. The correct preposition is described after the symbol “→” as in “→ Prep_at”. Taken together, it reads “*to store or to market* is erroneous and *to* should be replaced with *at*.”

Error case frames are furnished with feedback messages. Unlike verbs and cases, which are automatically filled based on corpus data, they are manually edited. A human annotator interprets error case frames and adds explanatory notes to them. This may seem time-consuming. However, the editing is far more efficient than manually creating correction rules with feedback messages from scratch because error case frames are highly abstracted as explained in Sect. 3. Above all, it is a significant advantage over the previous classifier-/LM-based methods considering that there exists no effective technique for augmenting these methods with feedback messages.

[†]Fig. 1 shows an example of error case frames for illustration purposes. They are formally expressed in a machine-readable format such as XML.

3. Generating Error Case Frames

The method proposed here exploits two sources of corpus data: native and learner corpora. Case frames (error case frames without the information about an error and a feedback message) can be automatically extracted from parsed sentences as Kawahara and Uchimoto [21] show. The proposed method generates error case frames by comparing case frames generated from the learner corpus with those from the native corpus. The basic approach is to extract, as error case frames, case frames which appear in the learner corpus but not in the native corpus. However, this approach is so naive that it extracts undesirable false error case frames which do not actually correspond to preposition errors. To reduce such false error case frames, the following procedures are applied:

- (1) Filtering input sentences
- (2) Extracting case frames
- (3) Recognizing optional cases
- (4) Grouping case frames
- (5) Selecting candidate error case frames
- (6) Determining correct prepositions
- (7) Enriching error case frames
- (8) Manually editing error case frames

(1) Filtering input sentences: This is a pre-process to filter out unsuitable input sentences for case frame generation. Accurate parsing is essential for accurate case frame generation. Parsing errors tend to occur in longer sentences. To reduce parsing errors, Kawahara and Uchimoto [21] propose filtering out sentences which are longer than 20 words. We adopt this filtering in our method. We also filter out sentences containing commas, which often introduce complex structures. We apply the filtering pre-process only to the native corpus; the availability of learner corpora is still somewhat limited and therefore we use all the sentences available in the learner corpus for better coverage of preposition errors.

(2) Extracting case frames: This procedure can be viewed as a slot filling task where the slots are the verb and the cases in a case frame. To achieve this, the corpus data are first parsed by a parser. Then, for each verb, the predicate-argument structures are extracted from the parses as shown in Fig. 3. Here, only head words are extracted as arguments. They are reduced to their base form when extracted. Certain classes of words are replaced with their corresponding sense (e.g., *John* to *PERSON*); the mapping between words and their senses is shown in Appendix A. For the learner corpus, mis-spelt words are automatically corrected using a spell-checker. Finally, a case frame is created by filling its slots with the extracted predicate-argument structures. Hereafter, case frames generated from the native and learner corpora will be referred to as the native and learner case frames, respectively.

(3) Recognizing optional cases: it is crucial for generating flexible error case frames to recognize optional preposition cases. Optional preposition cases are determined by

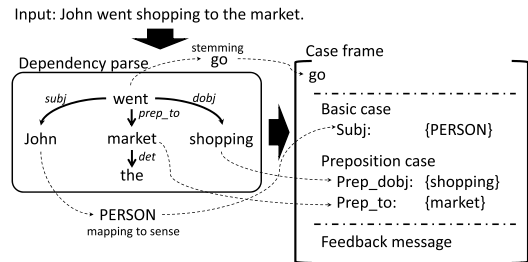


Fig. 3 Example of case frame extraction.

the following heuristic rules: (a) Objects are always obligatory; (b) The number of obligatory preposition cases (except objects) is at most one; (c) Prepositions appearing left of the verb are optional; (d) Prepositions appearing right of the verb are optional except the one which is nearest to the verb. Rule (a) states that objects are always recognized as obligatory[†]. Rule (b) constrains an error case frame to have at most one obligatory preposition. Certain verbs sometimes have more than one obligatory preposition as in *range from A to B*. However, the large majority of verbs satisfy rule (b). Rule (c) states that prepositions appearing left of the verb in the input sentence are optional preposition cases as in *In the morning, he went shopping*. Rule (c) is based on the assumption that obligatory cases are tied to the verb more strongly than optional cases. In other words, obligatory cases cannot easily change their position. Conversely, optional cases have more freedom of their position, which enables them to appear left of a verb. Admittedly, obligatory prepositions can appear left of a verb as in *To school, he went* in certain circumstances such as in poetry. However, this usage is not so frequent in corpora normally used as training data such as newspaper articles. Rule (d), together with rule (b), states that if more than one preposition appears right of the verb, the one nearest to the verb is obligatory and the rest are optional. Rule (d) is based on the same reasoning as in rule (c).

