Correcting Preposition Errors in Learner English Using Error Case Frames and Feedback Messages

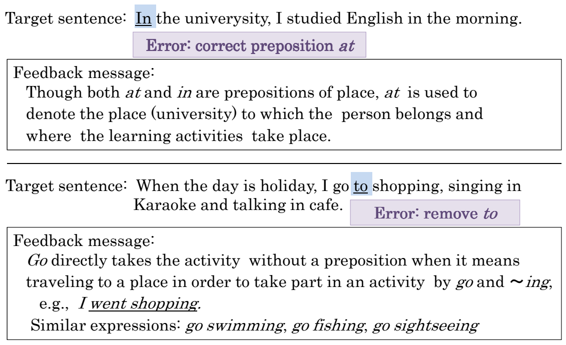
Ryo Nagata[1*] **Mikko Vilenius**[2] **Edward Whittaker**[3]
[1]Konan University / Kobe, Japan
[2]The Japan Institute for Educational Measurement, Inc. / Tokyo, Japan
[3]Inferret Limited / Northampton, England
nagata-acl@hyogo-u.ac.jp.

Abstract

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 conventional 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 by automatically generated rules, error case frames will likely be one of the major approaches for preposition error correction.

1 Introduction

This paper presents a novel framework for correcting preposition errors. Its most significant advantage over previous methods is that it 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. 1. Despite the fact that appropriate feedback messages are essential in language learning assistance (Ferris and Roberts, 2001; Robb et al., 1986), which is one of the immediate applications of grammatical error correction, almost all previous methods are incapable of providing feedback messages.


Target sentence: In the universysity, 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*

Figure 1: Error correction and feedback messages provided by the proposed method.

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 (Han et al., 2006; Nagata et al., 2006; Rozovskaya and Roth, 2011), preposition (Chodorow et al., 2007; Felice and Pulman, 2008; Rozovskaya and Roth, 2011; Tetreault et al., 2010), and tense (Nagata and Kawai, 2011; Tajiri et al., 2012), to name a few. Recently, Wu and Ng (2013) and Rozovskaya and Roth (2013) proposed methods for simultaneously correcting multiple types of errors using integer linear programming. Another major approach is to use a language model (LM) for predicting correct words or phrases for a given context. Some researchers (Brockett et al., 2006; Yoshimoto et al., 2013) 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.

As noted above, however, one of the crucial limitations of these previous methods is that they

[*]Part of this work was performed while the author was a visiting researcher at LIMSI, Orsay (France).

are not capable of providing feedback messages. They are not suitable for generating open-class text such as feedback messages by their nature. Some researchers (Kakegawa et al., 2000; McCoy et al., 1996) 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. 2. They are case frames specially designed for describing and correcting errors in preposition attached to a verb; the reader may be able to see that it describes preposition errors such as **John often goes shopping* *<u>to</u>* *the market with his family.* and that the preposition *to* should be replaced with *at*. This paper proposes 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 conventional 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. Sect. 2 introduces the definition of error case frames. Sect. 3 discusses the method for generating error case frames. Sect. 4 describes how to correct preposition errors with feedback messages by error case frames. Sect. 5 describes experiments conducted to evaluate error case frames. Sect. 6 discusses the experimental results.

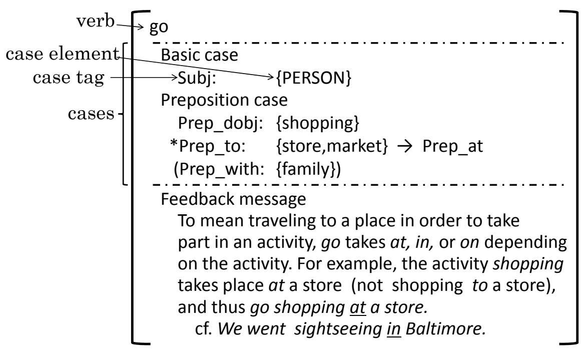
2 Error Case Frame

An error case frame consists of a verb, cases, and a feedback message as shown in Fig. 2[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. 2, 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

[1]Fig. 2 shows an example of error case frames for illustration purposes. They are formally expressed in a machine-readable format such as XML.



