Computational Linguistics in the Netherlands Journal 6 (2016) 39-55 Submitted 05/2016; Published 12/2016

Native-data Models for Detecting and Correcting Errors in Learners' Dutch

Lennart Kloppenburg L.KLOPPENBURG@RUG.NL
Malvina Nissim M.NISSIM@RUG.NL

CLCG, University of Groningen, The Netherlands

Abstract

We address the task of automatically correcting errors in text written by learners of Dutch by modelling language usage of native speakers. Specifically, we concentrate on two word classes, namely prepositions and determiners, with a focus on articles for the latter. For each of these two word classes, we build two models exploiting a large corpus of Dutch. The first is a binary model for detecting whether a preposition/article should be used at all in a given position or not. The second is a multiclass model for selecting the appropriate preposition/article in case one should be used. The models are tested on native as well as learners data. For the latter we exploit a crowdsourcing strategy to elicit native judgements. On native test data the models perform very well, showing that we can model preposition usage appropriately. However, the evaluation on learners' data shows that the models might be excessively tuned towards native data and there is still room for improving their adaptation to the intrinsic characteristics of learners' data. Reflecting on such results, we envisage various ways of improving performance, and report them in the final section of this article.

1. Introduction

Computer-assisted language learning is a field where language technology is combined with human language acquisition. To make it possible, we need systems that are able to recognise errors and suggest corrections. In spite of recent campaigns on automatic error correction (Ng et al. 2013, Ng et al. 2014) and in spite of some notable exceptions (Nicholls 2003), the lack of large amounts of error-annotated data remains a bottleneck in the development and evaluation of systems that support language learning, especially for L2s other than English.

One obvious solution is creating error-aware resources to build and test robust models of learners' language use (Han et al. 2010), which has however substantial costs, especially in terms of human effort. Another way of bypassing this shortage of data is to exploit *native* data, which can be considered as having gold labels of correct usage of language (Han et al. 2006, Tetreault and Chodorow 2008, Gamon et al. 2008, De Felice and Pulman 2009). We adopt this latter strategy in this work and we apply to two case studies, namely preposition and determiner errors in learners of Dutch. As far as we are aware, this is the first work on this topic for this language.

In this paper we describe a two-stage approach to the detection and correction of preposition and determiner errors in essays written by learners of Dutch. For each phenomenon we train two Support Vector Machine models on native data that are to be deployed in a pipeline. The first model takes care of determining if a preposition/determiner is correctly present at a given spot, while the second one determines whether the kind of preposition/determiner used is indeed appropriate. All models are tested both on withheld native data as well as learners' data. For the latter, we crowdsourced judgments from native speakers as the corpus we drew test data from is not specifically error-annotated.

2. Phenomena and Tasks

Prepositions and determiners possess several characteristics that make them suitable part-of-speech classes for experimenting with modelling errors produced by second language learners. Firstly, as noted by De Felice and Pulman (2008), who have also worked on those classes focusing on learners of English, prepositions and articles are among the most frequent sources of errors for L2 English speakers. They show this by annotating a small corpus where preposition and determiner errors amount to 12% and 17% respectively. Even without exact figures, we can assume that these phenomena will be problematic for Dutch as L2 as well, not least because it is closely related to English. Secondly, both prepositions and determiners occur very frequently in native data, thus making it at the same time important for learners to acquire their proper usage and possible for researchers to access plenty of evidence from naturally occurring L1 texts. Finally, these are closed and relatively small lexical classes, thereby allowing for a more accurate characterisation and manageable modelling, also in terms of providing feedback to learners. We characterise these two classes in Dutch in the following sections. For determiners, we focus on articles.

2.1 Prepositions

It has been noted that in many languages prepositional constructions are the most difficult for foreign language learners, because of their versatile and often ambiguous character (Dale et al. 2012). Although we have no specific figures on this kind of error in learners of Dutch (but see Section 5.3 below), we know that prepositions are used extremely frequently and that large corpora of native Dutch are available, which can be exploited to build models of correct preposition usage.
In Dutch, there are just over a hundred prepositions, and only some of them are used commonly. In the `cdb` (Alpino Treebank, (van Noord 2006)), a corpus of ca. 140,000 words from the Eindhoven corpus (newspaper text), about 15% of all tokens are prepositions, despite only representing 0.66% of all types (n=15 863), with the top 15 prepositions comprising almost 90% of all preposition use (see Table 1 for details). In other words, prepositions are used very frequently despite being a relatively small lexical class. It should be noted that the tokens in this table can also act as *particles* (i.e. *'Ik eet de boterham **op**'*, 'I am eating the sandwich', where 'op' is a particle of the verb 'opeten' (to eat (up)).), depending on the context. In these cases, the token is no longer a preposition grammatically, but it still is morphologically. It is unclear what characteristics this discrepancy has within the scope of language learning, and we have instead focused on the token itself.

2.2 Articles

Articles are part of the larger class of determiners. This work focuses on three Dutch articles: *de (58.6%), het (20.5%)* and *een (20.8%)* (frequencies were also drawn from `cdb`, where total n = 15,256). *de, het* (the) are definite articles which are used in noun phrases referring to specific or known entities or names, and *een* (a, an) is an indefinite article, commonly used when referring to new or unspecific entities in a sentence or text. The difference between *de* and *het* is gender-based: *de* is used for gender-specific nouns and *het* is used for genderless nouns. Additionally, all Dutch diminutives always appear with the article *het* (1), whereas all Dutch plurals always occur with the article *de*, independently of the article they take in their singular form. Finally, noun phrases with verbal heads also adopt the article *het* (2).