In addition to the above heuristics, optional preposition cases are sometimes determined naturally by comparing two case frames. In this case, one of them must consist of only the object(s) as its preposition case(s) as in “[go Subj:{PERSON} Prep_dobj:{shopping}]”. Then, the other case frame must consist of the same verb, the same basic cases, and the same object(s). The only difference between them is preposition cases (except the object(s)) (e.g., [go Subj:{PERSON} Prep_dobj:{shopping} Prep_at:{market}]). The case frame only with the object(s) proves the other to be valid without the preposition case(s). Thus, these preposition cases are recognized as optional (e.g., [go Subj:{PERSON} Prep_dobj:{shopping} (Prep_at:{market})]).

[†]A sentence can be constituted without objects as in *We sing*. Rule (a) always mistakenly recognizes such objects as obligatory. However, preposition errors never appear in sentences consisting of no object nor prepositions, and thus, the objects mistakenly recognized as obligatory never cause any problems in preposition error correction in practice.

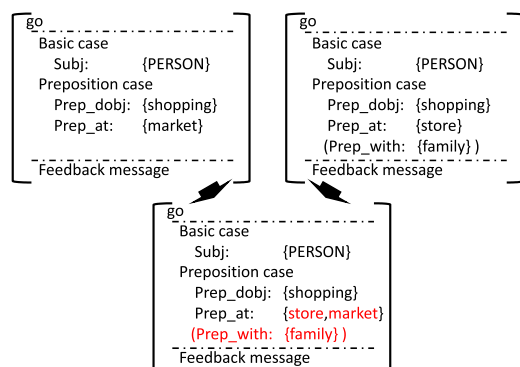


Fig. 4 Example of grouping case frames.

(4) Grouping case frames: Similar case frames in the native case frames are grouped into one, which will play an important role in **(7) Enriching error case frames**. Case frames comprising similar cases tend to denote similar usage of a verb. Considering this, case frames are merged into one if they consist of the same verb, the same basic cases, and the same case tags of the obligatory preposition cases. The grouping procedure is illustrated in Fig. 4. When preposition cases are obligatory in one case frame and optional in the other, the discrepancy is resolved by setting the preposition case to optional in the merged case frame. Note that this grouping procedure is not applied to the learner case frames so that erroneous usages in the learner case frames do not propagate to other (correct) learner case frames.

Because of the procedures so far, the extracted case frames, especially native case frames, are highly abstracted with optional cases and grouping. For instance, the grouped case frame shown in Fig. 4 is expanded to $2 \times 2 \times |\text{PERSON}|$ case frames where $|\text{PERSON}|$ denotes the number of words mapped to *PERSON*; if we set *PERSON* to personal pronouns, that is, $\text{PERSON} = \{\text{I, you, he, she, we, they}\}$, it is expanded to 24 case frames. In its actual use, the case frame covers the differences in the tense and aspect of the verb and the attachment of modifiers such as adjectives and adverbs; it corresponds to far wider expressions.

(5) Selecting candidate error case frames: Candidates for error case frames are selected from the learner case frames. If a learner case frame does not match, ignoring optional preposition cases, any native case frame, it is selected as a candidate for an error case frame on the assumption that case frames corresponding to erroneous usages do not appear in the native corpus.

Alternatively, an error-annotated learner corpus can be used to select error case frames; simply extracting case frames of which preposition is marked as an error gives error case frames. In this case[†], procedure (6) may be omitted and procedure (7) is directly applied after procedure (5).

[†]We do not make use of error-annotated learner corpora in this paper in order to reveal how well the proposed methods perform without such corpora. In practice, one can use error-annotated learner corpora together with raw learner corpora to achieve better performance.

(6) Determining correct prepositions: Now, correct prepositions for the candidate error case frames are explored. Each case tag of the preposition cases in a candidate is replaced, one at a time, with one of the other target prepositions. This replacement can be interpreted as error correction. Take as an example the following candidate error case frame: `[go Subj:{PERSON} Prep_dobj:{shopping} Prep_to:{market}]`. Replacing the case tag “Prep_to” with “Prep_at” corresponds to correct expressions such as *John often goes shopping at the market*. Note that replacing a direct object with one of the prepositions corresponds to correcting an omission error as in “Prep_dobj” with “Prep_to” in `[go Subj:{PERSON} Prep_dobj:{market}]`. Similarly, replacing a preposition with an object corresponds to correcting an extra-preposition error (e.g., “Prep_to” with “Prep_dobj” in `[go Subj:{PERSON} Prep_to:{shopping}]`).

To examine whether each correction is valid or not, the native case frames are again used; if the replaced case frame matches one of the native case frames, the correction is determined to be valid. Here, we define the match as the two case frames consisting of the same verb, the same basic cases, the same obligatory preposition cases, and the same preposition case to which the correction is applied (if it is an optional one). If the condition is satisfied, the information on the error and correction is added to the candidate error case frame. If a valid correction is found, the candidate is determined to be a valid error case frame. In total, their validity is double-checked, once in (5) and once in (6), by comparing them with the native case frames.

