VTEX Determiner and Preposition Correction System for the HOO 2012 Shared Task

Vidas Daudaravičius
VTEX
Akademijos 4
LT-08412 Vilnius, Lithuania
vidas.daudaravicius@vtex.lt

Abstract

This paper describes the system has been developed for the HOO 2012 Shared Task. The task was to correct determiner and preposition errors. I explore the possibility of learning error correcting rules from the given manually annotated data using features such as word length and word endings only. Furthermore, I employ error correction ranking based on the ratio of the sentence probabilities using original and corrected language models. Our system has been ranked for the ninth position out of thirteen teams. The best result was achieved in correcting missing prepositions, which was ranked for the sixth position.

1 Introduction

The correct usage of determiners and prepositions is one of the toughest problems in English language use for non-native speakers, especially those living in a non-English speaking environment. The issues have been explored extensively in the literature (see Leacock et al. (2010)). It was interesting to find that this error correction topic was chosen for the HOO 2012 Shared Task.

This paper describes the experimental system developed by VTEX team for this task – to correct determiner and preposition errors in CLC FCE Dataset. It explores the possibility of learning error correcting rules from the given manually annotated data using features such as word length and word endings only. Furthermore, it employs error correction ranking based on the ratio of sentence probabilities using original and corrected language models.

2 The data

The training data consisted of 1000 files drawn from the publicly available FCE dataset and converted into HOO data format (see Dale et al. (2012)). I used the HOO 2012 training and test data only. The training data had 8432 manually annotated corrections of the following six error types:

MD – Missing Determiner;

MT – Missing Preposition;

UD – Unwanted Determiner;

UT – Unwanted Preposition;

RD – Replacement Determiner;

RT – Replacement Preposition.

The total size of the training data was 374680 words. The test data consisted of 100 previously unseen files without error correction annotations. For more details about the training and test data, see (Dale et al., 2012).

I have not used any other dictionaries, corpora or language processing tools (like taggers or parsers). Thus, the system is language independent and based on supervised learning of manually annotated corrections.

3 Word length and word ending

The training corpus was small and insufficient to get complete and reliable features and statistics of error corrections based on the corrected words. Therefore I needed to find features which describe the contexts of error corrections

The 7th Workshop on the Innovative Use of NLP for Building Educational Applications, pages 225–232,
Montréal, Canada, June 3-8, 2012.

in a more generalized way. After some experimentation, I chose word length and the word last n characters. Words in the dataset were transformed into tokens using these functions. I have tested three word transformation combinations:

word – keeps the whole word (e.g. *make* $\mapsto$ *make*);

2end – takes the length of a word and adds the last two characters (*make* $\mapsto$ *4_ke*);

1end – takes the length of a word and adds the last character (*make* $\mapsto$ *4_e*).

I have also used lists of reserved words that were used to preserve the primary form of a word:

corrections – words that were corrected to/from in HOO 2012 Gold Edits data;

mod – functional words such as: *have*, *has*, *can*, *not*, *make*, *made*, *be*, *was*, *were*, *am*, *are*, *and*, *or*;

pronouns – pronouns that were not used as corrections: *we*, *he*, *she*, *they*, *yours*, *ours*, *them*.

For instance, using 2end transformation, the incorrect sentence *I feel that festival could be even better next year* was transformed into *I 4el that 8al 5ld be 4en 6er next 4ar*, and the corrected sentence into *I 4el that the 8al 5ld be 4en 6er next 4ar*.

In Section 5, I show that the word length and ending retain a lot of information about the word.

Each participating group in HOO 2012 Shared Task was allowed to submit up to ten runs. I have submitted nine runs that differ in word length and word ending only. The different runs are:

0 – 1end: all words except reserved correction words were encoded as word length + the last character;

1 – 2end: all words except reserved correction words were encoded as word length + two last characters;

2 – word: no transformations;

3 – 1end+mod: all words except reserved correction and mod words were encoded as word length + the last character;

4 – 2end+mod: all words except reserved correction and mod words were encoded as word length + two last characters;

5 – 1end+pron: all words except reserved correction and pronoun words were encoded as word length + the last character;

6 – 2end+pron: all words except reserved correction and pronoun words were encoded as word length + two last characters;

7 – 1end+mod+pron: all words except reserved correction, pronoun and mod words were encoded as word length + the last character;

8 – 2end+mod+pron: all words except reserved correction, pronoun and mod words were encoded as word length + two last characters.