(1) **De** eend (the duck)

Het eend*je* (the duck*ling*)

(2) NP. **het** krijgen [van] nakomelingen (***[the]** getting [of] descendants*)

VP. nakomelingen krijgen (*getting descendants*)

Table 1: Frequency distribution of the 15 most Dutch prepositions in the **cdb**. The total number of prepositions in the corpus is 18,790.

Preposition	**Translation**	**Frequency**
van	of	22.8%
in	in	16.3%
op	on	7.2%
te	at	7.1%
voor	for/before	6.3%
met	with	5.7%
aan	on, to	4.3%
door	through, by	3.3%
bij	at, with	2.8%
uit	out	2.6%
om	by, around	2.6%
over	over, about	2.4%
tot	until, to	2.2%
naar	to	2.1%
als	if, as	1.2%
total		88.9%
others		11.1%

2.3 Tasks

The problem of automatically correcting preposition and article errors in texts written by language learners is treated uniformly in our approach, with a single method for both, which however yields different models that exploit only partially the same features.

The general approach that we propose is to divide the general task into two subtasks: the first is *error detection*, i.e., train a system to spot errors, while the second is *error correction*, i.e., the task of providing the language learner with appropriate feedback or correction.

This strategy follows naturally from the analysis of the three basic situations of misuse that have been identified in error correction literature (Eeg-Olofsson and Knutsson 2003, Chodorow et al. 2007, Liu 2008):

1. insertion (a token is invoked erroneously)[1]
2. deletion (a token is omitted erroneously)
3. substitution (a token is picked erroneously)

Insertion errors are shown in Examples (3) and (4) for prepositions and articles, respectively, while deletion errors are shown in Examples (5) and (6). In Example (a) we report the learner's sentence with the error, and in (b) the correct version.

(3) a. *Hij zet tot 8 soorten **van** dranken, maar geen chocolade melk. *(It can make up to eight types **of** drinks, but no hot chocolate.)*

b. Hij zet tot 8 soorten **ø** drank, maar geen chocolademelk.

(4) a. *Hij lust zowel **de** appels als **de** peren. *He likes both **the** apples and **the** pears.*

b. Hij lust zowel **ø** appels als **ø** peren.

1. In our study, a *token* can be either a preposition or a determiner.

(5) a. *Roger Federer werd **ø** 1981 geboren. *(*Roger Federer was born **ø** 1981.)*

b. Roger Federer werd **in** 1981 geboren.

(6) a. * [...] grotere steden tonen **ø** steeds hogere temperatuur. (*Larger cities measure **ø** constantly higher temperature.*)

.... grotere steden tonen **een** steeds hogere temperatuur.

In a substitution error, as in Example (7), the learner understands that a preposition must be used, but picks the wrong one (a). The task is to find this out, and replace the token with the correct one (b). A similar situation can be observed in the Example (8), where "de" is used instead of "het".

(7) a. *Vanaf nu gaat alle communicatie **door** email, en niet de telefoon. *(From now on, all communication will be done **through** email, rather than the telephone.)*

b. Vanaf nu gaat alle communicatie **via** email, en niet de telefoon.

(8) a. *Ze hebben **de** kind gestraft. *(They punished the child.)*

b. Ze hebben **het** kind gestraft.

This classification effectively separates distinct types of errors which might require different strategies to solve. Note that the *deletion* and *insertion* errors are mirrored, as shown in Table 2, hinting at a possibly joint treatment.

Table 2: Mirroring of *deletion* and *insertion* error types, where T is a specific article or preposition.

	Correct use	**Learner sample**
Deletion	token T	no token
Insertion	no token	token T

Indeed, for *insertion* and *deletion*, the task is to model presence or absence of a preposition/article in any given context, and could be thus conceived as a binary classification task.

3. Data

To build and test our models we used two datasets. The first is LASSY Large (LArge Scale SYntactic Annotation of Written Dutch (van Noord et al. 2011, van Noord et al. 2013)), a syntactically annotated corpus of native Dutch which includes newswire and Wikipedia articles, for a total of 700M words and over 64M sentences.

The second is Leerdercorpus Nederlands (Perrez and Degand 2009), an as of yet relatively rarely used corpus which contains 3,468 essays and argumentative texts written by students who study Dutch as a second language, with a variety of native languages. The total amount of tokens is reported at 774,658. The distribution of native languages expresses quite a bit of variation, including French (1247), German (877), Polish (599), Hungarian (413), Indonesian (197), English (9) and others which are not defined further (125), making it not quite possible to rely on the regularities of source-language-specific error types.

To the best of our knowledge, this is the only learner corpus for Dutch, but it is not error-annotated, meaning that the density of errors is unknown and that evaluation cannot be done automatically against a gold standard (see Section 5.2 for further details). Still, the samples in Example 3 illustrate that the corpus exhibits typical L2 learners' behaviour. A large subset of the corpus was converted to a treebank (similar to LASSY Large) using the Alpino syntactic parser. This way we could reproduce the process of extracting feature vectors from learners' data in the same way we did for native data, which was syntactically annotated already.

(9) a. *Dat zijn mooie dromen. Ik hoop dat ze *de* werkelijkheid zullen worden.

b. *Those are beautiful dreams. I hope they will become *the* reality.

c. Dat zijn mooie dromen. Ik hoop dat ze ø werkelijkheid zullen worden.

Comments. An article insertion error

(10) a. *Ze hadden een relatie en ze konden geen *sexueele* (=seksuele) relatie voor het huwelijk hebben omdat het *religie tegen* (tegen hun religie) was. Dus zijn ze gehuwelijkt (=getrouwd)!

b. *They were in a relationship and could not engage in sexual relations before marriage because that was against religion. So they married.