(7) Enriching error case frames: The generated error cases are limited in error coverage because the procedures so far solely rely on preposition errors appearing in the learner corpus. In other words, it is impossible to generate error case frames corresponding to preposition errors which do not appear in the learner corpus. To overcome this limitation, the generated error case frames are enriched using the native case frames. For each error case frame, we already know the corresponding native (thus, correct) case frame, which is obtained in (6). The corresponding native case frame is normally much richer in preposition cases because of the optional cases and grouping given by procedures (3) and (4), as shown at the top of Fig. 5. These additional cases are useful to enrich error case frames.

For the preposition case which is determined to be erroneous, its correct preposition is found in the error case frame (e.g., “→ Prep_at” at the top-left of Fig. 5). Also, its correct preposition case is found in the corresponding native case frame (e.g., “Prep_at:{market,store}” at the top-right). Replacing the case element of the erroneous case by one of the case elements of the correct preposition case gives a new candidate for an error case frame (e.g., replacing *market* of “*Prep_to:{market}” by *store* gives `[go Subj:{PERSON} Prep_dobj:{shopping} *Prep_to:{store}]`). It should be emphasized that this new error case frame is still a candidate at this point and the usage might be correct. To verify if it really describes an erroneous preposition use, the native

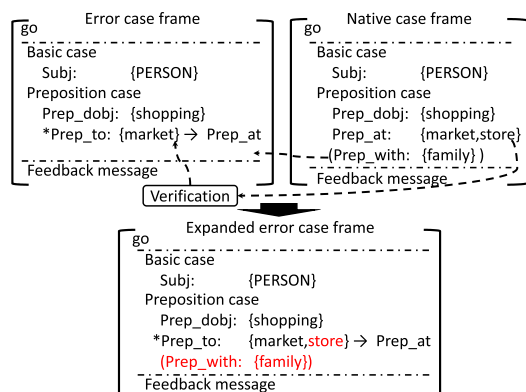


Fig. 5 Enriching an error case frame.

case frames are searched for; if it matches one of them, that means that the use of the preposition actually appears in the native corpus. Therefore, it should be discarded. Only if a match is not found, is the case element added to the erroneous preposition case in the original error case frame. This process is illustrated in the box denoted as *Verification* in Fig. 5.

For the other preposition cases which are not erroneous, the enriching procedure is much simpler. They are simply added to the error case frame as shown in Fig. 5. One thing we should take care of is that there might be a discrepancy in obligatory/optional between the cases of the error case frame and the native case frame. This discrepancy is solved by setting the preposition case in the error case frame to optional. The resulting expanded error case frame after procedure (7) is shown at the bottom of Fig. 5 where the enriched cases are shown in red.

These are all the procedures needed to automatically generate error case frames. As has been shown in this section, the requirements as language resources are raw native and learner corpora. In other words, the proposed method does not require error annotated corpora, which is one of its advantages over the previous classifier-based methods requiring them.

(8) Manually Editing Error Case Frames: The most important editing is the addition of feedback messages. A human annotator interprets the generated error case frames and adds explanatory notes to them. Although this basically requires manual editing, part of feedback messages can be automatically created to facilitate the procedure. For example, example sentences corresponding to an error case frame can be automatically added to it, whether correct or error examples, because the original sentences from which the (error) case frames are extracted are available in the native and learner corpora. Besides, setting a variable to the feedback message allows it to be adaptable to correction results as shown in Fig. 6. In Fig. 6, $X_{\text{Prep_to}}$ is a variable. It is replaced with one of the case elements of “Prep_to:” depending on correction results. Also, it will be beneficial to link similar error case frames each other, which allows the user to obtain additional information. For

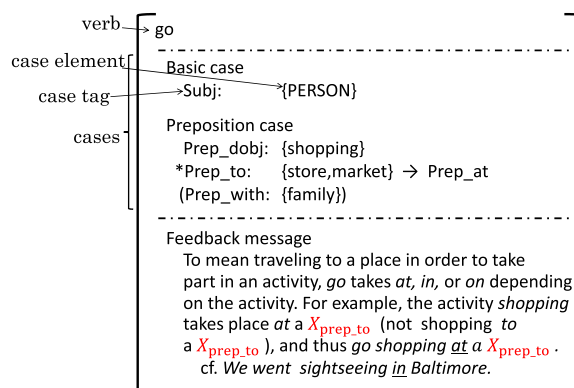


Fig. 6 Error case frame with a variable.

example, the example error case frame in Fig. 6 may be linked to similar case frames such as “[go Subj:{PERSON} Prep_dobj:{sightseeing} *Prep_to:{Baltimore} → Prep_in].” One can retrieve similar error case frames from the generated error case frames where the similarity between two error case frames are defined by the overlap in the verb, the basic cases, and the case tags of the preposition cases.

