Computational Linguistics in the Netherlands Journal 10 (2020) 19–36 Submitted 10/2020; Published 12/2020

Automatically Correcting Dutch Pronouns *Die* and *Dat*

Liesbeth Allein* LIESBETH.ALLEIN@KULEUVEN.BE
Artuur Leeuwenberg** A.M.LEEUWENBERG-15@UMCUTRECHT.NL
Marie-Francine Moens* SIEN.MOENS@KULEUVEN.BE

**Department of Computer Science, KU Leuven, Leuven, Belgium*

***Julius Center, University Medical Center Utrecht, Utrecht University, Utrecht, The Netherlands*

Abstract

The correct use of Dutch pronouns *die* and *dat* is a stumbling block for both native and non-native speakers of Dutch due to the multiplicity of syntactic functions and the dependency on the antecedent's grammatical gender and number. Drawing on previous research on neural context-dependent dt-mistake correction (Heyman et al. 2018), this study constructs the first neural network model for Dutch demonstrative and relative pronoun correction that specifically focuses on *die* and *dat*. Several datasets are built with sentences obtained from the Dutch Europarl corpus (Koehn 2005) - which contains the proceedings of the European Parliament from 1996 to the present - and the SoNaR corpus (Oostdijk et al. 2013) - which contains Dutch texts from a variety of domains such as newspapers, blogs and legal texts. Firstly, a binary classification model solely predicts the correct *die* or *dat*. The classifier with a bidirectional long short-term memory architecture achieves 84.56% accuracy. Secondly, a multitask classification model simultaneously predicts the correct *die* or *dat* and its part-of-speech tag. The model consisting of a sentence and context encoder with both a bidirectional long short-term memory architecture yields 88.63% accuracy for *die/dat* prediction and 87.73% accuracy for part-of-speech prediction. More evenly-balanced data, larger word embeddings, an extra bidirectional long short-term memory layer and integrated part-of-speech knowledge positively affects *die/dat* prediction performance, while a context encoder architecture raises part-of-speech prediction performance. This study shows promising results and can serve as a starting point for future research on automated, fine-grained grammar correction.

1. Introduction

Following previous research on automatic detection and correction of dt-mistakes in Dutch (Heyman et al. 2018), this paper investigates another stumbling block for both native and non-native speakers of Dutch: the correct use of *die* and *dat*. Both words can function as demonstrative and relative pronouns, while *dat* can also function as a subordinating conjunction. The choice between pronouns *die* and *dat* is related to the choice between Dutch definite determiners *de* and *het*. Masculine, singular nouns and all plural nouns take *de* and *die*, while *het* and *dat* can only be used with neuter, singular nouns. Acquiring the target grammar for definite determiners is a slow process with first-language learners of Dutch (i.e. not before the age of six) and an even slower process with second-language learners (Cornips and Hulk 2006). Incorrect gender estimations of nouns can lead to incorrect determiner-noun agreements. Analogous to this, wrongly used *die* and *dat* are often the result of incorrect pronoun-antecedent agreements following incorrect antecedent gender estimations. The multiplicity of syntactic functions and the dependency on the antecedent's grammatical gender and number thus make *die/dat* correction a challenging task for both human and computer.

The grammar concerning *die* and *dat* is threefold. Firstly, they can be used as dependent or independent demonstrative pronouns (*aanwijzend voornaamwoord*), with the first replacing the article before the noun it modifies (1-3) and the latter being a noun phrase that refers to a preceding/following noun phrase (4-5) or sentence (6). The choice between *die* and *dat* depends on the gender and number of the antecedent: *dat* refers to neuter, singular nouns (1, 5) and sentences (6), while *die* refers to masculine, singular nouns (2) and plural nouns independent of their gender (3-4).

1. “Ik zie **dat** meisje het huis binnengaan.” - antecedent: neuter, singular noun
   “*I see **that** girl entering the house.*”

2. “Ik zie **die** jongen het huis binnengaan.” - antecedent: masculine, singular noun
   “*I see **that** boy entering the house.*”

3. “Ik zie **die** kinderen het huis binnengaan.” - antecedent: plural noun
   “*I see **those** children entering the house.*”

4. “Er lopen daar kinderen. Ik zie **die** het huis binnengaan.” - antecedent: plural noun
   “*There are children running over there. I see **them** entering the house.*”

5. “Er staat daar een reuzenrad. Het pretpark heeft **dat** vorig jaar aangekocht.” - antecedent: neuter, singular noun
   “*There is a Ferris wheel over there. The amusement park bought **that** last year.*”

6. “Leen komt niet. Ik heb **dat** meermaals gezegd vandaag.” - antecedent: sentence
   “*Leen is not coming. I have said **that** several times today.*”

Secondly, *die* and *dat* can be used as relative pronouns (*betrekkelijk voornaamwoord*) introducing relative clauses, which provide additional information about the directly preceding antecedent it modifies. Similar rules as for demonstrative pronouns apply: masculine, singular nouns (7) and plural nouns (8) are followed by relative pronoun *die*, neuter singular nouns (9) by *dat*.

7. “Ik zie de jongen **die** het huis binnengaat.” - antecedent: masculine, singular noun
   “*I see the boy **who** enters the house.*”

8. “Ik zie de kinderen **die** het huis binnengaan.” - antecedent: plural noun
   “*I see the children **who** enter the house.*”

9. “Ik zie het meisje **dat** het huis binnengaat.” - antecedent: neuter, singular noun
   “*I see the girl **who** enters the house.*”

Lastly, *dat* can be used as a subordinating conjunction (*onderschikkend voegwoord*) introducing a subordinating clause (10). In contrast to the pronominal use of *die* and *dat*, *dat* as subordinating conjunction does not depend on the gender and number of any antecedent. It is therefore less likely that someone would then use *die* instead of *dat*. However, we do not omit subordinating conjunction *dat* from the datasets, as not every dataset includes part-of-speech (POS) tags that can be used to remove sentences with a subordinating conjunction *dat* from the training, validation and test sets. We hypothesize that our model will be able to implicitly learn the syntactic functions or POS of *die* and *dat* in a given sentence, and consistently predict *dat* in case of a subordinating conjunction. A brief overview of the grammar is given in Table 1.