Comments. The indefinite form of 'sexual', in Dutch, is 'seksueel', which could have induced the confusion that the definite form ('seksuele') also needs two *e*s. 'gehuwelijkt' does not exist, just like the perceived infinitive 'huwelijken' is not a correct translation of 'to marry'. The writer of this sentence spoke French as L1, where 'mariage' (*marriage*) directly stems from '(se) marier' (*to marry*), and this analogy was probably incorrectly carried over to Dutch.

(11) a. *Afsluitend zijn *demedia* heel *interresant* (=interessant) en belangrijk voor ons.

b. In conclusion, the media are very interesting and important to us.

Comments. *demedia* was misspelled, but this might well have been a typo rather than a language mistake. *interresant* is misspelled as well. In case of a spelling error, Alpino will still try to determine the most likely category for the token. In the case of *interresant*, however, Alpino guessed that it was a *noun* rather than an *adjective*. This will directly lead to erroneous featurisation; the surrounding POS n-grams will all be wrong, and the token values will not be recognised because we assume that the token itself does not exist in the native corpus.

4. Method

In order to detect potential errors, one needs to be able to discriminate between good language use and incorrect language use. In absence of error-annotated data, we trained models of correct usage on LASSY Large, with the assumption that native data provides gold labels on grammatical choices. It is worth noting that because LASSY Large is already fully syntactically annotated, all linguistic properties that we will present as features were extracted from the corpus without additional preprocessing steps. Building on the observations on error types in Section 2, we trained two different models for each phenomenon. The first is a *binary* detection model trained on positive and negative events, and used to predict whether a given context contains a preposition/article or not. The second is a *multiclass* selection model. For prepositions, we trained a classifier with fifteen prepositions as class values, used to select a preposition if none was present (as indicated by the detection model), or if the present preposition was wrong. Among the three articles we consider, two are definite ("het", and "de"), while "een" is indefinite. While it could be a possibly successful alternative, which we might explore in future work, our selection model does not consciously take any steps beforehand to determine if a given determiner token should be indefinite or definite, so that we are relying on its ability to do this inherently. For both tasks, the detection and selection models are deployed in a pipeline, meaning that the actions of the selection model depend on the outcome of the detection model, as outlined in the following sections. In what follows, for the sake of space and in order to avoid repetitions, we primarily describe the functioning of the preposition models. A similar procedure has been used to develop the article models. When the models differ we make this distinction explicit and report details for both phenomena.

4.1 Detection model

Using an SVM with a linear kernel,[2] we trained a model to detect insertion and deletion errors: given a feature-vector, the model predicts whether this vector is built around a white-space (absence) or a preposition (presence). We define a white-space vector as a vector which is extracted around a case which featured no preposition, but could potentially have one given its surrounding linguistic properties. In other words; such a vector expresses a surrounding structure that can frequently be found *with* a token. We determined this by comparing POS n-gram chunks without prepositions with POS bigram windows on either side of a preposition. We did this by first extracting POS fourgram chunks *without prepositions/articles* from the native data corpus and creating a frequency distribution of these. Every chunk would have a) the frequency that this chunk was found *around* a preposition/article and b) a value denoting the frequency that this chunk was found *without* a preposition/article. This essentially creates a simple model that indicates how rare certain patterns are for either the presence or absence of a preposition or article. To give an example, let us consider a case where we have a POS pattern that includes a preposition: *VERB_ADV_PREP_NOUN_$* (where $ is end of sentence). Any case with the same POS context *but without a preposition*, such as *VERB_ADV_NOUN_$*, can be considered as a *non-trivial case of preposition absence*. A pairwise example to illustrate the concept of this *non-trivial* preposition absence is given in (12) and (13).

(12) De man gaat vaak **naar** concerten. *(The man goes **to** concerts often.)*

(13) De man bezoekt vaak **ø** concerten. *(The man visits **ø** concerts often.)*

Note how in the first sentence, the verb "gaat" (*goes*) needs a preposition, in this case "naar" (*to*). The second sentence is written without a preposition but has exactly the same surrounding structure in terms of parts of speech. The difference here is that "bezoek" (*visits*) does not require a preposition as it is a transitive verb and takes a direct object. As explained above, since the surrounding POS pattern could hypothetically hold a preposition (as in Example (12)), we call this a case of non-trivial preposition absence. Because we are in fact talking about the *absence* of a preposition, these have to be considered as *negative* instances and their (preposition) class value is set to "false". A preposition case, i.e., a *positive* instance with class value "true", is a vector whose linguistic properties were extracted around a preposition. The actual task is to then train the model to find distinctions between these two types of vectors.

The same principle can be applied to the use of articles; one may refer to nouns without the use of a preceding article if context permits it. Such contexts usually involve names, general knowledge concepts (e.g. *global warming*) or fixed/idiomatic constructions (e.g. *both apples and pears*). For articles, we therefore decided to experiment with new ways of modelling the (non-)triviality of absence or presence which we have described for prepositions above. This resulted in four candidate classes: the trivial or non-trivial presence or absence of articles. This extra distinction was added to try and model when articles are used or omitted whereas the context would indicate otherwise. This leads to the proposed method of how we try to catch insertion and deletion errors.

The model is informed by n-gram-based features (see Table 3) for both prepositions and articles, both at the word- and at the POS-level, which define the linguistic context in which the preposition/article or the whitespace occurs.
The possible outcomes of the model, both for prepositions and for articles, are shown in Table 4.

Finally, since we had concluded our experiments on prepositions before including the article segment of this paper, we were able to reflect on the results on the detection task for prepositions (please refer to Section 5.3.1). Because of this, we decided to make a discrepancy between *triviality* and *non-triviality*, rather than between just *absence* or *presence*.

2. As per `http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC`, with default parameters

Table 3: Features for the detection model for both prepositions and articles.