The generated error case frames may be further edited to enrich them. As we can see in Fig. 5, the generated error case frames are easy to interpret. This property enables us to manually edit them to enrich their preposition cases. For example, one might add a case element such as *supermarket* to the preposition case “Prep_to:{market,store}” in the example error case frame. Conversely, one might discard unnecessary case elements, cases, or even error case frames.

4. Correcting Preposition Errors

Preposition errors are corrected by applying the generated error case frames to the target text. Case frames are first extracted from the target text by the same procedures (2) and (3) in Sect. 3. Then, each extracted case frame is examined if it matches one of the error case frames. If a match is found, the preposition is detected as an error and the correct preposition is suggested with the feedback message according to the matched error case frame. The match between a case frame and an error case frame is defined in the exact same manner as in procedure (4) in Sect. 3. Sometimes, a case frame matches more than one error case frame suggesting different corrections. In this case, the most frequent correction among the candidates is chosen to correct the error, which was applied in the evaluation described in Sect. 5[†]. Alternatively, we propose suggesting all available corrections with the feedback messages and letting the learner select the appropriate correction. This way of correction may give learners better learning effects than just displaying a correct preposition considering that the selection process helps the learner think deeply.

One of the advantages of error case frames is that they do not require an error-annotated corpus as explained in

[†]Ties were broken by random selection.

the previous section. This means that the target text itself can be used as part of a learner corpus for generating error case frames at the time of error correction. Applying procedures (2) to (7) to the target text generates additional error case frames[†]. Although feedback messages are not available in these additional error case frames, they are still useful for improving correction performance, especially in recall. Hereafter, this way of error case frame generation will be referred to as *active generation*.

A pre-experiment using a development data set revealed that there were some preposition errors for which error case frames were not generated even though the corresponding erroneous and correct preposition usages appeared in the learner and native corpora, respectively. They are preposition errors where the preposition is incorrectly used with an adverb as in **John went to there*. To be precise, they are either an adverb denoting a place (e.g., *there*) with a preposition concerning a place (*at*, *in*, *on*, and *to*) or a noun denoting time, frequency, and duration with a preposition concerning time, frequency, and duration (*at*, *for*, *in*, and *on*). In the native corpus, these adverbs or nouns are correctly used without a preposition and thus they are not recognized as a prepositional phrase by a parser. Therefore, corresponding native case frames are never found for these types of errors in procedure (6), and in turn error case frames are never generated for them.

Considering that they are limited in number because they are independent of verbs and basic cases, we decided to manually create error case frames describing these types of errors. In these error case frames, the verb and the basic cases are filled with ANY denoting any word. The preposition cases are manually filled based on the linguistic knowledge known as *absence of preposition* [22]. For example, an error case frame for the above error would be “[ANY Subj:{ANY} *Prep_to:{here,somewhere,there} → Prep_dobj].” Certain errors involve a phrase such as **John goes shopping in every morning*. To handle these cases, these manually created error case frames are allowed to have phrases as their case elements (e.g., [ANY Subj:{ANY} *Prep_in:{every morning} → Prep_dobj].

5. Evaluation

We evaluated the proposed method from two points of view: correction performance and usefulness of feedback messages. We measured correction performance by recall, precision, and *F*-measure. In the evaluation on usefulness of feedback messages, three human raters (a teacher of English at college and two who have a master degree in TESOL) separately examined whether each feedback message was useful for learning the correct usage of the preposition. We defined usefulness by the ratio of feedback messages evaluated as useful to the total number of feedback messages.

The aim of the usefulness evaluation is to evaluate (i) whether or not human raters can interpret the given error

Table 2 Statistics on the data sets for evaluation.

Name	# of tokens	# of errors
KJ training	22,701	327
KJ test	8,065	131
Dev. set	47,217	774
EDR	1,745,863	—
Reuters	28,431,228	—
LOCNESS	294,325	—

case frames and edit their feedback messages properly (as described in Sect. 3, error case frames are highly abstracted and this might make human raters misinterpret them); (ii) whether or not feedback messages are appropriate in the contexts where they are shown (they might be inappropriate due to other grammatical errors appearing around the corrected preposition).

We used the following data sets in the evaluation. We selected the Konan-JIEM (KJ) learner corpus [23] as the target texts. The KJ learner corpus is fully annotated with grammatical errors. In addition, it includes error correction results of several benchmark systems. This means that one can directly compare correction results of a new method with those of the benchmark systems, which reveals where the method is strong and weak compared to the benchmark systems. The KJ corpus consists of training and test sets. We used the training set to generate error case frames and evaluated correction performance on the test set. In addition to these data sets, we created a development set, which we had collected to develop the proposed method. We did not use it in the final evaluation. As a native corpus, we used the EDR corpus [24], the Reuters-21578 corpus^{††}, and the LOCNESS corpus^{†††}. We used the lexicalized dependency parser in the Stanford Statistical Natural Language Parser (ver.2.0.3) [25] to obtain parses for the data sets. Table 2 shows the statistics on the data sets.