To mean traveling to a place in order to take part in an activity, *go* takes *at, in,* or *on* depending on the activity. For example, the activity *shopping* takes place *at* a store (not shopping *to* a store), and thus *go shopping <u>at</u> a store.*
cf. *We went sightseeing <u>in</u> Baltimore.*

Figure 2: Example of an error case frame.

to appear as the argument. For instance, in Fig. 2, "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,$\cdots$} 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})".

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 "*". So, the preposition case "*Prep_to:{store,market}" is erroneous in Fig. 2. The correct preposition is described after the symbol "→" as in "→ Prep_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.

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 (2008) 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 simple that it extracts undesirable false error case frames which do not actually correspond to preposition errors. To overcome the problem, 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 (2008) 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. In the case of 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.

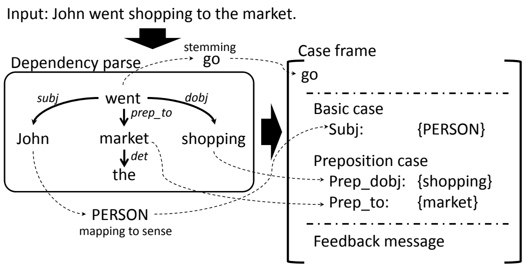


Figure 3: Example of case frame extraction.

(3) Recognizing optional cases: it is crucial for generating flexible error case frames to recognize optional preposition cases. Optional preposition cases are determined by 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[2]. 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

[2]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.

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).

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})]).

(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.

(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[3], procedure (6) may be omitted and procedure (7) is directly applied after procedure (5).

(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

[3]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.

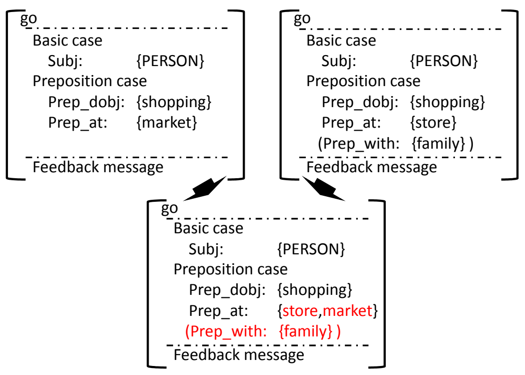


Figure 4: Example of grouping case frames.

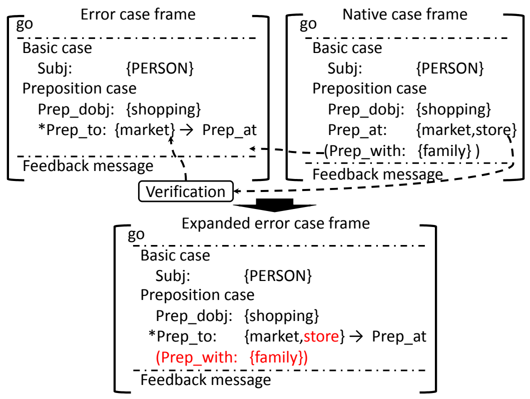


Figure 5: Enriching an error case frame.

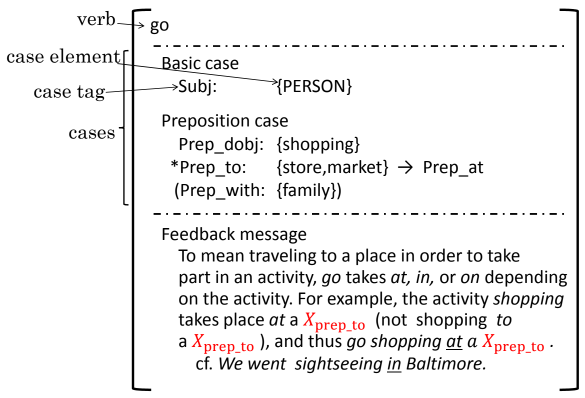
native case frames.