Feature	**Description**
bigram_left	Token bigram left of token
bigram_right	Token bigram right of token
trigram_left	Token trigram left of token
trigram_right	Token trigram left of token
bigram_postags_left	POS bigram left of token
bigram_postags_right	POS bigram right of token
trigram_postags_left	POS trigram left of token
trigram_postags_right	POS trigram right of token

Table 4: Outcomes and procedure for a detection model.

Observed	**Predicted**	**Message**	**Action**
Presence	Presence	No error (yet)	Feed to *selection model*
Presence	Absence	*Insertion error*	Delete token
Absence	Presence	*Deletion error*	Feed to *selection model*
Absence	Absence	No error	Stop.

4.2 Selection model

A selection model is used for selecting suitable tokens. As outlined in Table 4, this can occur in two situations: because of a *substitution* error or because of a *deletion* error. Note that before feeding the vector with a token to this selection model, it is still unknown whether this token (a preposition or an article) is correct or incorrect. Attesting this is done by analysing the output from the model. Table 5 illustrates the different outcomes of the selection model.

Table 5: Outcomes and procedure for a selection model. *Assigned by detection model.

Observed	**Predicted**	**Message**	**Action**
None	P_j	Deletion error*	Correction: use P_j
P_j	P_j	No error	None
P_j	P_i	Substitution error	Correction: use $P_{\neg j}$

For the task of preposition error correction, we selected the 15 most frequent prepositions in Dutch (see Table 1), and trained an SVM model with a linear kernel on vectors of native data, using the features in Table 6, inspired by Chodorow et al. (2007). The selected preposition is predicted as a single-label classification task. For articles, we used the three previously described articles ("de", "het", "een"). It is important to note that the class labels are stored but not used as feature during the prediction/error correction tasks (Section 5). A generic sentence, processed by Alpino, is shown in Figure 1. How the features and their values are encoded is subsequently exemplified in the third column of Table 6. The features for the article model differ from those for the preposition model. The features from the preposition model were heavily inspired by Chodorow et al. (2007), but we decided to add some more features that could be able to discern between indefinite (een) and definite (de/het) articles in Dutch, as well as between the gender characteristics of definite articles. The use of articles in Dutch and English is very similar, but there are also a few differences (Donaldson 2008). For example, in Dutch, articles are omitted when describing professions (*Ik ben ø tandarts. I am **a** dentist.*) and instruments (*Ik speel ø piano. I play **the** piano.*). The reverse is true in case of seasons and languages (*in **het** Nederlands, in ø Dutch*). Whether, how exactly, and how much such characteristics of Dutch can in practice be exploited towards better error detection is left to future work.

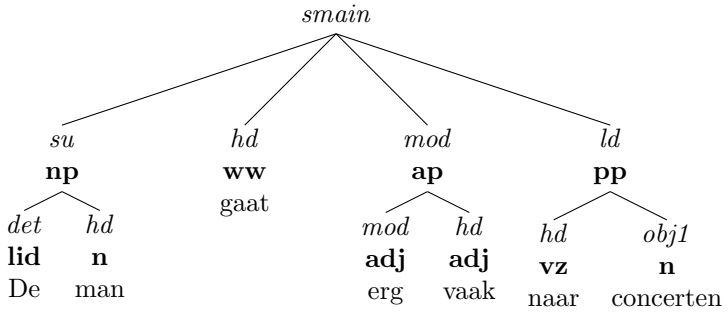

Figure 1: Example Alpino dependency tree: *The man goes to concerts often.*

Table 6: Feature sets for the preposition (left) and the article (right) substitution models.

Prepositions		**Value**
bigram_left	Token bigram left of prep	erg vaak
bigram_right	Token bigram right of prep	concerten .
trigram_left	Token trigram left of prep	gaat erg vaak
trigram_right	Token trigram left of prep	concerten . $
PRE_verb	Verb preceding prep	gaat
FOLL_verb	Verb following prep	-
PRE_noun	Noun preceding prep	man
FOLL_noun	Noun following prep	concert
bigram_postags_left	POS bigram left of prep	adj adj
bigram_postags_right	POS bigram right of prep	noun punct
trigram_postags_left	POS trigram left of prep	ww adj adj
trigram_postags_right	POS trigram right of prep	noun punct $
FOLL_phr_head	Headword of following phrase	-
PRE_phr	Preceding phrase-type	mod
FOLL_phr	Following phrase-type	vaak

Articles
token bigram right
POS bigram right
token trigram right
POS trigram right
token bigram left
POS bigram left
token trigram left
POS trigram left
gender noun right
count noun right
token noun right
category of phrase of article
relation of phrase of article
category of parent phrase
relation of parent phrase

5. Results

The detection and selection models were trained on 2M and 20M feature vectors, respectively, for both phenomena. This difference in training size is due to the fact that the detection model is only a binary class and the selection model makes more complex discrepancies. For the detection model for articles, 74% of all training events were vectors of article absence, of which 6% were 'non-trivial'. 26% of all training events were vectors of article presence, of which 15% were 'non-trivial'. For prepositions, 71% of all training events were vectors of preposition absence, and 29% of preposition presence. The preposition models were subsequently tested on 268,895 feature vectors of unseen native data and on 1,499 feature vectors of the L2 data. Article models were tested on 250K unseen native test data, and similarly to what we have done for the models' descriptions, in reporting results and explaining evaluation strategies, we are mainly focusing on the preposition cases, highlighting differences wherever relevant. Actual figures are included for both phenomena. Performance is assessed using precision, recall, and f-score, calculated per class c as follows and then averaged:

$$prec_c = \frac{\#correct\ decisions_c}{\#positivedecisions_c} \qquad rec_c = \frac{\#correct\ decisions_c}{\#gold\ cases_c} \qquad fscore_c = \frac{2*prec_c*rec_c}{(prec_c+rec_c)}$$

5.1 Baselines

In order to have a lower bound to compare our system to, we devised a few baselines for each model, with increasing predictive power.