10. “Ik zie **dat** de jongen het huis binnengaat.”
    “*I see **that** the boy enters the house.*”

In this paper, we aim at developing a classification model that automatically predicts and corrects *die* and *dat* instances in texts. We approach the *die/dat* correction task as a highly specialized word prediction task: the model takes as input a sentence in which a single *die* or *dat* is replaced by a specific prediction token and returns the *die* or *dat* label with the highest probability. Subsequently, we transform the classification model to a multitask classification model that jointly predicts the correct *die/dat* label and its POS tag. We hypothesize that integrating the POS prediction task in the original *die/dat* classification model will positively influence *die/dat* prediction. This paper reports work conducted as part of a master thesis (Allein 2019). Our newly-proposed *die/dat* prediction task has already been used as a downstream task for evaluating a Dutch language model (Delobelle

Function	**Demonstrative pronoun**	**Relative pronoun**	**Subordinating conjunction**
Refer to antecedent			
singular, masculine noun	*die*	*die*	-
singular, neuter noun	*dat*	*dat*	-
plural noun	*die*	*die*	-
sentence	*dat*	-	-
Introduce subordinating clause	-	-	*dat*

Table 1: Grammar concerning *die* and *dat*.

et al. 2020), which benchmarks against our model as presented in a preprint version of this paper (Allein et al. 2020). The remainder of the paper is structured as follows: we explore existing work in scientific literature in Section 2, present the model architecture of the binary and multitask classification models in Section 3, state the experimental setup and results in Section 4 and 5, analyze those results in Section 6, perform an ablation study and error analysis in Section 7 and 8, and finally conclude the paper in Section 9.

2. Related Work

Previous works on automatic Dutch language correction have differentiated between spelling and grammar errors. Concerning spelling correction, the CLIN28 shared task challenged participants to construct Dutch spell checkers that automatically correct common spelling mistakes in Dutch Wikipedia texts (Beeksma et al. 2018). The list of spelling mistakes comprised of real-world confusions, split/runon errors, missing/redundant words and punctuation, capitalisation errors, archaic spelling and non-word errors. Tijhuis (2014) focused on forgotten diacritics, among others. Sevens et al. (2016) found that internet users with intellectual disabilities profoundly make spelling errors caused by phonetic confusion, and applied character-based fuzzy matching to correct those spelling mistakes. While spell checkers tackle a range of spelling errors, grammar correction systems detect and correct errors on a more fine-grained level. Stehouwer and van den Bosch (2008) and Heyman et al. (2018), for example, strongly focused on context-dependent dt-mistakes, with the first applying a IGTree classifier and the latter an attention-based neural network to predict a verb's suffix. van den Bosch and Berck (2012) combined four classifiers to detect and predict correct prepositions and determiners. Kloppenburg and Nissim (2016), on the other hand, used a linear SVM to detect wrongly inserted and deleted prepositions and determiners, and trained a second linear SVM to select the correct replacement.

In this paper, we construct a specialized grammar error detection and correction system that automatically predicts *die* and *dat* labels for masked tokens. We are - to our knowledge - the first to focus on this specific kind of error correction.

3. Model Architecture

We inspire the architecture of our binary and multitask models on the dt-correction model presented in Heyman et al. (2018). We opt for a similar architecture, as it has proven to be highly effective for the dt-correction task, in which the suffix of the finite verb depends on the number and position of the subject. In the dt-correction model, a context encoder (bidirectional LSTM) and verb encoder (LSTM) respectively encode a given sentence, in which the finite verb is replaced by its stem, and the finite verb. The resulting context and verb representations are then concatenated and fused by a feedforward neural network. Ultimately, the model returns the correct suffix for the finite verb.

3.1 Binary Classification Model

For the binary classification model that predicts the correct *die* or *dat* for each sentence, a bidirectional Long-Short Term Memory (BiLSTM) neural network is deployed. As the antecedent can be rather distant from its modifying pronoun due to adjectives and sentence boundaries, an LSTM architecture is chosen over a regular Recurrent Neural Network as the latter does not cope well with learning non-trivial long-distance dependencies (Chiu and Nichols 2016). Furthermore, a BiLSTM is chosen over a single left-to-right LSTM, because the antecedent can be either before or after the *die* or *dat*. The architecture of the binary classification model is provided in Fig. 1. An input sentence is first sent through an embedding layer where each token is transformed to a 100-dimensional word embedding which has been initially trained on the dataset of sentences containing at least one *die* or *dat* using the Word2Vec Skip-gram model (Mikolov et al. 2013). The weights of the embedding layer are trainable. The word embeddings are then sent through a BiLSTM layer. The BiLSTM concatenates the outputs of two LSTMs: the left-to-right $LSTM_{forward}$ computes the states $\overrightarrow{h_1}..\overrightarrow{h_N}$ and the right-to-left $LSTM_{backward}$ computes the states $\overleftarrow{h_N}..\overleftarrow{h_1}$. This means that at time t for input x, represented by its word embedding $E(x)$, the bidirectional LSTM outputs the following:

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}]^1 \tag{1}$$

$$\overrightarrow{h_t} = LSTM_{forward}(\overrightarrow{h_{t-1}}, E(x_t)) \tag{2}$$

$$\overleftarrow{h_t} = LSTM_{backward}(\overleftarrow{h_{t+1}}, E(x_t)) \tag{3}$$

Next, the concatenated output is sent through a maxpooling layer, a linear layer and, eventually, a softmax layer that generates a probability distribution over the *die/dat* labels. To prevent model overfitting and co-adaptation, dropout regularization is implemented in the embedding layer and the linear layer. In both layers, dropout is set to $p = 0.5$, which randomly zeroes out nodes in each layer using samples from a Bernoulli distribution. During training, the binary cross-entropy loss is minimized:

$$BCE_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(\hat{y}_i)) + (1 - y_i) \cdot log(1 - p(\hat{y}_i)) \tag{4}$$

where y_i is the ground truth label (0 for *dat* and 1 for *die*) and $p(\hat{y}_i)$ is the probability of the predicted label for all N training examples.