(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., "$\rightarrow$ 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 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.

(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 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.



To mean traveling to a place in order to take part in an activity, *go* takes *at, in,* or *on* depending on the activity. For example, the activity *shopping* takes place *at* a $X_{\text{prep\_to}}$ (not shopping *to* a $X_{\text{prep\_to}}$), and thus *go shopping at a* $X_{\text{prep\_to}}$. cf. *We went sightseeing in Baltimore.*

Figure 6: Error case frame with a variable.

For 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[4].

One of the advantages of error case frames is that they do not require an error-annotated corpus as explained in 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[5]. 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 <u>to there</u>*. 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* (Quirk et al., 1985). 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 <u>in every morning</u>*. To handle these cases,

[4]Ties are broken by random selection.

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

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.

We used the following data sets in the evaluation. We selected the Konan-JIEM (KJ) learner corpus (Nagata et al., 2011) 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 (Japan electronic dictionary research institute Ltd, 1993), the Reuters-21578 corpus[6], and the LOCNESS corpus[7]. We used the lexicalized dependency parser in the Stanford Statistical Natural Language Parser (ver.2.0.3) (de Marneffe et al., 2006) to obtain parses for the data sets. Table 1 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 (Rozovskaya and Roth, 2011): *about*, *at*, *by*, *for*, *from*, *in*, *of*, *on*, *to*, *with*.

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	—

Table 1: Statistics on the data sets for evaluation.

For comparison, we selected two conventional methods. One was the best-performing system among the benchmark systems, which is the classifier-based method (Sakaguchi et al., 2012) which had participated in the HOO 2012 shared task (Dale et al., 2012). The other was the SMT-based method (Yoshimoto et al., 2013) which was the best-performing system in preposition error correction in the CoNLL 2013 shared task (Ng et al., 2013). 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 2 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.

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

[6]Reuters-21578, Distribution 1.0, `http://www.research.att.com/~lewis`

[7]`http://www.uclouvain.be/en-cecl.html`

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 2: Correction performance in recall (R), precision (P), and F-measure (F).

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 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: **<u>In</u> the univerysity, I studied English in the morning*[8]. 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,$\cdots$} *Prep_in:{university} $\rightarrow$ 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 <u>look beautiful view</u>.* was corrected as *look <u>at</u> beautiful view* by the proposed method but as *<u>see</u> 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 a noun phrase (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

[8]The word *univerysity* is a mis-spelt word of *university*. Note that mis-spelt words are automatically corrected by a spell-checker when case frames are extracted.

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 of such errors 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.

In addition to correction performance, error case frames are effective in providing feedback messages; Fig. 1 (on the first page) 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 correction 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 conventional 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. Considering these advantages and the fact that it has been difficult to provide feedback messages by automatically generated rules, error case frames will likely be one of the major approaches for preposition error correction.

Appendix A. Sense mapping

The following list shows the mapping between words and senses developed based on the WordNet (Miller, 1995) and GSK dictionary of places and facilities (2nd Ed.)[9]. Each line consists of a token for a sense, its definition, examples of its member. 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

Acknowledgments

We would like to thank Daisuke Kawahara for his advice on case frame generation. We also would like to thank Keisuke Sakaguchi and Mamoru Komachi for providing the authors with their system outputs. Finally, we would acknowledge the help from the members of the ILES group at LIMSI, Orsay (France) where the first author performed part of this work. This work was partly supported by Kaken Grant-in-Aid for Young Scientists (B) (26750091).

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

References

Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting ESL errors using phrasal SMT techniques. In *Proc. of 21th International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 249–256, Sydney, Australia, July.

Martin Chodorow, Joel R. Tetreault, and Na-Rae Han. 2007. Detection of grammatical errors involving prepositions. In *Proc. of 4th ACL-SIGSEM Workshop on Prepositions*, pages 25–30.