Detection model Baseline 1 is a most-frequent-class simple model always predicting no token for any given context. Baseline 2 is based on the 200 most common POS patterns for *token* presence cases: if a certain pattern has been observed in training data as more likely to occur in presence of a token rather than in its absence, this baseline will predict a token. Baseline 3 is as Baseline 2, but *all* POS patterns are considered. In both cases, POS patterns are based on trigrams. In fact, Baselines 2 and 3 are simpler versions of our model rather than pure baselines. For this reason, we decided to include only Baseline 1 in the article experiment.

Selection model Baseline 1 always predicts the preposition *van* or the article *de*, each being the most frequent value for their respective part of speech class. Baseline 2 is a basic classifier trained on the surrounding POS bigrams only.

5.2 Evaluation on native data

Table 7 outlines the results (including results for the baselines) on native test data for the preposition detection model, which only determines either the absence or the presence of a preposition. The results for the detection model are surprisingly high (but so are those for Baseline 3). This might be due to the fact that negative preposition vectors are all extracted in the same way (as explained in Section 4.1). In other words, it could depend on the way the data is represented. Future work will investigate this further, especially in terms of what exactly makes a non-trivial whitespace case. Results for the article detection model are given in Table 8 and Figure 2. Figure 2 shows that the non-trivial classes gain a lot of performance in recall but little in precision when more training data is added. They are much harder to identify than the trivial classes, which, based on the premise of how they are extracted (Section 4), is to be expected.
Table 9 shows the results on native test data for the selection model per preposition. Note that this only concerns the cases *with* a preposition.

As we can see from the tables, results show that the system is very accurate for some prepositions (*van*, *in*, *te*, *op*), while not so much for others (*als*, *bij*, *door*). Most prepositions with low scores seem to occur less than the more easily predicted prepositions (*als* and *bij*), though this is not always the case (*voor*, *met*). Conversely, the system performs quite well for the preposition *tot*, which is also quite infrequent. In order to identify how well performance for these prepositions develop over

Table 7: Preposition detection model's and baselines' performance on native test data.

	Precision	**Recall**	**F-score**
Detection Model	1.00	1.00	1.00
Baseline 1	0.25	0.50	0.33
Baseline 2	0.75	0.51	0.36
Baseline 3	0.97	0.97	0.97

Table 8: Article detection model's performance on native test data.

	Prec	**Rec**	**F-score**	**#cases**
False, non-trivial	0.75	0.31	0.43	11.4K
False, trivial	0.95	0.99	0.97	169K
True, non-trivial	0.79	0.30	0.43	10.8K
True, trivial	0.88	0.97	0.92	58.6K
Average	0.92	0.93	0.92	*Total*: 250K
Baseline	0.46	0.68	0.55	250K

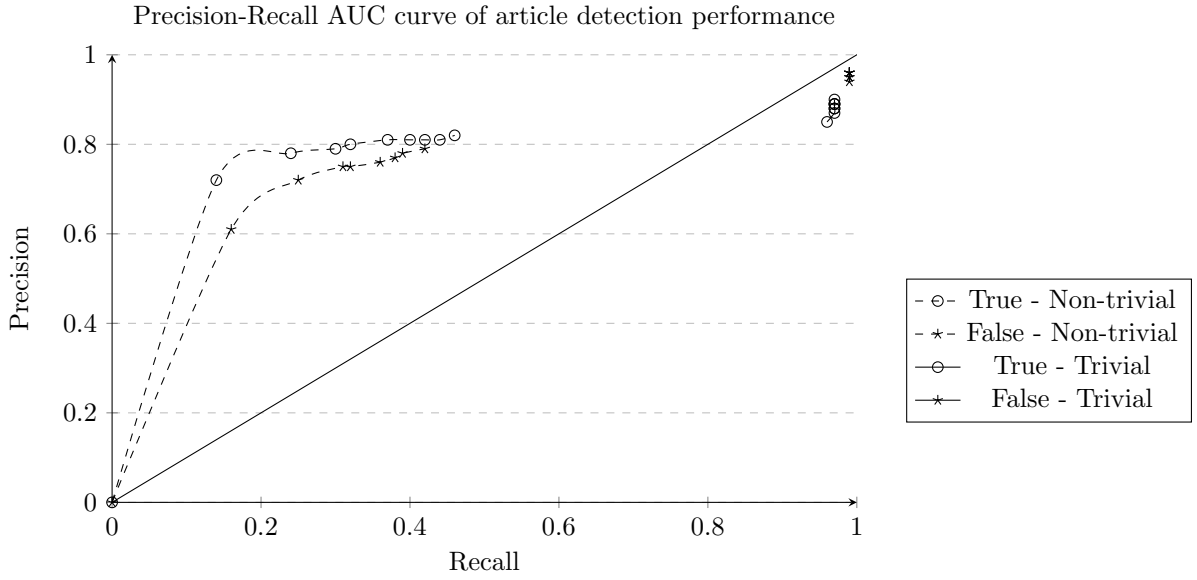

Figure 2: This graph illustrates the growth in precision and recall over various training sizes for the article detection task. The dots correspond to training sizes 250K, 1M, 2M, 2.5M, 5M, 7.5M, 10M, 15M and 20M respectively.

Table 9: Performance on native test data for each single preposition. The reported total for preposition is the number of actual occurrences of that string in the whole dataset.