3.2 Multitask Classification Model

We introduce POS knowledge in the *die/dat* classification model by transforming it to a multitask classification model which performs two distinct prediction tasks: the binary classification of *die* and *dat*, and the prediction of three POS tags, namely *subordinating conjunction*, *relative pronoun* and *demonstrative pronoun*. We hypothesize that integrating this POS prediction task will positively influence *die/dat* prediction performance, as the model will be more grammar-aware. We experiment with two different model architectures (Fig. 2), with the second architecture laying more focus on the immediate context surrounding the *die/dat* token. For the BiLSTM model, the first layer is the embedding layer in which the weights are initialized by means of the 200-dimensional pre-trained embedding matrix. The embedding weights are updated after every epoch. The second layer consists of two bidirectional LSTMs where the output of the first BiLSTM serves as input to the second BiLSTM. This layer has dropout regularization $p = 0.2$. The two-layer BiLSTM layer concatenates the outputs at time t into a 64-dimensional vector and sends it through a maxpooling layer. Until this point, the two tasks share the same parameters. The model then splits into

1. [;] denotes concatenation

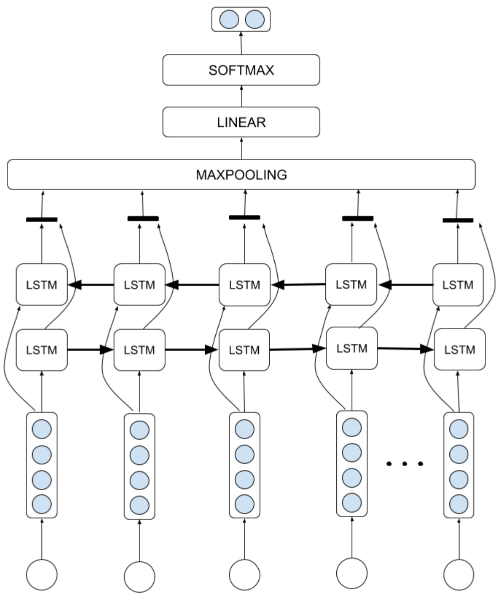
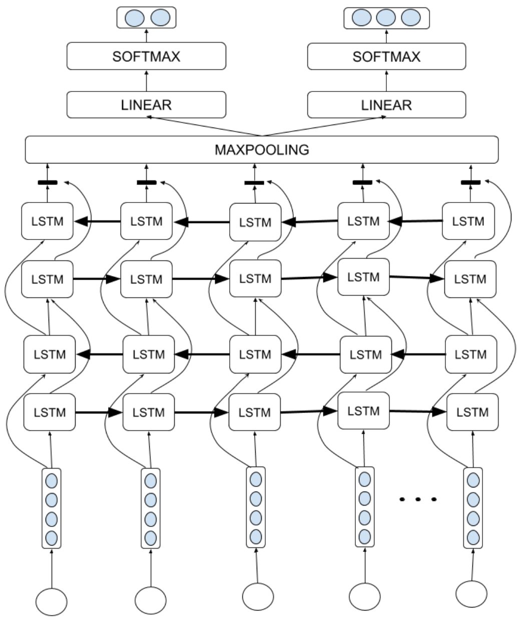
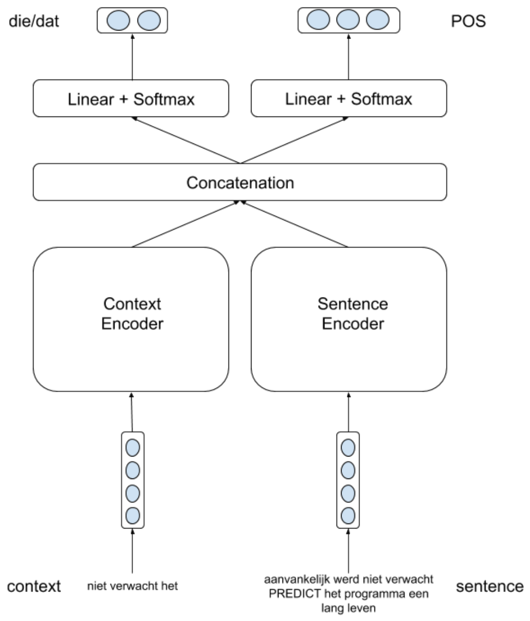


Figure 1: Model architecture of the binary classification model.



(a) BiLSTM model



(b) Context + Sentence Encoder

Figure 2: Overview of the two multitask classification model architectures.

two separate linear layers. The left linear layer transforms the 64-dimensional vector to a two-dimensional vector on which the softmax is computed. That softmax layer outputs the probability distribution over the *dat* and *die* labels. The right linear layer transforms the 64-dimensional vector to a three-dimensional vector to which a softmax function is applied. The softmax layer outputs the probability distribution over the *subordinating conjunction*, *relative pronoun* and *demonstrative pronoun* labels. The second multitask classification model takes the immediate context around the 'PREDICT' token (two tokens before and one token after) as additional input. Both the windowed sentence and context are first transformed into their word embedding representations. They are then sent through a sentence encoder and context encoder, respectively. The sentence encoder has the same architecture as the second and third layer of the BiLSTM model, namely a two-layer BiLSTM and a maxpooling layer. For the context encoder, we experiment with two different architectures: a feedforward neural network and a one-layer BiLSTM with dropout $p = 0.2$, both followed by a maxpooling layer. Both sentence and context encoder output a 64-dimensional vector which are, consequently, concatenated to a 128-dimensional vector. As in the BiLSTM model, the resulting vector is sent through two separate linear layers that output a probability distribution over the *die/dat* and POS labels, respectively. For the *die/dat* prediction task, binary cross-entropy is minimized. For the POS prediction task, cross-entropy is minimized:

$$CE(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c} log(p_{i,c}) \tag{5}$$

where N is the number of training examples, C is the number of classes (in this case three), $y_{i,c}$ is the binary indicator (1 or 0) marking whether class label c is the correct label for input sentence i or not, and p is the probability of sentence i having class label c.

4. Experimental Setup

4.1 Datasets

The datasets used for training, validation and testing contain sentences extracted from the Europarl corpus (Koehn 2005) and SoNaR corpus (Oostdijk et al. 2013). The Europarl corpus is an open-source parallel corpus containing proceedings of the European Parliament. The Dutch section consists of 2,333,816 sentences and 53,487,257 words. The SoNaR corpus comprises two corpora: SONAR500 and SONAR1. The SONAR500 corpus consists of more than 500 million words obtained from different domains. Examples of text types are newsletters, newspaper articles, legal texts, subtitles and blog posts. All texts except texts from social media have been automatically tokenized, POS tagged and lemmatized. The SONAR500 corpus contains significantly more data and more varied data than the Europarl corpus. Due to the high amount of data in the corpus, only three subparts are used: Wikipedia texts, reports and newspaper articles. These subparts are chosen because the number of wrongly used *die* and *dat* is expected to be low.