Robert Dale, Ilya Anisimoff, and George Narroway. 2012. HOO 2012: A report on the preposition and determiner error correction shared task. In *Proc. 7th Workshop on Building Educational Applications Using NLP*, pages 54–62.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. of 5th International Conference on Language Resources and Evaluation*, pages 449–445.

Rachele De Felice and Stephen G. Pulman. 2008. A classifier-based approach to preposition and determiner error correction in L2 English. In *Proc. of 22nd International Conference on Computational Linguistics*, pages 169–176.

Dana Ferris and Barrie Roberts. 2001. Error feedback in L2 writing classes: How explicit does it need to be? *Journal of Second Language Writing*, 10(3):161–184.

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

Japan electronic dictionary research institute Ltd. 1993. *EDR electronic dictionary specifications guide*. Japan electronic dictionary research institute ltd.

Jun'ichi Kakegawa, Hisayuki Kanda, Eitaro Fujioka, Makoto Itami, and Kohji Itoh. 2000. Diagnostic processing of Japanese for computer-assisted second language learning. In *Proc. of 38th Annual Meeting of the Association for Computational Linguistics*, pages 537–546.

Daisuke Kawahara and Kiyotaka Uchimoto. 2008. A method for automatically constructing case frames for English. In *Proc. of 6th International Conference on Language Resources and Evaluation*.

Kathleen F. McCoy, Christopher A. Pennington, and Linda Z. Suri. 1996. English error correction: A syntactic user model based on principled "mal-rule" scoring. In *Proc. of 5th International Conference on User Modeling*, pages 69–66.

George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41.

Ryo Nagata and Atsuo Kawai. 2011. Exploiting learners' tendencies for detecting English determiner errors. In *Lecture Notes in Computer Science*, volume 6882/2011, pages 144–153.

Ryo Nagata, Atsuo Kawai, Koichiro Morihiro, and Naoki Isu. 2006. A feedback-augmented method for detecting errors in the writing of learners of English. In *Proc. of 44th Annual Meeting of the Association for Computational Linguistics*, pages 241–248.

Ryo Nagata, Edward Whittaker, and Vera Sheinman. 2011. Creating a manually error-tagged and shallow-parsed learner corpus. In *Proc. of 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1210–1219.

Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. 2013. The CoNLL-2013 shared task on grammatical error correction. In *Proc. 17th Conference on Computational Natural Language Learning: Shared Task*, pages 1–12.

Randolph Quirk, Sidney Greenbaum, Geoffrey Leech, and Jan Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman, New York.

Thomas Robb, Steven Ross, and Ian Shortreed. 1986. Salience of feedback on error and its effect on EFL writing quality. *TESOL QUARTERY*, 20(1):83–93.

Alla Rozovskaya and Dan Roth. 2011. Algorithm selection and model adaptation for ESL correction tasks. In *Proc. of 49th Annual Meeting of the Association for Computational Linguistics*, pages 924–933.

Alla Rozovskaya and Dan Roth. 2013. Joint learning and inference for grammatical error correction. In *Proc. of Conference on Empirical Methods in Natural Language Processing*, pages 791–802.

Keisuke Sakaguchi, Yuta Hayashibe, Shuhei Kondo, Lis Kanashiro, Tomoya Mizumoto, Mamoru Komachi, and Yuji Matsumoto. 2012. NAIST at the HOO 2012 shared task. In *Proc. of 7th Workshop on the Innovative Use of NLP for Building Educational Applications*, pages 281–288.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for ESL learners using global context. In *Proc. of 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 198–202.

Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010. Using parse features for preposition selection and error detection. In *Proc. of 48nd Annual Meeting of the Association for Computational Linguistics Short Papers*, pages 353–358.

Yuanbin Wu and Hwee Tou Ng. 2013. Grammatical error correction using integer linear programming. In *Proc. of 51st Annual Meeting of the Association for Computational Linguistics*, pages 1456–1465.

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 *Proc. of 17th Conference on Computational Natural Language Learning: Shared Task*, pages 26–33.