Preposition	**Prec**	**Rec**	**F-score**	**# cases**
aan	0.72	0.69	0.71	11,690
als	0.81	0.51	0.63	5,157
bij	0.57	0.43	0.49	8,369
door	0.66	0.55	0.60	9,154
in	0.72	0.81	0.76	47,649
met	0.67	0.67	0.67	17,167
naar	0.69	0.62	0.65	6,232
om	0.77	0.69	0.73	10,881
op	0.78	0.78	0.78	23,111
over	0.65	0.59	0.62	6,036
te	0.97	0.97	0.97	19,746
tot	0.75	0.68	0.71	6,703
uit	0.69	0.56	0.62	6,514
van	0.80	0.87	0.83	71,102
voor	0.64	0.58	0.61	19,380
Average	0.75	0.75	0.75	*Total*: 268,895
Baseline 1	0.07	0.26	0.11	
Baseline 2	0.38	0.42	0.35	

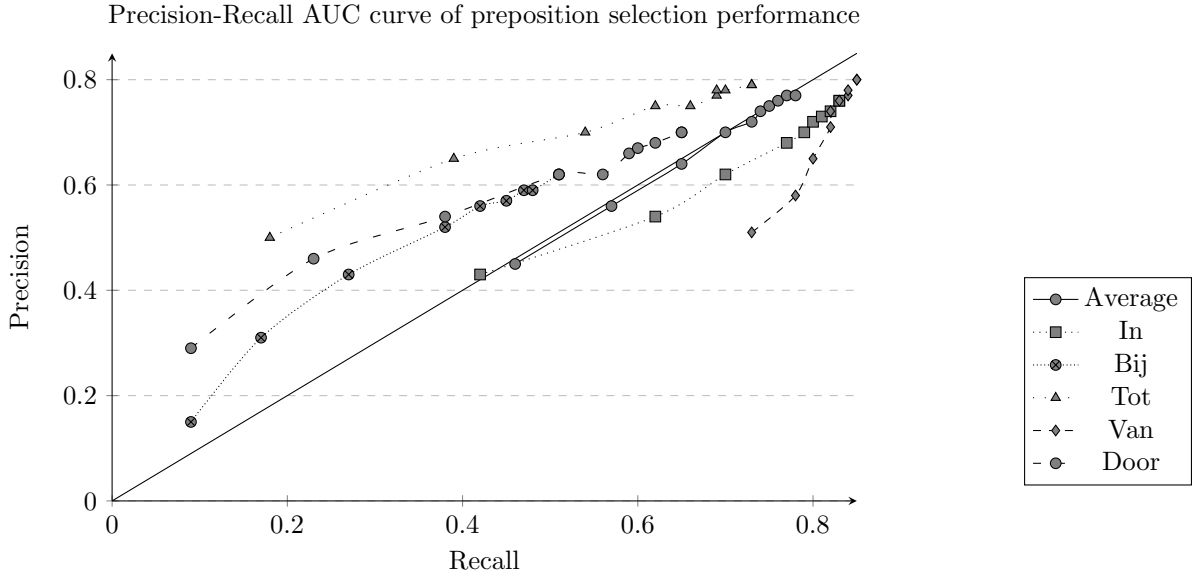

Figure 3: This graph illustrates the growth in precision and recall over various training sizes for the preposition selection task. The dots correspond to training sizes 20K, 50K, 250K, 1M, 2.5M, 5M, 10M, 15M and 20M respectively.

Table 10: Article selection model's performance

Article	**Prec**	**Rec**	**F-score**	**# cases**
de	0.92	0.95	0.93	140.6K
een	0.85	0.78	0.81	61.5K
het	0.90	0.92	0.91	47.9K
Average	0.90	0.90	0.90	*Total*: 250K
Baseline	0.32	0.56	0.41	

various training sizes, we plotted a precision-recall curve in Figure 3. This shows that prepositions *in* and *van* gain most in terms of **precision** with steep slopes to the right of the graph, generally below the reference curve. This means that recall is higher than precision. Adding training data thus reduces the amount of false positives for these cases. In other words: Other, less frequent classes are better informed so frequent classes are not 'overpredicted' as much. On the other hand, less frequent prepositions such as *bij* and *door* also profit in terms of **recall**, with a slope that more closely resembles that of the reference curve. These prepositions stay above this reference curve, which means that precision is higher than recall, indicating a higher degree of *false negatives* due to being 'underpredicted'. For every preposition, the dots are plotted closer after each gain in size, which indicates that performance is not likely to grow much more after the last training size of 20M. More discerning features could be the key to further increase performance.
We assume that learner data will contain more erroneous spelling but less variety in structure because of the fact that most data from the Leerdercorpus Nederlands consists of student essays. Erroneous spelling, poor lexical choice and grammatical errors will likely contribute to the results on learner test data.

5.3 Evaluation on learners' data

Because the learners' corpus is not error-annotated we collected human judgements via crowdsourcing. Annotators were asked to assess whether any given spot would contain a preposition and, if so, which one. More than one choice was allowed, so that a system's decision in the selection model is deemed correct if the predicted preposition is included in the set of those chosen by the annotators.

In order to gather annotations through crowdsourcing, we developed a web application that allowed users to create an account and annotate sentences written by L2 students, directly extracted by the Leerdercorpus Nederlands. Sentences of any length or structure were matched for extraction. An important reason to refrain from performing any preselection was to exploit the data in its actual form and distribution. For every sentence, a preposition (or white-space) was replaced by a question mark and participants were asked to select prepositions (from the fifteen prepositions that were included in the model) that would fit the context or indicate that the context should not have a preposition at all. Figure 4 shows a screenshot of the annotation interface. The crowdsourcing website was only used to gather data on prepositions and not articles, which were annotated by a native speaker and added to this project at a later stage. The reason we adopted a crowdsourcing solution was because it makes it possible to distribute the initial thesis project in an interactive way. Furthermore, this would make it possible to gather as many different answers as possible which would illustrate the difficulty of preposition usage as a whole.