4.2 Preprocessing

The sentences in the Europarl corpus are tokenized and parsed using the Dutch version of TreeTagger (Schmid 1994). Only sentences which contain at least one *die* or *dat* are extracted from the corpora. Subsequently, each single occurrence of *die* and *dat* is detected and replaced by a unique token ('PREDICT'). When there are multiple occurrences in one sentence, only one occurrence is replaced at a time. Consequently, a sentence can appear multiple times in the training and test dataset - with the unique token for *die* and *dat* at a different place in the sentence. Each sentence is paired with its automatically assigned ground-truth label for *die* and *dat.* The resulting datasets consist of 103,871 (Europarl) and 1,269,091 (SoNaR) sentences. The Europarl dataset, on the one hand, contains 70,057 *dat*-labeled and 33,814 *die*-labeled sentences. The SoNaR dataset, on the

Dataset	# sentences	dat/ die	subordinating conjunction/ relative pronoun/ demonstrative pronoun
Europarl	103,871	70,057/ 33,814	- -
SoNaR	1,269,091	736,987/ 532,104	407,848/ 387,292/ 473,951

Table 2: Overview of datasets.

other hand, has more than ten times the number of labeled sentences - with 736,987 *dat*-labeled and 532,104 *die*-labeled. Considering the imbalance in both datasets, it may be argued that *dat* occurs more frequently than *die* due to its syntactic function as subordinating conjunction and not its use as demonstrative pronoun, as it can only refer to singular, neuter nouns. As for the multitask classification model, the POS tags for *die* and *dat* present in the SoNaR corpus are extracted and stored as ground-truth labels: 407,848 *subordinating conjunction*, 387,292 *relative pronoun* and 473,951 *demonstrative pronoun.* From a brief qualitative assessment on the POS tags for *die* and *dat* in both corpora, the POS tags in the SoNaR corpus appear to be more reliable than the POS tags generated by TreeTagger in the Europarl corpus. Therefore, only the SoNaR dataset is used for training and testing the multitask classification models. An overview of the datasets after preprocessing is given in Table 2.

4.3 Experimental Setup

Each dataset is randomly divided into a training (70%), validation (15%) and test set (15%). The data is fed to the model in batches of 128 (binary classification model) or 516 (multitask classification model) samples, and is reshuffled at every epoch. For the *die/dat* prediction task in both the binary and multitask classification models, model weights are optimized using Stochastic Gradient Descent with learning rate = 0.01 and momentum = 0.9. For the POS prediction task in the multitask classification model, model weights are optimized using Adam optimization with learning rate = 0.0001. The data is fed to the model in 24 epochs (binary classification model) and 35 epochs (multitask classification model).

5. Results

5.1 Binary Classification Model

An overview of the performance results is given in Table 3. We compare model performance when trained and tested on the two corpora individually and experiment with different settings of the two corpora in order to investigate the effect of dataset alterations on model performance. There are three settings: *full* in which the datasets contain full sentences, *windowed* in which sentences are windowed around the unique prediction token without exceeding sentence boundaries (max. five tokens before and after the token, including token), and *windowed no_boundaries* in which the windows can exceed sentence boundaries. When limiting the input sentences to windowed sentences in the Europarl corpus (2), model performance increases significantly on all metrics, especially for *die* prediction performance. As a consequence, we start training and testing the model on the SoNaR dataset using the windowed sentences (3) instead of the full sentences, as we assume that similar performance differences would be reported for the SoNaR dataset. This assumption is later supported by the performance differences for *die/dat* classification between full and windowed SoNaR sentences in the multitask classification model (Table 4). The difference in model performance when trained and tested on the Europarl (2) and SoNaR (3) windowed datasets is particularly noticeable in the precision, recall and F1 scores. Model performance for *dat* prediction is better for the Europarl dataset than for the SoNaR dataset, while model performance for *die* prediction is notably better for the SoNaR dataset than for the Europarl dataset. Lastly, a change in windowing seems to have a positive impact on the overall model performance: the model trained and tested on the SoNaR dataset with windows exceeding sentence boundaries (4) outperforms the model trained and tested on the SoNaR dataset with windows within sentence boundaries (3) on every metric.

Binary Classification Model					
Dataset	**Accuracy**	**Balanced Accuracy**	**Precision *dat/die***	**Recall *dat/die***	**F1 *dat/die***
Model 1: BiLSTM					
Europarl, *full* (1)	75.03%	68.49%	78.11%/ 65.68%	87.45%/ 49.54%	82.41%/ 56.05%
Europarl, *windowed* (2)	**83.27%**	**80.70%**	**87.19%/ 74.97%**	**88.14%/ 73.26%**	**87.58%/ 73.83%**
SoNaR, *windowed* (3)	82.34%	81.72%	85.35%/ 77.94%	84.94%/ 78.50%	85.06%/ 78.05%
SoNaR, *windowed no_boundaries* (4)	**84.56%**	**84.18%**	**87.71%/ 80.13%**	**86.16%/ 82.20%**	**86.85%/ 80.99%**

Table 3: Performance results of the binary classification model on the Europarl dataset containing full sentences (1), the Europarl dataset containing windowed sentences within sentence boundaries (2), the SoNaR dataset containing windowed sentences within sentence boundaries (3) and the SoNaR dataset containing windowed sentences exceeding sentence boundaries (4).

5.2 Multitask Classification Model

An overview of the performance results for *die/dat* prediction is given in Table 4. The same dataset settings as for the binary classification model are used: *full* in which the datasets contain full sentences, *windowed* in which sentences are windowed around the unique prediction token without exceeding sentence boundaries (max. five tokens before and after the token, including token), and *windowed no_boundaries* in which the windows can exceed sentence boundaries. As mentioned in section 4.2, we only use the SoNaR dataset. The multitask classification models generally perform better with the *windowed* and *windowed no_boundaries* dataset settings for *die/dat* prediction. Concerning the model architectures, it can be argued that altering the model architecture has no large impact on model performance for *die/dat* prediction. However, changing the model architecture from an architecture with merely a sentence encoder (Model 2) to an architecture with both a sentence and a context encoder (Model 3 and 4) does have a stronger positive impact on model performance for POS prediction (Table 5). For the POS prediction task, the multitask classification model with a BiLSTM context encoder (Model 4) trained and tested on *windowed* SoNaR sentences reaches best performance results on almost all evaluation metrics.