4 Error correction

Error correction consists of **the rules** that the system is able **to learn**, and **the actions** that the system is able **to apply**.

4.1 Error correction rules

Using error correction annotations from Gold Edits of the training corpus we have built the error correction rules. The error correction rule is the error correction and the context of this correction. From the training corpus I gather contextual correction rules. The context are tokens on the left- or right-hand side of the error correction. The best choice would be to take at least two tokens on the left-hand side and two tokens on the right-hand side and to express error correction rule as a 5-gram with the error correction in the middle. For instance, in the training data, the error correction of *for* to *about* of type *RT* is found within the the left-hand side context *i asked* and the right-hand side context *the discounts*.

The main problem of learning of correction rules was the small size of the training corpus.

Bigger corpora could help in learning more correction rules. But it is hard to get bigger corpora because it is very expensive to prepare them. Two or three word context on each side of a corrected fragment can produce good but rarely applicable correction rules. Therefore, I have implemented a smoothing technique for generating new rules that do not appear in the training data.

I use trigrams to generate smoothed 5-gram error correction rules. Three types of trigrams were used for the smoothing:

centered – one token on the left-hand side of the correction, then the correction and one token on the right-hand side of the correction (see line 1 in Table 1);

left – two tokens on the left-hand side of the correction and the correction (see lines 2 and 3 in Table 1);

right – two tokens on the right-hand side of the correction and the correction (see lines 4–13 in Table 1).

There are 8432 corrections in the training data. Figure 1 shows the number of distinct trigram rules for the different runs described in Section 3. Most of the trigram rules appear once. For instance,

L_2	L_1	type	original	correction	R_1	R_2
	asked	RT	for	about	the	
i	asked	RT	for	about		
to	asked	RT	for	about		
		RT	for	about	the	camp
		RT	for	about	the	discounts
		RT	for	about	the	experience
		RT	for	about	the	first
		RT	for	about	the	new
		RT	for	about	the	news
		RT	for	about	the	play
		RT	for	about	the	prise
		RT	for	about	the	terrible
		RT	for	about	the	very

Table 1: Trigram error correction rules.

- the most frequent (38 occurrences) left-context trigram rule without word encoding is *stay in* ***/a/MD***;
- the most frequent (44 occurrences) right-context trigram rule is ***on/in/RT*** *july because*; and
- the most frequent (38 occurrences) centered-context trigram rule is *travel* ***on/in/RT*** *july*.

We could expect similar generalization power for left, right or centered contexts, but in Fig. 1 we can see that the number of distinct right-hand side contexts is lower by 5% compare to