Using these data sets, we implemented three versions of the proposed method. The first one was based on error case frames generated from the training set of the KJ corpus. The second one was the first one with active generation. To implement the third one, we manually edited the error case frames of the first version to remove unnecessary error case frames and case elements (but no addition) and to add feedback messages to them. After this, active generation was applied to augment the edited error case frames. In implementing the proposed methods, we selected as target prepositions the ten most frequent prepositions, the same as in previous work [4]: *about*, *at*, *by*, *for*, *from*, *in*, *of*, *on*, *to*, *with*.

For comparison, we selected two previous methods. One was the best-performing system among the benchmark systems, which is the classifier-based method [20] which had participated in the HOO 2012 shared task [26]. The other was the SMT-based method [13] which was the best-performing system in preposition error correction in the

^{††}Reuters-21578, Distribution 1.0, <http://www.research.att.com/~lewis>

^{†††}<http://www.uclouvain.be/en-cecl.html>

[†]Recall that procedure (1) is only applied to the native corpus.

Table 3 Correction performance in recall (R), precision (P), and F -measure (F).

Method	R	P	F
ECF	0.107	0.823	0.189
ECF with AG	0.130	0.680	0.218
ME-ECF with AG	0.130	0.708	0.219
Classifier-based	0.167	0.310	0.217
SMT-based	0.115	0.385	0.176
Classifier hybrid	0.235	0.369	0.287
SMT hybrid	0.191	0.446	0.267

ECF: Error Case Frame, ME-ECF: Manually Edited Error Case Frame, AG: Active Generation

Table 4 The number of (error) case frames generated in each procedure.

Procedure	Number of (error) case frames
Step (2)	1,519
Step (4)	1,089
Step (6)	46
Step (8)	35
Expanded error case frames	31×10^{15}

CoNLL 2013 shared task [1]. In addition, we evaluated performance of hybrid methods combining the correction results of the third version of the proposed method with those of the classifier-/SMT-based method; we simply took the union of the two.

Table 3 shows the evaluation results. The simple error case frame-based method achieves an F -measure of 0.189. It improves recall when combined with active generation, which shows the effectiveness of active generation for augmenting error case frames. It further improves precision without decreasing recall by manual editing; note that manual editing was only applied to the error case frames generated from the training data but not to those generated by active generation. The performance is comparable to both classifier-/SMT-based methods. The hybrid methods achieve the best performances in F -measure.

Table 4 shows the number of (error) case frames generated in each procedure (of the third version of the proposed method). Each step in the table corresponds to the procedures in Sect. 3. The row *Expanded error case frames* shows the theoretical number of error case frames when the manually edited error case frames are expanded to a single error case frame, which shows that the proposed method can generate a large number of correction rules even from a small learner corpus such as KJ[†].

In the usefulness evaluation, the third version of the proposed method was able to provide 20 feedback messages for the target texts. The three human raters evaluated 80%, 80%, and 85% of the 20 feedback messages as useful (82% on average). The agreement among the raters was $\kappa = 0.67$ in Fleiss's κ .

6. Discussion

As the experimental results show, the proposed method

[†]Note that not all rules correspond to correct English sentences and might never be used in the correction procedure.

achieves a comparable correction performance with the classifier-/SMT-based methods. A closer look at the correction results reveals the differences in correction tendencies between these methods, which explains well why the hybrid methods achieve better performance.

One of the tendencies is that the proposed method performs better on preposition errors where relatively wider contexts are required to correct them. Error case frames naturally exploit wider contexts based on the cases which are extracted by parsing. In contrast, classifier-/SMT-based methods rely on narrower contexts such as a few words surrounding the preposition in question. Take as an example the following sentence which appeared in the test set: **In the univarsity, I studied English in the morning^{††}*. To confirm that the preposition *In* is erroneous requires the verb *studied* and the object *English*. The proposed method successfully corrected this error by the error case frame “[study Subj:{PERSON} Prep_dobj:{english,math,...} *Prep_in:{university} → at]” in the evaluation. This would be difficult for methods relying on only a few words surrounding the preposition *In*.

It is also difficult for classifier-/SMT-based methods to correct missing preposition errors. Classifier-based methods need to be informed of the position of the preposition to predict a correct preposition. Because the position of a missing preposition is implicit, classifier-based methods would have to make a prediction at every single position between words, which would be inefficient. Because of this, the classifier-based method used in the evaluation (and often other classifier-based methods) excludes missing preposition errors from its target. SMT-based methods do not perform well either on missing preposition errors because of the fact that they implicitly, but not directly, handle missing preposition errors. In contrast, error case frames directly model missing prepositions by treating objects as one of the preposition cases (i.e., *Prep_dobj*).