6. Discussion

Overall, the best performing multitask classification model (Model 4) is able to predict *die* and *dat* more accurately than the binary classification model (Model 1). We can draw several conclusions

Multitask Classification Model					
Dataset	**Accuracy**	**Balanced Accuracy**	**Precision *dat/die***	**Recall *dat/die***	**F1 *dat/die***
Model 2: BiLSTM					
SoNaR, *full*	78.52%	77.56%	81.59%/ 73.87%	82.60%/ 72.52%	82.06%/ 73.14%
SoNaR, *windowed*	86.36%	85.08%	86.26%/ 86.53%	**91.73%**/ 78.44%	88.89%/ 82.25%
SoNaR, *windowed, no_boundaries*	88.36%	88.15%	**91.05%**/ 84.59%	89.24%/ **87.06%**	90.12%/ 85.77%
Model 3: Feedforward Context Encoder					
SoNaR, *windowed*	88.16%	87.79%	90.37%/ 84.93%	89.70%/ 85.88%	90.02%/ 85.37%
SoNaR, *windowed, no_boundaries*	88.36%	88.14%	90.99%/ 84.66%	89.31%/ 86.97%	90.13%/ 85.77%
Model 4: BiLSTM Context Encoder					
SoNaR, *windowed*	88.63%	87.93%	89.58%/ **87.15%**	91.58%/ 84.28%	90.55%/ 85.66%
SoNaR, *windowed no_boundaries*	**88.85%**	**88.51%**	90.95%/ 85.83%	90.29%/ 86.73%	**90.60%**/ **86.25%**

Table 4: Performance of the three multitask classification models for *die/dat* prediction.

following an analysis of the performance metric results for *die/dat* prediction. Firstly, the model consistently predicts *dat* more precisely than *die*. This could be due to the higher number of *dat* instances in both corpora: 67.45% *dat* labels in Europarl, 58.07% *dat* labels in SoNaR. When taking into account the different label distributions of the two corpora, we can argue that the binary classification model's performance differences between *die* and *dat* prediction are inversely proportional to a dataset's label distribution: the differences are considerably smaller when the model is trained on a more evenly balanced SoNaR-based dataset than when it is trained on a more imbalanced Europarl-based dataset. Secondly, *die/dat* prediction performance increases for the binary classification model (Model 1) and the BiLSTM multitask classification model (Model 2) when the window over the sentences is not limited to sentence boundaries (SoNaR *windowed no_boundaries*). It could be argued that the model is able to detect antecedents in the preceding or following sentence, while it is not able to do so when it is trained and tested on boundary-constraint windowed sentences (SoNaR *windowed*). Nonetheless, this argument does not hold for the other multitask classification models (Model 3 and 4), as the performance differences between the two windowed datasets are not considerably large. Concerning the POS prediction performance in the multitask classification model, it appears that the *relative pronoun* tag is the hardest label to predict. In order to answer the hypotheses stated earlier in this paper, we need to conduct further fine-grained analyses. We first perform an ablation study, in which we not only examine the effects of the POS prediction task on the *die/dat* prediction task ("*We hypothesize that integrating this POS prediction task will positively influence die/dat prediction performance*"), but also benchmark our binary and multitask approaches against a pipeline approach and explore the effects of model architecture alterations on *die/dat* prediction performance (Section 7). We then check whether the model consistently predicts a subordinating conjunction as *dat* ("*We hypothesize that our model will be able to implicitly learn the syntactic functions or POS of die and dat in a given sentence, and*

Multitask Classification Model					
Dataset	**Accuracy**	**Balanced Accuracy**	**Precision sc/rp/dp**	**Recall sc/rp/dp**	**F1 sc/rp/dp**
Model 2: BiLSTM					
SoNaR, *full*	70.72%	70.66%	71.99%/ 63.34%/ 75.30%	73.92%/ 68.29%/ 69.76%	72.88%/ 65.65%/ 72.38%
SoNaR, *windowed*	83.15%	82.68%	84.35%/ 79.42%/ 84.53%	86.98%/ 76.92%/ 84.15%	85.61%/ 78.09%/ 84.31%
SoNaR, *windowed no_boundaries*	85.69%	85.42%	88.78%/ 79.88%/ 87.24%	87.09%/ 82.49%/ 86.68%	87.90%/ 81.11%/ 86.93%
Model 3: Feedforward Context Encoder					
SoNaR, *windowed*	86.46%	86.14%	89.00%/ 80.24%/ 88.71%	87.80%/ 82.88%/ 87.75%	88.37%/ 81.49%/ 88.20%
SoNaR, *windowed no_boundaries*	84.79%	84.76%	88.58%/ 77.04%/ 87.48%	86.23%/ 83.73%/ 84.31%	87.35%/ 80.19%/ 85.84%
Model 4: BiLSTM Context Encoder					
SoNaR, *windowed*	**87.73%**	**87.38%**	**90.12%**/ **82.63%**/ **89.27%**	**88.47%**/ 84.12%/ **89.55%**	**89.26%**/ **83.31%**/ **89.39%**
SoNaR, *windowed no_boundaries*	85.51%	85.48%	87.99%/ 78.90%/ 88.31%	86.98%/ **84.41%**/ 85.04%	87.45%/ 81.51%/ 86.61%

Table 5: Performance results of three multitask classification models for POS prediction: *subordinating conjunction*(sc), *relative pronoun* (rp) and *demonstrative pronoun* (dp).

consistently predict dat when the instance is a subordinating conjunction"), and conduct an error analysis to distinguish linguistic patterns in the prediction errors (Section 8).

7. Ablation Study

In this section, we analyze which model alterations contribute to the performance gains in the multitask classification model, and compare our approaches to a pipeline approach. The multitask classification models differ from the binary classification model in terms of batch size, embedding dimension, number of LSTM layers in the sentence encoder, number of tasks, and input. We start with the influence of the increased batch size and word embedding dimension. We first retrain the multitask classification model and feed the data in batches of 128 (used for binary classifier training) instead of 512 samples. We then retrain the multitask classification model and let the embedding layer transform the input data to 100-dimensional word embeddings instead of 200-dimensional word embeddings. Table 6 shows that there is little consistent difference in performance when batch size is 512 or 128. Therefore, it can be suggested that an increased batch size has no directly positive influence on model performance. Concerning the word embedding dimension, it appears that an increase in word embedding dimension indeed causes a slight increase in model performance.

Ablation study: batch size and embedding dimension					
Model	**Accuracy**	**Balanced Accuracy**	**Precision *dat/die***	**Recall *dat/die***	**F1 *dat/die***
Batch size-128	87.46%	88.73%	89.43%/ 86.94%	91.45%/ 84.02%	90.37%/ 85.33%
Batch size-512	88.36%	88.15%	91.05%/ 84.59%	89.24%/ 87.06%	90.12%/ 85.77%
Embedding-100	86.94%	87.77%	88.54%/ 86.54%	91.29%/ 82.58%	89.88%/ 84.48%
Embedding-200	88.36%	88.15%	91.05%/ 84.59%	89.24%/ 87.06%	90.12%/ 85.77%

Table 6: The influence of batch size and embedding dimension on performance of the SoNaR-based, sentence-exceeding windowed trained multitask classification model (Model 2, SoNaR *windowed no_boundaries*).