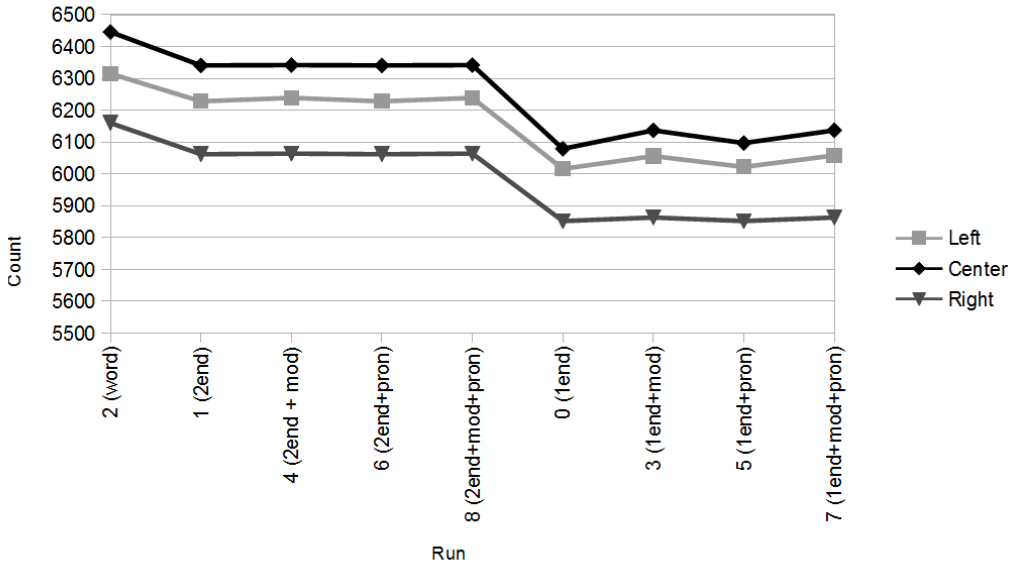

Figure 1: The number of context trigrams of error corrections for the different runs.

the number of distinct centered contexts. Surprisingly, the number of trigram rules does not degrade significantly whether the encoding 1end is used or not.

The new smoothed 5-gram rules are extensions of the centered trigram rules. The extension on the left hand-side is the union of centered trigrams and the left trigrams when the error correction and L_1 match. And the extension on the right hand-side is the union of the centered trigrams and the right trigrams when error correction and R_1 match. For instance, the error correction of *for* to *about* of type *RT* within the the left-hand side context *i asked* and the right-hand side context *the discounts* is extended as follows:

- take centered trigram (see line 1 in Table 1);
- take left trigrams, where correction and L_1 match (see lines 2 and 3 in Table 1);
- take right trigrams, where correction and R_1 match (see lines 4–13 in Table 1);
- after that I have the following smoothed rule: L_2 = [*I, to*], L_1 = *asked,* C = *for/about/MT,* R_1 = *the,* R_2 = [*camp, discounts, experience, first, news, play, prise, terrible, very*].

This technique allows the generation of error correction rules that do not appear in the training data, e.g. in the latter example I generate 18 smoothed 5-gram rules that do not appear in the training data. The new smoothed 5-gram error correction rule is boolean operation and the rule does not contain any probabilistic information.

4.2 Error correction actions

The error correction system applies error correction rules using the following actions:

do not change – word is kept as is;

insert – missing word is inserted;

delete – unnecessary word is deleted;

replace – word is replaced by another one.

Each action is tested at each word but only one at a time. In case the context allows to apply several actions at one place then these actions are treated as alternatives. Alternative actions are not combined and no selection between

Doc ID	Run	Rules applied	OC ratio	sentence correction
2025	2	the//MD/ that/this/RD/	0.451	is the 8 th july till **the** end of **that** month , what do you think ?
		that/this/RD/	0.559	is the 8 th july till end of **that** month , what do you think ?
		the//MD/	0.633	is the 8 th july till **the** end of this month , what do you think ?
		–	0.785	is the 8 th july till end of this month , what do you think ?
	7	the//MD/ that/this/RD/	0.345	3s the 8 2h 4y till **the** 3d of **that** 5h , what 2o you 5k ?
		that/this/RD/	0.441	3s the 8 2h 4y till 3d of **that** 5h , what 2o you 5k ?
		the//MD/	0.533	3s the 8 2h 4y till **the** 3d of this 5h , what 2o you 5k ?
		–	0.683	3s the 8 2h 4y till 3d of this 5h , what 2o you 5k ?
2043	2	/for/UT/	0.976	i am writing in response to your last letter , to answer and ask you ~~for~~ some questions .
		–	1.035	i am writing in response to your last letter , to answer and ask you for some questions .
	7	/for/UT/	0.966	i am 7g in 8e to your 4t 6r , to 6r and 3k you ~~for~~ some 9s .
		a//MD/ /for/UT/	1.022	i am 7g in **a** 8e to your 4t 6r , to 6r and 3k you ~~for~~ some 9s .
		–	1.025	i am 7g in 8e to your 4t 6r , to 6r and 3k you for some 9s .
		a//MD/	1.085	i am 7g in **a** 8e to your 4t 6r , to 6r and 3k you for some 9s .

Table 2: Examples of ranking, selection and application of actions for sentence correction.

them is made at this step. The example of correction alternatives is shown in Table 2. Besides, the probability of the action can be taken into account but I do not do this and all actions are considered equally possible.