5.3.1 Prepositions

The resulting annotated subset of the learners' data consists of 1,499 cases. These included 971 cases based on actual prepositions and picked randomly from the system's decisions, while ensuring that every preposition would be represented. The rest were attempts of the system to detect deletion errors, so that the presence of a preposition is the system's guess. In the test set, the annotators

Figure 4: Screenshot of the web-based annotation interface used for crowdsourcing native judgements on sentences extracted from the Leerdercorpus Nederlands.

identified 105 errors (7%). They included 72 substitution errors, 29 insertion errors and 4 deletion errors, as shown in Table 11. We do not report performance on correction for the baselines, as they have not been deployed in the same decision-making pipeline where both the detection and selection models were combined, but were only applied separately.

We also show the confusion matrix for the selection model in Table 12. Apart from the encouraging diagonal line showing the correct decisions, we cannot observe any specific confusion patterns. We hypothesise that this can be due to the varied nature of the native languages represented in the Leerdercorpus Nederlands, making it unlikely for any regular error trend to arise. A more focused analysis on a single native language might show a different picture in this sense, but is left to future work.

5.3.2 Articles

For detection, the model performs much worse than L2 students (75% for positive and 17% for negative instances, vs. 92% and 90% respectively for L2 learners), when compared to gold standard annotation. Therefore, we can say that the detection model, for both phenomena, is far from ready. This is in line with the results on L1 data for the same task – where non-trivial cases already proved difficult to identify. Human annotation identified 47 insertion errors and 33 deletion errors. Keeping the 25 substitution errors in mind, this distribution is very different from that of the preposition errors. Article selection is reported at an F-score of 84%, down from its 90% on L1 data. There were 25 substitution errors in an annotated set of about 750 cases across 279 sentences, and 11 of these were spotted. Even though this seems unimportant or underperforming at first, it is encouraging that the system performs well when identifying **correct** articles.

6. Discussion and Outlook

With an F1-score of 75% on L1 data for prepositions, and results generally above the baselines, we show that it is possible to build a fairly accurate model of fifteen Dutch prepositions. The considerably lower results on L2 data indicate that the differences between native and learner language are substantial, especially because native data does not provide any information on the types of errors

Table 11: Results on learners' data for prepositions.

	Detection		Correction	
	Prec	**Rec**	**Accuracy**	**N (gold)**
Substitution	0.14	0.90	0.60	72
Insertion	0.22	0.21	–	29
Deletion	0.01	0.75	1.00	4
Correct	0.99	0.36	–	1,394
Average/Total	0.91	0.38	0.62	1,499
Baseline 1	0.14	0.37	–	–
Baseline 2	0.39	0.37	–	–
Baseline 3	0.43	0.55	–	–

Table 12: Confusion matrix for model selection (columns) and gold standard annotation (rows) in learners' test data.

	aan	**als**	**bij**	**door**	**in**	**met**	**naar**	**om**	**op**	**over**	**te**	**tot**	**uit**	**van**	**voor**
aan	**24**	0	2	2	4	7	0	1	0		0	0	1	0	4
als	1	**1**	2	1	3	0	0	1	1	2	0	1	0	0	2
bij	2	0	**14**	2	9	3	0	1	4	0	0	0	0	5	2
door	1	0	3	**2**	3	2	0	3	1	1	0	0	2	0	1
in	2	0	5	0	**117**	4	0	1	5	0	0	0	2	5	3
met	0	0	5	1	15	**41**	1	7	2	3	0	0	0	6	1
naar	1	0	1	0	2	1	**12**	0	2	0	0	0	0	2	0
om	0	0	1	0	8	3	1	**35**	0	2	0	1	0	2	1
op	1	1	1	0	12	0	3	1	**48**	1	0	0	1	2	2
over	1	0	2	1	6	3	0	3	1	**20**	0	0	0	0	1
te	1	0	0	0	2	0	0	0	1	4	**87**	0	0	2	2
tot	0	0	0	1	0	1	0	3	3	0	0	**5**	0	0	0
uit	0	0	2	0	1	0	0	0	0	0	0	1	**7**	0	1
van	2	0	2	2	15	6	1	2	3	3	0	0	0	**93**	8
voor	5	0	8	3	8	12	0	8	3	0	2	0	1	4	**70**

and confusions that learners make (Han et al. 2010). The detection model suffers mostly from the transition from L1 to L2 test data. The model is clearly too biased towards preposition presence, which makes little sense, as the chances that any given whitespace contains a preposition are intuitively low, even if the POS pattern is commonly found with a preposition. This is amplified by the fact that we tried to capture as many deletion errors as possible, which also maximises the amount of false positives. Insertion error detection and correction are both rather low, yet promising. Most insertion errors were confused with substitution errors, another consequence of the detection model because it would rather see a preposition than none. The high recall for substitution is encouraging, and the low precision is worrying; they are both due to the fact that the selection model picks one preposition only, even if other candidates might be equally acceptable. For articles, the selection performance on L2 data is still strong at 84%, although this is expected because there are fewer classes. However, detecting error types still proves to be difficult. In many cases, the model still marks certain decisions as wrong when they should be right (and vice versa), especially through decisions by the detection model. This shows all the more that modelling 'nothing' is difficult, and errors that are related to **not** using words of very important lexical building blocks are even more difficult to spot for a system. In short, it is very difficult to model how learners might misuse or forget articles using only native data.