The BiLSTM multitask classification model (Model 2) contains two BiLSTM layers opposed to the binary classification model that has only one layer. Table 7 shows the influence of the number of layers on the performance of the binary classification model. When the binary classification model is retrained with an additional BiLSTM layer, all the evaluation metrics rise with approximately 2%. However, when the binary classification model has three BiLSTM layers, model performance drops significantly. It appears that the doubled number of layers is indeed one of the reasons why the multitask classification models perform better than the binary classification model. However, not every rise in number of layers necessarily influences a model's performance in a positive manner: performance drops significantly when we add a third BiLSTM layer.

Concerning the influence of the POS prediction task on *die/dat* prediction performance, a comparison between a two-layer BiLSTM binary classification model (Model 1) and the two-layer BiLSTM multitask classification model (Model 2) is made and displayed in Table 8. It seems that the integration of POS knowledge positively influences *die/dat* prediction performance, as all evaluation metrics have increased. This supports our hypothesis about the positive influence of the POS prediction task on *die/dat* prediction performance.

Ablation study: number of layers					
Layers	**Accuracy**	**Balanced Accuracy**	**Precision *dat/die***	**Recall *dat/die***	**F1 *dat/die***
BiLSTM-1 layer	84.56%	84.18%	87.71%/ 80.13%	86.16%/ 82.20%	86.85%/ 80.99%
BiLSTM-2 layers	87.21%	86.83%	89.62%/ 83.76%	88.82%/ 84.84%	89.15%/ 84.16%
BiLSTM-3 layers	75.75%	76.89%	80.01%/ 72.02%	81.54%/ 69.97%	80.74%/ 70.93%

Table 7: The influence of number of layers on performance of the SoNaR-based, sentence-exceeding windowed trained binary classification model (Model 1, SoNaR *windowed no_boundaries*).

Ablation study: integrated POS knowledge					
POS Knowledge	**Accuracy**	**Balanced Accuracy**	**Precision *dat/die***	**Recall *dat/die***	**F1 *dat/die***
Yes	88.36%	88.15%	91.05%/ 84.59%	89.24%/ 87.06%	90.12%/ 85.77%
No	87.21%	86.83%	89.62%/ 83.76%	88.82%/ 84.84%	89.15%/ 84.16%

Table 8: The influence of integrated POS knowledge on *die/dat* prediction performance. Comparison between Model 1 with an extra BiLSTM layer (*No*) and Model 2 (*Yes*), both trained and tested using the SoNaR *windowed no_boundaries* dataset.

Next, we examine the influence of a context encoder on *die/dat* performance. The similar performance results of the three multitask classification models suggest that the addition of a context encoder has little to no influence on *die/dat* prediction performance. Moreover, the encoder architecture does not cause a considerable difference in *die/dat* prediction performance between the model with a feedforward context encoder (Model 3) and the model with a BiLSTM context encoder (Model 4). It can thus be suggested that the multitask models do not necessarily profit from a different architecture and that an extra focus on immediate context is not additionally advantageous for the *die/dat* prediction task.

Lastly, we benchmark our binary and multitask approach against a pipeline approach in which a dedicated POS tagger first predicts the 'PREDICT' token's POS tag and a *die/dat* classifier then predicts *die/dat* labels in case it is a pronoun. In case of a predicted subordinating conjunction tag, the model automatically yields the *dat* label. As POS tagger, we take a pretrained Dutch language model, BERTje (de Vries et al. 2019), add a linear and softmax layer on top, and fine-tune the model on predicting three POS tags: *subordinating conjunction*, *relative pronoun* and *demonstrative pronoun.* The BERTje tokenizer preprocesses the *windowed no_boundaries* SoNaR sentences in the SoNaR training set, which are then used as input to the POS tagger. During fine-tuning, the linear layer is trained while the other layers are frozen. We apply the following hyperparameters for fine-tuning BERTje on the POS tagging task: number of epochs [30], batch size [512], learning rate [2e-5], optimization [Adam], loss function [cross-entropy]. The POS predictions will be used as a mask over the *die/dat* predictions yielded by the *die/dat* classifier. For the *die/dat* classifier, we use the trained SoNaR, *windowed no_boundaries* model from Section 5.1. Results can be found in Table 9. Overall, the multitask model outperforms the other two approaches on every performance metric.

Ablation study: Pipeline Approach					
Model	**Accuracy**	**Balanced Accuracy**	**Precision *dat/die***	**Recall *dat/die***	**F1 *dat/die***
Binary	84.56%	84.18%	87.71%/ 80.13%	86.16%/ 82.20%	86.85%/ 80.99%
Multitask	88.85%	88.51%	90.95%/ 85.83%	90.29%/ 86.73%	90.60%/ 86.25%
Pipeline	80.46%	82.15%	82.42%/ 81.60%	89.08%/ 71.84%	85.60%/ 76.36%

Table 9: Performance results of a binary classification model (Model 1: SoNaR, *windowed no_boundaries*), a multitask classification model (Model 4: SoNaR, *windowed no_boundaries*) and a pipeline approach (POS tagger + Model 1: SoNaR, *windowed no_boundaries*) for the *die/dat* prediction task.

8. Error Analysis

In this section, we analyze incorrectly predicted *die/dat*-POS combinations in a qualitative manner. From all correctly predicted POS tags, we take a random subset of 100 test sentences for which the model predicts *die* instead of *dat*, and 100 test sentences for which the model predicts *dat* instead of *die*. We then look for error patterns in the random subsets. The qualitative analysis is conducted on the errors yielded by the multitask classification model with BiLSTM context encoder (Model 4), trained and tested on the SoNaR *windowed no_boundaries* dataset.