Grammatical errors other than preposition errors influence both the proposed and classifier-/SMT-based methods, but differently. Grammatical errors appearing around the preposition in question seem to influence the previous methods more significantly than the proposed method because they rely on words surrounding the preposition. On the other hand, structural errors such as errors in voice tend to degrade performance of the proposed method. For instance, if an error in voice occurs as in **I excited _ this*, correctly, *I was excited by this*, error case frames are not properly applied.

The precisions of the proposed methods are high compared to those of the previous methods. To be precise, the number of false positives is only seven in the third version of the proposed method. Out of seven, four false positives are due to problems with the used error case frames themselves. Two are the influence of other grammatical errors (e.g., **I like to look beautiful view*. was corrected as *look at*

^{††}The word *university* is a mis-spelt word of *university*. Note that mis-spelt words are automatically corrected by a spell-checker when case frames are extracted.

beautiful view by the proposed method but as *see beautiful view* in the error annotation).

Unlike false positives, it is difficult to precisely point out causes for false negatives, which often involve several factors. One cause which is theoretically clear is errors in preposition attached to an NP, which amounts to 11% of all false negatives. Since error case frames describe errors in preposition attached to a verb, they do not target these types of errors. Extending error case frames to general frames might overcome this limitation, which will require further investigation. Similarly, error case frames are not generated for preposition errors where prepositions are incorrectly used with words other than a noun as in **make me to happy* (5% of all). Although error case frames can describe these types of errors, case frames are not extracted for their corresponding correct usages from the native corpus. This is because the word in question (e.g., *happy*) correctly appears without the erroneous preposition in the native corpus, and thus it is not recognized as a preposition case. This means that a corresponding correct case frame is never found for any error of these types in the generation procedure (6). Accordingly, error case frames are never generated for these types of errors. The most influential cause of false negatives, which is also a major cause of false negatives in the previous methods, is other grammatical errors (at least 22% of all). One such error is errors in voice as already explained (4%). Another is the omission of the object of a verb (4%). In these cases, even if an appropriate error case frame exists, it is not applied because of the grammatical error.

Because the proposed method relies on a parser to extract case frames from a corpus, parsing accuracy affects its correction performance. Berzak et al. [27] show that a parser developed for native English performs slightly worse on learner English than on native English (label attachment score for the former and the latter are 86.3% and 88.3%, respectively). Tetreault et al. [7] show that it can reliably extract parse features for grammatical error correction from learner English. These previous findings suggest that present parsers are effective in extracting case frames to some extent. In our evaluation, parsing errors did not cause false positives but did cause some of the false negatives. A standard parser is often not able to extract the parse features needed to generate a case frame from a sentence containing a grammatical error. Such errors include prepositions that are incorrectly used with words other than a noun, errors in voice, and omissions of the object, which amounts to 13% of all negatives in the evaluation as described above.

While the error case frame/classifier-based hybrid method achieves the best performance in *F*-measure, there is still plenty of room for improvement, especially in recall. The fact that case elements are compared in a complete-match manner in both generation and correction processes reduces its generalization power, which results in low recall. In other words, errors are corrected only if an error case frame exactly matches the erroneous instance. For example, the error *he went shopping *to the supermarket.* is

not corrected even if there exists a very similar error case frame such as “[go Subj:{PERSON} Prep_dobj:{shopping} Prep_at:{market,store,shop}]”. To overcome the drawback, the comparison should be done in a more abstract manner. One way of achieving this is to use the distributed representations of words [28], [29]. In them, case elements can be represented more abstractly as vectors and those having similar sense, such as *market* and *supermarket*, are regarded as similar case elements. This should increase the generalization power in error correction process based on error case frames.

In practice, correction performance, especially recall, of the proposed method needs to be improved in order to provide a learning effect to learners of English; it can detect and correct a limited number of preposition errors. This is also one of the major problems in all previous methods including the classifier-based and SMT-based methods. One advantage of the proposed method is that its precision is very high compared to the previous methods. According to Nagata et al. [30], precision-oriented methods are preferable in terms of learning effect. They show that a method for detecting article errors has a learning effect when precision and recall are 0.72 and 0.25, respectively. Although the direct comparison is difficult because the error rates of article errors and preposition errors are different (normally, article errors are more frequent), it is still preferable to have a method with a high precision.

Besides correction performance, error case frames are effective in providing feedback messages; Fig. 2 (in Sect. 1) shows excerpts of the feedback messages provided in the evaluation. The evaluation shows that 82% of the provided feedback messages were actually rated as useful for language learning on average (the rest were mostly evaluated as not-useful due to false positive corrections). With the feedback messages of error case frames, we now have the following three choices as the way of error correction: (a) just indicating the correct preposition (as in previous methods); (b) indicating the correct preposition with a feedback message; (c) displaying only a feedback message. In (a), the learner might just copy the correct preposition to correct his or her writing, which would result in little or no learning effect. This suggests that the ultimate goal of grammatical error correction for language learning assistance is not to correct all errors in the given text but to maximize learning effect for the learner. (b) might give a similar result because the learner can copy the correct preposition without reading the feedback message. In (c), the learner has to actually read and understand the feedback message to select the correct preposition. Taking these into consideration, (c) will likely give the learner better learning effect than the other two. Therefore, we propose applying the feedback (c) to language learning assistance. To the best of our knowledge, it is only the error case frame-based method that is capable of this manner of error correction.