5 Language model

I use language trigram modeling to estimate the probability of a sentence. The probability of a sequence of words is estimated as the product of probabilities of trigrams:

$$p(x) = \prod_i \hat{p}(x_i \,|\, x_{i-2}, x_{i-1}).$$

To avoid zero probability I have used Kneser–Ney trigram smoothing (Kneser and Ney, 1995) technique as follows:

$$\begin{aligned}
&\hat{p}(x_i \,|\, x_{i-2}, x_{i-1}) \\
&\quad = \frac{\max[(\mathrm{freq}(x_{i-2}, x_{i-1}, x_i) - c_3), 0]}{\max[\mathrm{freq}(x_{i-2}, x_{i-1}), 1]} \\
&\qquad + \frac{c_3 * |x_{i-2}, x_{i-1}, \bullet|}{\max[\mathrm{freq}(x_{i-2}, x_{i-1}), 1]} \\
&\qquad\quad \times \frac{\max[(\mathrm{freq}(x_{i-2}, x_{i-1}) - c_2), 0]}{\max[\mathrm{freq}(x_{i-2}), 1]} \\
&\qquad + \frac{c_2 * |x_{i-2}, \bullet, \bullet|}{\max[\mathrm{freq}(x_{i-2}), 1]} \\
&\qquad\quad \times \frac{\max[(\mathrm{freq}(x_{i-2}) - c_1), 0]}{N} + \frac{c_1 * T}{N},
\end{aligned}$$

where $c_3 = 0.8$, $c_2 = 0.6$, $c_1 = 0.4$, $T = |\bullet|$, and N is the corpus size.

I have built two language models: one for the original language and one for the corrected language. The original language model (O) was built using the corpus without corrections. The corrected language model (C) was built using the corpus with error corrections applied. The different runs yield different number of token trigrams. But the number does not degrade significantly as we might expect when words are encoded with the `1end` transformation (see Fig. 2). Thus, the `1end` transformation retains a lot of information, although, the number of trigrams of the original language model is always a little bit higher than the number of trigrams of the corrected language model.

6 The probability ratio of the original and corrected language models

The probability of a sentence depends on the length of the sentence. The longer the sentence the lower the probability. Error correction actions can change the length of a sentence. Thus, it is hard to implement the error correction system which should rank different length sentences. Therefore, I have used the ratio of the probabilities of the sentence using the original language model (O) and the corrected language

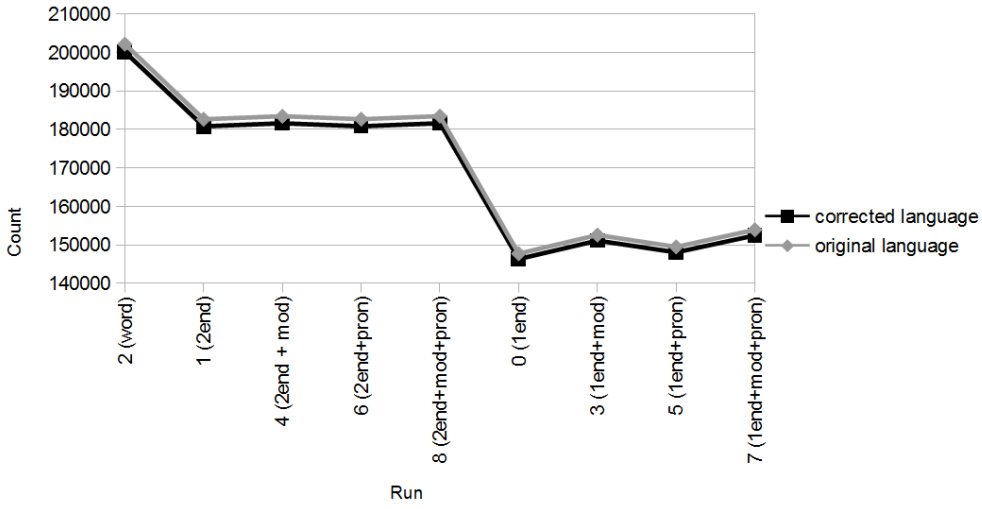

Figure 2: The number of trigrams of the original and corrected language models for different runs.