8.1 *Die/dat* as relative pronouns

The model yields a correct *relative pronoun* tag but an incorrect *die/dat* label. We analyze 200 sentences with a relative pronoun, of which 100 sentences are labeled with *dat* instead of *die*, and vice versa. Firstly, antecedents containing more than one noun appear to be a source of confusion for the model (42 sentences). In Dutch, the antecedent is always mentioned before the relative pronoun, but the distance between the two can vary: the relative pronoun immediately follows the antecedent in sentences 1 and 2, while the distance between the antecedent and the pronoun is larger in sentences 3 and 4. As a result, the model cannot simply rely on position and has to choose the most plausible antecedent of the two preceding nouns given the context. However, both nouns are plausible in sentence 2 and 3, so the model does not necessarily make a wrong prediction given the input. It is also possible that the model is able to detect the correct antecedent, but merely fails to estimate its grammatical gender.

1. "...bij het schilderij van Franciscus, **die** het kruis omhelst uit hetzelfde..." (predicted: **dat**)
2. "...initiatief van de seniorenraad, **die** de senioren tot meer beweging..." (predicted: **dat**)
3. "...nichtje van Lily Allen, **dat** dan uit het hoofd van..." (predicted: **die**)
4. "...instituut voor de voeding, **dat** het terrein gebruikt voor zijn..." (predicted: **die**)

The phrase "*het/een aantal* + plural noun" occurs eight times in the error subset. This is a rather large number, as it is a very specific combination. Although it is grammatically correct to refer to it with either *die* or *dat*, they cannot be used interchangeably. The number of the finite verb in the relative clause signals whether the writer regards the antecedent as a group (singular verb, sentence 5) or individual members (plural verb, sentence 6). In the first case, the pronoun agrees with the gender of *"aantal"* (neuter), so only *dat* is correct. In the latter case, the pronoun agrees with the gender of the plural noun, so only *die* is correct. These errors indicate that the model does not rely on the number of the finite verb in the relative clause.

5. "...maar het aantal mensen **dat** hulp nodig heeft, is..." (predicted: **die**)
6. "...Nederland een aantal creaties mee **die** gemaakt zijn op basis van..." (predicted: **dat**)

The model does not deal well with proper names. In the relative pronoun error subset where *die* is predicted instead of ground-truth label *dat*, 47 of the 100 sentences have an antecedent that denotes either a sports club that is referred to by a city name (7-8), or an organisation (9-10). These nouns are generally neuter, so only *dat* is correct. However, if the name of an organisation is an abbreviation and has a meaningful keyword (11-12), the antecedent's grammatical gender depends on the keyword's gender. We found that for the latter two sentences the ground-truth label is incorrect: *"sp.a"* means *"socialistische partij anders"*, while *"VVD"* is *"Volkspartij voor Vrijheid en Democratie"*. Both keywords are meaningful, and thus the grammatical gender of the complete antecedent relies on the grammatical gender of the keywords - in this case, masculine. The model indeed provided correct predictions. However, it is highly unlikely that the model is aware of this rule and knows the abbreviations in full. Consequently, the model might have predicted the correct label by chance.

7. "...jeugdwerking showen tegen Oudenaarde, **dat** eveneens een aardig jeugdpotentieel heeft." (predicted: **die**)
8. "Hij verwees naar Moeskroen, **dat** geen licentie kreeg omdat de..." (predicted: **die**)
9. "Academic Sofia, **dat** de Russiche olieraffinaderij Lukoil als..." (predicted: **die**)
10. "Nike, **dat** afgelopen maand nog zijn topman..." (predicted: **die**)

11. “De huidige coalitiepartner sp.a, **dat*** met de 28-jarige Simon Bekaert...” (predicted: **die**, correct)
12. “...van de liberale VVD, **dat*** een strenger asielbeleid bepleitte.” (predicted: **die**, correct)

Lastly, the model has not learned to refer to people with *die* (13-14). In fourteen sentences, the model predicts an incorrect *dat* label. We assume that the names did not occur frequently enough in the training set for the model to learn its grammatical gender, and that the model was unable to generalize over the various names and detect that these words refer to people.

13. “...gewassen tegen Delphine Verburgh, **die** de drie wedstrijden won.” (predicted: **dat**)
14. “Hilaire Van der Schueren, **die** VDB voorlopig uit competitie houdt...” (predicted: **dat**)

8.2 *Die/dat* as demonstrative pronouns

The model outputs a correct *demonstrative pronoun* tag but an incorrect *die/dat* label. We analyze 200 sentences with a demonstrative pronoun, of which 100 sentences are labeled with *dat* instead of *die*, and vice versa. The windowed sentences approach shows its disadvantages when a *die/dat* is an independent demonstrative pronoun referring to a distant noun or a previous sentence (82 sentences). As the window is limited to five tokens before and after the prediction token, only a part of the antecedent is given (15-16). Sometimes, the antecedent is not given at all (17-20). As a result, the model is unable to detect the entire antecedent and its grammatical gender.

15. “...computer op de zolderkamer. **Dat** zorgde voor een hevige rookontwikkeling...” (predicted: **die**)
16. “...nieuwe vriend. Ik las **dat** ook in de krant...” (predicted: **die**)
17. “Het wordt tijd dat ze **dat** ernstig gaan nemen.” (predict: **die**)
18. “...wil Xandee niet prijsgeven. **Die** heeft ze, samen met...” (predicted: **dat**)
19. “...overslaat op mensen, is **die** in bijna de helft van...” (predicted: **dat**)
20. “...in het Groot-Hertogdom Luxemburg. **Die** begon al van op enige...” (predicted: **dat**)

The model appears to struggle with demonstrative pronouns in combination with the adjective “*laatste*”: “*dat laatste*”, “*die laatste*” (six sentences). In this case, the pronoun depends on the grammatical gender of the omitted noun. The model thus needs to infer which noun is omitted and what its grammatical gender and number are. The omitted noun could be “*recreatiedomein*” or “*recreatiepark*” in sentence 22, which are both neuter, singular nouns, while it could be neuter, singular noun “*product*” in sentence 23. The yielded model predictions indicate that the model based it predictions on the noun closest to the prediction token. These errors are not surprising, as inferring an omitted noun requires a broad vocabulary and extensive world knowledge.

21. “...fruit kost. Al is **die** laatste vooral water en amper...” (predicted: **dat**)
22. “...Leen en de Blaarmeersen (**dat** laatste is van de stad...” (predicted: **die**)
23. “...de kindershampoo de konijnenshampoo. **Dat** laatste blijkt een zachte shampoo...” (predicted: **die**)

Lastly, the model performs poor on predicting *dat* in combination with a plural copula and a plural predicate nominal (seven sentences). In this special case, there is no agreement between subject (“*dat*”) and verb (“*zijn*”), but between verb and the plural predicate nominal (“*jongens*”, “*mensen*”). Our model might have learned that prediction tokens followed by plural verbs usually take *die*, so that the subject-verb agreement is respected. It arguably failed to learn this special case.