7. Conclusions

This paper presented a novel framework called error case frames for correcting preposition errors with feedback messages. The evaluation showed that (i) automatically generated error case frames achieve a performance comparable to previous methods; (ii) they are intuitively interpretable and manually modifiable to improve them; (iii) feedback messages provided by error case frames are effective in language learning assistance.

Acknowledgments

This work was partly supported by Grant-in-Aid for Young Scientists (B) Grant Number JP26750091.

References

- [1] H.T. Ng, S.M. Wu, Y. Wu, C. Hadiwinoto, and J. Tetreault, "The CoNLL-2013 shared task on grammatical error correction," Proc. 17th Conference on Computational Natural Language Learning: Shared Task, Sofia, Bulgaria, pp.1–12, Aug. 2013.
- [2] N.-R. Han, M. Chodorow, and C. Leacock, "Detecting errors in English article usage by non-native speakers," *Natural Language Engineering*, vol.12, no.2, pp.115–129, June 2006.
- [3] R. Nagata, K. Morihiro, A. Kawai, and N. Isu, "A feedback-augmented method for detecting errors in the writing of learners of English," Proc. 44th Annual Meeting of Association for Computational Linguistics, Sydney, Australia, pp.241–248, July 2006.
- [4] A. Rozovskaya and D. Roth, "Algorithm selection and model adaptation for ESL correction tasks," Proc. 49th Annual Meeting of the Association for Computational Linguistics, Portland, USA, pp.924–933, June 2011.
- [5] M. Chodorow, J.R. Tetreault, and N.-R. Han, "Detection of grammatical errors involving prepositions," Proc. 4th ACL-SIGSEM Workshop on Prepositions, Prague, Czech Republic, pp.25–30, June 2007.
- [6] R.D. Felice and S.G. Pulman, "A classifier-based approach to preposition and determiner error correction in L2 English," Proc. 22nd International Conference on Computational Linguistics, Manchester, UK, pp.169–176, Aug. 2008.
- [7] J. Tetreault, J. Foster, and M. Chodorow, "Using parse features for preposition selection and error detection," Proc. 48th Annual Meeting of the Association for Computational Linguistics Short Papers, Uppsala, Sweden, pp.353–358, July 2010.
- [8] R. Nagata and A. Kawai, "Exploiting learners' tendencies for detecting English determiner errors," *Lecture Notes in Computer Science*, Kaiserslautern, Germany, vol.6882, pp.144–153, Sept. 2011.
- [9] T. Tajiri, M. Komachi, and Y. Matsumoto, "Tense and aspect error correction for ESL learners using global context," Proc. 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Jeju, Republic of Korea, pp.198–202, July 2012.
- [10] Y. Wu and H.T. Ng, "Grammatical error correction using integer linear programming," Proc. 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, pp.1456–1465, Aug. 2013.
- [11] A. Rozovskaya and D. Roth, "Joint learning and inference for grammatical error correction," Proc. Conference on Empirical Methods in Natural Language Processing, Seattle, USA, pp.791–802, Oct. 2013.
- [12] C. Brockett, W.B. Dolan, and M. Gamon, "Correcting ESL errors using phrasal SMT techniques," Proc. 21th International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, pp.249–256, July 2006.
- [13] I. Yoshimoto, T. Kose, K. Mitsuzawa, K. Sakaguchi, T. Mizumoto, Y. Hayashibe, M. Komachi, and Y. Matsumoto, "NAIST at 2013 CoNLL grammatical error correction shared task," Proc. 17th Conference on Computational Natural Language Learning: Shared Task, Sofia, Bulgaria, pp.26–33, Aug. 2013.
- [14] D. Ferris and B. Roberts, "Error feedback in L2 writing classes: How explicit does it need to be?," *Journal of Second Language Writing*, vol.10, no.3, pp.161–184, Aug. 2001.
- [15] J. Kakegawa, H. Kanda, E. Fujioka, M. Itami, and K. Itoh, "Diagnostic processing of Japanese for computer-assisted second language learning," Proc. 38th Annual Meeting of the Association for Computational Linguistics, Hong Kong, China, pp.537–546, Oct. 2000.
- [16] K.F. McCoy, C.A. Pennington, and L.Z. Suri, "English error correction: A syntactic user model based on principled "mal-rule" scoring," Proc. 5th International Conference on User Modeling, Kailua-Kona, USA, pp.69–66, Jan. 1996.
- [17] R. Nagata, M. Vilenius, and E. Whittaker, "Correcting preposition errors in learner English using error case frames and feedback messages," Proc. 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, USA, pp.754–764, June 2014.
- [18] A. Rozovskaya, K.W. Chang, M. Sammons, and D. Roth, "The university of illinois system in the CoNLL-2013 shared task," Proc. 17th Conference on Computational Natural Language Learning: Shared Task, Sofia, Bulgaria, pp.13–19, Aug. 2013.
- [19] R. Nagata and A. Kawai, "A method for detecting determiner errors designed for the writing of non-native speakers of english," *IEICE Transactions on Information and Systems*, vol.E95-D, no.1, pp.230–238, Jan. 2012.
- [20] K. Sakaguchi, Y. Hayashibe, S. Kondo, L. Kanashiro, T. Mizumoto, M. Komachi, and Y. Matsumoto, "NAIST at the HOO 2012 shared task," Proc. 7th Workshop on the Innovative Use of NLP for Building Educational Applications, Montreal, Canada, pp.281–288, June 2012.
- [21] D. Kawahara and K. Uchimoto, "A method for automatically constructing case frames for English," Proc. 6th International Conference on Language Resources and Evaluation, Marrakech, Morocco, pp.3314–3317, Jan. 2008.
- [22] R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik, *A Comprehensive Grammar of the English Language*, Longman, New York, 1985.
- [23] R. Nagata, E. Whittaker, and V. Sheinman, "Creating a manually error-tagged and shallow-parsed learner corpus," Proc. 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, USA, pp.1210–1219, June 2011.
- [24] Japan electronic dictionary research institute Ltd, *EDR electronic dictionary specifications guide*, Japan electronic dictionary research institute Ltd, Tokyo, 1993.
- [25] M.C. de Marneffe, B. MacCartney, and C.D. Manning, "Generating typed dependency parses from phrase structure parses," Proc. 5th International Conference on Language Resources and Evaluation, Genoa, Italy, pp.449–445, May 2006.
- [26] R. Dale, I. Anisimoff, and G. Narroway, "HOO 2012: A report on the preposition and determiner error correction shared task," Proc. 7th Workshop on Building Educational Applications Using NLP, Montreal, Canada, pp.54–62, May 2012.
- [27] Y. Berzak, J. Kenney, C. Spadine, J.X. Wang, L. Lam, K.S. Mori, S. Garza, and B. Katz, "Universal dependencies for learner English," Proc. 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, pp.737–746, Aug. 2016.
- [28] T. Mikolov, W.t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," Proc. 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Atlanta, USA, pp.746–751, June 2013.
- [29] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, and J. Dean, "Dis-