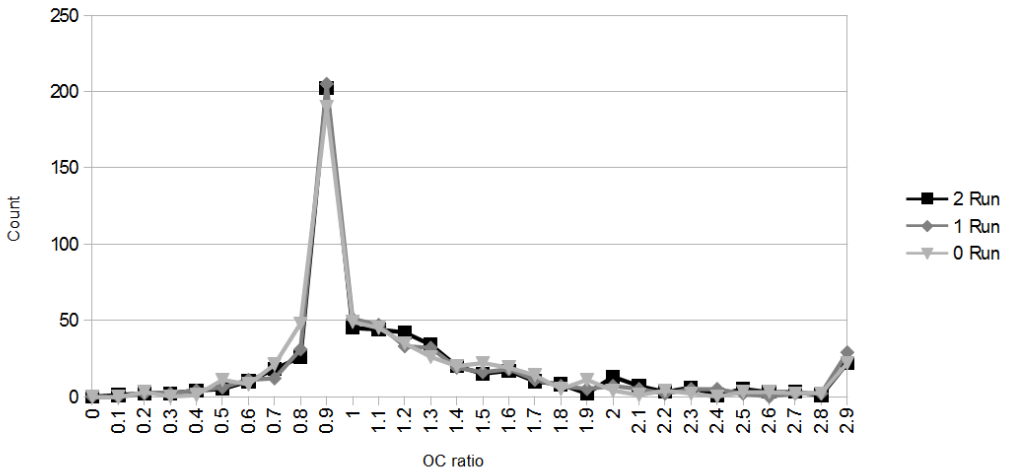

Figure 3: The histogram of *OC ratio* in the test data.

model (C):

$$OC\ ratio = \frac{\hat{p}(O)}{\hat{p}(C)},$$

where $\hat{p}(O)$ is the probability of a sentence using the original language model and $\hat{p}(C)$ is the probability of the same sentence using the corrected language model.

The lower the value of this ratio, the higher the chance that the sentence is correct, i.e. closer to corrected language rather than to original language. In Fig. 3, I show the histogram of the highest *OC ratios* of the corrected test sentences. This histogram shows that most of the ratios are close to 1, i.e. the probabilities of the sentence are almost equal using both language models. The histogram does not depend on the type of word encoding. In Table 2, I show examples of corrections and the *OC ratios* for each set of corrections. The error correction system takes corrections which are applied for the sentence with the lowest *OC ratio* (see Table 2).

7 The results and conclusions

The results for different runs of the error correction system are shown in the Table 3. The best determiner and preposition correction F-score results are achieved with *Run 5*, which is using 1end + pron encoding: all words except reserved correction and pronoun words were encoded as word length + the last character. This result was ranked for ninth position out of 14 teams.

Nevertheless, the results for different types of corrections are quite different. The error correction system was capable of performing UT, MT and MD type error corrections but hopeless for UD, RD and RT type error corrections. The best results are for:

MT – missing preposition error correction, no encoding is used;

MD - missing determiner error correction, 2end encoding is used;

UT - unwanted preposition error correction, any type of encoding except no encoding.

Surprisingly, we had to use whole words for missing preposition error correction, but never for unwanted preposition error correction. Our system was ranked at the seventh position for UT error correction using F-score.

The result for MT error correction shows that smoothed 5-gram rule generation was useful and the whole word should be used. But encoding with word length should never be used. Our system is ranked at the sixth position for MT error correction.

The result for MD error correction shows that the system degrades when encodings with fewer characters are used.

Run	All			MT			MD		
	P	R	F	P	R	F	P	R	F
0	8.15	**4.19**	5.54	4.65	3.50	4.00	7.84	**9.60**	8.63
1	24.5	2.87	5.13	12.5	**3.51**	5.48	34.8	6.40	**10.8**
2	**35.5**	2.43	4.54	**25.0**	**3.51**	**6.15**	**46.7**	5.60	10.0
3	8.41	3.75	5.19	5.56	**3.51**	4.30	8.27	8.80	8.53
4	25.0	2.87	5.15	13.3	**3.51**	5.56	34.8	6.40	**10.8**
5	8.76	**4.19**	**5.67**	5.00	**3.51**	4.12	8.57	**9.60**	9.06
6	24.5	2.87	5.14	12.5	**3.51**	5.48	34.8	6.40	**10.8**
7	9.04	3.75	5.30	5.71	**3.51**	4.35	9.17	8.80	8.98
8	25.0	2.87	5.15	13.3	**3.51**	5.56	34.8	6.40	**10.8**