For prepositions, one logical next step is rebuilding the selection procedure into a *multilabel* model, which would reduce the number of false positives, and also improve the learner's experience by suggesting more than one option to a (likely) wrongly used preposition. Multiple suggestions would reduce the number of false positives when detecting substitution errors (De Felice and Pulman 2009). Adding more information to the model, such as semantic features relative to word classes, for example, could also yield better results. For example, features could be inherited by similar words, so as to cope with unseen nouns or verbs. To this end, similarity information could be exploited either in the form of simple synonym lookup via lexical resources such as WordNet (Fellbaum 1998), or via corpus-derived distributional information in the form, for example, of word embeddings (Mikolov et al. 2013).

However, a prime direction for improvement is to make the model more familiar with L2 language. One simple first step could just be using a spell corrector to improve the quality of L2 data and overall preprocessing. Linguistically-informed features on learners' use of prepositions should also be investigated, though the variety of native languages in the L2 corpus does not make this straightforward. Another interesting possibility for investigating in this context would be to add artificially generated errors to the native training corpora (as suggested in (Rozovskaya and Roth 2010)), so as to reduce the differences between native and learner language. Specifically, (Rozovskaya and Roth 2010) show that introducing artificial errors in article usage in English in the training data increases results on learner test data. They make sure that the distribution of these errors is similar to the distribution of errors in ESL (English as a Second Language) data. This could prove very useful as it teaches the system what kind of errors can be expected and which errors are rare, and might be especially interesting for the case of prepositions as they are a closed class but with many possible different confusions.

As far as the corpus itself is concerned, the Leerdercorpus Nederlands proved to be useful and promising for this task because it features typical learner errors. There is much more data in the corpus which we did not annotate in the short amount of time we had. A next step for the Leerdercorpus Nederlands would be to make it error-annotated so tasks on automatic error correction or language acquisition can be tackled more conveniently.

Acknowledgements

We would like to sincerely thank Liesbeth Degand for providing us with the Leerdercorpus Nederlands, and those who partook in annotating the L2 data. We are also grateful to those who attended our talk at CLIN 26 in Amsterdam for their interesting and helpful comments on an earlier version

of this work. Finally, we would also like to thank the CLIN Journal reviewers for their helpful comments and suggestions that have doubtlessly contributed to the quality of this paper.

References

Chodorow, Martin, Joel R. Tetreault, and Na-Rae Han (2007), Detection of grammatical errors involving prepositions, *Proceedings of the 4th ACL-SIGSEM Workshop on Prepositions* pp. 25–30.

Dale, Robert, Ilya Anisimoff, and George Narroway (2012), Hoo 2012: A report on the preposition and determiner error correction shared task, *The 7th Workshop on the Innovative Use of NLP for Building Educational Applications* pp. 54–62.

De Felice, Rachele and Stephen G. Pulman (2008), A classifier-based approach to preposition and determiner error correction in l2 english, *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 169–176.

De Felice, Rachele and Stephen Pulman (2009), Automatic detection of preposition errors in learner writing, *CALICO Journal* pp. 512–528.

Donaldson, Bruce (2008), *Dutch: A Comprehensive Grammar*, Routledge.

Eeg-Olofsson, Jens and Ola Knutsson (2003), Automatic grammar checking for second language learners the use of prepositions, *Proceedings of Nodalida 2003*.

Fellbaum, Christiane (1998), *WordNet: An Electronic Lexical Database*, Bradford Books.

Gamon, Michael, Jianfeng Gao, Chris Brockett, Alexandre Klementiev, William B. Dolan, Dmitriy Belenko, and Lucy Vanderwende (2008), Using contextual speller techniques and language modeling for ESL error correction, *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I*, pp. 449–456.

Han, Na-Rae, Joel Tetreault, Soo hwa Lee, and Jin young Ha (2010), Using an error-annotated learner corpus to develop and esl/efl error correction system, *Prcoeedings of LREC 2010*.

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

Liu, Yingliang (2008), The effects of error feedback in second language writing, *Arizona Working Papers in SLA & Teaching* **15**, pp. 65–79.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013), Efficient estimation of word representations in vector space, *Proceedings of the ICLR Workshop*, pp. 1–12.

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

Ng, Hwee Tou, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault (2013), The conll-2013 shared task on grammatical error correction, *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, Association for Computational Linguistics, Sofia, Bulgaria, pp. 1–12.

Nicholls, D. (2003), The Cambridge learner corpus—Error coding and analysis for lexicography, *in* Archer, D., P. Rayson, A. Wilson, and T. McEnery, editors, *Proceedings of the Corpus Linguistics 2003 Conference*, UCREL, Lancaster University, Lancaster, UK, pp. 572–581.

Perrez, Julien and Liesbeth Degand (2009), Het Leerdercorpus Nederlands in theorie en in de praktijk, *IVN - 17e Colloquium Neerlandicum*.

Rozovskaya, Alla and Dan Roth (2010), Training paradigms for correcting errors in grammar and usage, *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, Los Angeles, California, pp. 154–162.

Tetreault, Joel R. and Martin Chodorow (2008), The ups and downs of preposition error detection in ESL writing, *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 865–872.

van Noord, Gertjan (2006), At Last Parsing Is Now Operational, *TALN 2006 Verbum Ex Machina, Actes De La 13e Conference sur Le Traitement Automatique des Langues naturelles*, Leuven, pp. 20–42.

van Noord, Gertjan, Gosse Bouma, Frank Van Eynde, Daniel De Kok, Jelmer Van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste (2013), Large scale syntactic annotation of written Dutch: Lassy, *Essential Speech and Language Technology for Dutch*, Springer, pp. 147–164.

van Noord, Gertjan, Ineke Schuurman, and Gosse Bouma (2011), Handleiding lassy syntactische annotatie, *Technical report*, University of Groningen.