tributed representations of words and phrases and their compositionality,” *Advances in Neural Information Processing Systems* 26, Lake Tahoe, USA, pp.3111–3119, Dec. 2013.

- [30] R. Nagata and K. Nakatani, “Evaluating performance of grammatical error detection to maximize learning effect,” *Proc. 23rd International Conference on Computational Linguistics*, poster volume, pp.894–900, 2010.
- [31] G.A. Miller, “WordNet: A lexical database for English,” *Communications of the ACM*, vol.38, no.11, pp.39–41, Nov. 1995.

Appendix: Sense Mapping

The following list shows the mapping between words and senses developed based on the WordNet [31] and GSK dictionary of places and facilities (2nd Ed.)[†]. Each line consists of a token for a sense, its definition, examples of its members.

DRINK (drink): tea, coffee

FOOD (food): cake, sandwich

MONTH (names of months): January, February

MINST (musical instruments): guitar, piano

PERSON (persons): John, he

PLACE (place names): Canada, Paris

SPORT (sports): football, tennis

SPORTING (sporting activities): swimming

WEEK (the days of the week): Monday

VEHICLE (vehicles): train, bus



Edward Whittaker graduated from the Department of Electrical Engineering, Nottingham Univ. in 1996 and completed his doctoral thesis on statistical language modelling for speech recognition in the Engineering Department of Cambridge Univ. in 2000. He was then a visiting researcher at Compaq Cambridge Research Labs, Mass. and Philips Research Labs in Aachen, Germany before moving to Tokyo Institute of Technology, Japan. For the past eight years he has been the CEO of the speech recog-

nition and natural language processing company Inferret, as well as a member of Yahoo! Japan’s research lab. His research interests include language modelling, speech recognition and machine learning. He is a member of the IEEE.



Ryo Nagata graduated from the Department of Electrical Engineering, Meiji Univ. in 1999 and completed the doctoral program in information engineering at Mie Univ. in 2005 and became a research associate at Hyogo Univ. of Teacher Education. Since 2008, he has been an associate professor at Konan Univ. His research interests are language modeling, language acquisition, and grammatical error detection and correction. He is a member of the Association for Natural Language Processing.

[†]GSK dictionary of places and facilities second edition:
<http://www.gsk.or.jp/en/catalog/gsk2012-c/>