Run	UT			UD			RT			RD		
	P	R	F	P	R	F	P	R	F	P	R	F
0	**100**	**4.65**	**8.89**	**4.76**	**1.89**	**2.70**	1.67	**1.47**	**2.70**	0	0	0
1	**100**	**4.65**	**8.89**	0	0	0	16.7	0.74	1.41	0	0	0
2	**100**	2.33	4.55	0	0	0	**20.0**	0.74	1.42	0	0	0
3	**100**	**4.65**	**8.89**	0	0	0	16.7	**1.47**	**2.70**	0	0	0
4	**100**	**4.65**	**8.89**	0	0	0	16.7	0.74	1.41	0	0	0
5	**100**	**4.65**	**8.89**	**4.76**	**1.89**	**2.70**	16.7	**1.47**	**2.70**	0	0	0
6	**100**	**4.65**	**8.89**	0	0	0	16.7	0.74	1.41	0	0	0
7	**100**	**4.65**	**8.89**	0	0	0	16.7	**1.47**	**2.70**	0	0	0
8	**100**	**4.65**	**8.89**	0	0	0	16.7	0.74	1.41	0	0	0

Table 3: Scores for correction of different runs.

The main conclusion is that there are no common features for all error corrections and the different systems for different error types should be implemented.

References

Robert Dale, Ilya Anisimoff, and George Narroway. 2012. Hoo 2012: A report on the preposition and determiner error correction shared task. In *Proceedings of the Seventh Workshop on Innovative Use of NLP for Building Educational Applications*, Montreal, Canada, June.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume I, pages 181–184, Detroit, Michigan, May.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2010. *Automated Grammatical Error Detection for Language Learners*. Morgan and Claypool Publishers.

MD

Team	Run	Precision	Recall	F-Score
CU	0	83.33	8.0	14.6
KU	2	1.98	20.0	3.6
LE	0	54.43	34.4	42.16
NA	1	29.09	38.4	33.1
NU	0	51.02	40.0	44.84
TC	3	6.21	7.2	6.67
TH	3	9.54	26.4	14.01
UI	0	51.92	**43.2**	**47.16**
UT	6	36.7	32.0	34.19
VA	0	6.4	6.4	6.4
VT	1	34.78	6.4	10.81

MT

Team	Run	Precision	Recall	F-Score
CU	1	5.68	8.77	6.9
KU	1	0.51	**19.3**	1.0
LE	0	50.0	5.26	9.52
NA	3	11.43	7.02	8.7
NU	0	38.46	17.54	**24.1**
TC	3	4.65	3.51	4.0
UI	5	42.86	15.79	23.08
VA	1	1.71	7.02	2.75
VT	2	25.0	3.51	6.15

UT

Team	Run	Precision	Recall	F-Score
CU	1	4.83	**39.53**	8.61
JU	1	2.91	6.98	4.11
KU	5	60.0	13.95	22.64
LE	1	32.14	20.93	25.35
NA	3	40.91	20.93	**27.69**
NU	0	40.0	13.95	20.69
TC	9	4.69	30.23	8.13
TH	1	10.32	30.23	15.38
VA	0	12.9	18.6	15.24
VT	0	**100.0**	4.65	8.89

UD

Team	Run	Precision	Recall	F-Score
CU	3	17.86	18.87	18.35
JU	1	4.84	5.66	5.22
KU	8	26.92	13.21	17.72
LE	0	22.67	32.08	26.56
NA	5	40.0	11.32	17.65
NU	0	33.33	9.43	14.71
TC	9	5.11	16.98	7.86
TH	1	38.89	13.21	19.72
UI	2	23.38	**33.96**	**27.69**
VA	0	7.06	11.32	8.7
VT	0	4.76	1.89	2.7

Table 4: Scores for correction.