- "...Vanendert , Vanavermaet en Cornu **dat** zijn toch geen kleine jongens" (predicted: **die**)
- "Waarschijnlijk zijn **dat** mensen met een heel groot..." (predicted: **die**)

8.3 *Dat* as subordinating conjunction

We found that the model always predicts a *subordinating conjunction* label in combination with *dat*. In other words, it has learned that it should never assign the *subordinating conjunction* POS tag to a jointly-predicted *die* label. This confirms our hypothesis that our model will have learned to consistently predict *dat* when it is a subordinating conjunction. However, we found 217 sentences in the entire dataset - of which nine in the test set - that had *die* instances with a *subordinating conjunction* label as gold standard. This indicates that the ground-truth POS tags are not highly accurate. An evaluation of the POS tag quality in the SoNaR corpus would therefore be interesting. As this is beyond the scope of this work, we leave this for future work. We manually correct the ground-truth POS tags for the nine test sentences and compare the multitask classification model's *die/dat* and POS predictions with the corrected POS ground-truth labels. The model predicts for seven sentences the correct *die* label, while it predicts the corrected ground-truth POS tag for five sentences.

9. Conclusion and Future Work

Deciding which pronoun to use in various contexts can be a complicated task. The correct use of *die* and *dat* as Dutch pronouns entails knowing the linguistic class of the antecedent and - if the antecedent is a noun - its grammatical gender and number. We experimented with neural network models to examine whether *die* and *dat* instances in sentences can be computationally predicted and, if necessary, corrected. Our binary classification model reaches a promising 84.56 % accuracy. In addition, we extended the model to a multitask model which simultaneously predicts *die/dat* and its POS (*demonstrative pronoun*, *relative pronoun* and *subordinating conjunction*). By increasing the word embedding dimension, doubling the number of bidirectional LSTM layers and integrating POS knowledge in the model, the multitask classification models raise *die/dat* prediction performance by approximately 4 %. Concerning POS prediction performance, the multitask classification model consisting of a sentence and context encoder performs best on all evaluation metrics and reaches 87.78 % accuracy.

There are ample opportunities to further analyze, enhance and/or extend the *die/dat* prediction model. A qualitative study of the learned model weights, for example, could provide more insight in the prediction mechanism of the models. We already obtain excellent results with a simple neural architecture comprising relatively few parameters. We believe that more complex architectures such as a transformer architecture (Vaswani et al. 2017) with multihead attention will improve results. It might also be interesting to look at the possibility of integrating a language model such as BERT (Devlin et al. 2018) in the classification model (e.g., as pretrained embeddings). Moreover, the binary classification task could be extended to a multiclass classification task to predict not only *die* and *dat* labels, but also respectively equivalent *deze* and *dit* labels. The difference between *die/dat* and *deze/dat*, however, entails a difference in time and space: while *die/dat* indicates a physically distant or earlier mentioned antecedent, *deze/dit* implies that the antecedent is physically near or later mentioned in the text. Moreover, *die/dat* and *deze/dit* are preferably used for anaphoric and cataphoric reference, respectively. The difference in reference (1-2) and spatial understanding (4) between *dat/dit* and *die/deze* is demonstrated below.

1. "Je bent gek. **Dat** heb ik je al gezegd." (VRT Taal 2020)
   "*You are crazy. I have told you* ***that*** *already.*"
2. "Ik heb je **dit** al gezegd: je bent gek."
   "*I have told you* ***this*** *already: you are crazy.*"

3. “Ik heb je al gezegd **dat** je gek bent.”
   “*I have told you already* ***that*** *you are crazy.*”

4. “Lees eerst **deze** <u>boeken</u>, dan **die** <u>andere</u>.” (Taaltelefoon 2020)
   “*First, read* ***these*** *books, than* ***those*** *other.*”

Dat in example 1 indicates an anaphoric reference to the previous sentence. The same message is conveyed in example 2, but the sentence is referred to cataphorically using *dit*. Example 3 is very similar to example 2 in terms of sequence in which the information is provided. However, *dat* and *dit* differ in POS: *dit* is an independent demonstrative pronoun and functions as direct object (2), whereas *dat* is a subordinating conjunction and the entire subordinate clause ”*dat je gek bent*” functions as direct object (3). In addition, the word order differs in both examples. Finally, *deze* (4) indicates that its antecedent is spatially close to the speaker, whereas *die* is spatially distant. In order to learn the difference between *dat/dit* and *die/deze*, the model may need to focus more on the antecedent’s position with respect to the pronoun, POS, word order and other tokens in the sentences such as colons, and it will need to infer the spatial (and temporal) relation between the speaker and the antecedent.

References

Allein, Liesbeth (2019), *Automatic prediction and correction of Dutch pronouns die and dat using recurrent neural networks*, Master’s thesis, KU Leuven. Faculteit Ingenieurswetenschappen, Leuven.

Allein, Liesbeth, Artuur Leeuwenberg, and Marie-Francine Moens (2020), Binary and multitask classification model for Dutch anaphora resolution: Die/dat prediction, *arXiv preprint arXiv:2001.02943*.

Beeksma, Merijn, Maarten van Gompel, Florian Kunneman, Louis Onrust, Bouke Regnerus, Dennis Vinke, Eduardo Brito, Christian Bauckhage, and Rafet Sifa (2018), Detecting and correcting spelling errors in high-quality Dutch wikipedia text, *Computational Linguistics in the Netherlands Journal* **8**, pp. 122–137.

Chiu, Jason P.C. and Eric Nichols (2016), Named entity recognition with bidirectional LSTM-CNNs, *Transactions of the Association for Computational Linguistics* **4**, pp. 357–370.

Cornips, Leonie and Aafke Hulk (2006), External and internal factors in bilingual and bidialectal language development: Grammatical gender of the Dutch definite determiner, *L2 acquisition and creole genesis. Dialogues*, John Benjamins Publishing, pp. 355–378.

de Vries, Wietse, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim (2019), BERTje: A Dutch BERT model, *arXiv preprint arXiv:1912.09582*.

Delobelle, Pieter, Thomas Winters, and Bettina Berendt (2020), RobBERT: a Dutch RoBERTa-based language model, *arXiv preprint arXiv:2001.06286*.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018), BERT: Pre-training of deep bidirectional transformers for language understanding, *arXiv preprint arXiv:1810.04805*.

Heyman, Geert, Ivan Vuli, Yannick Laevaert, and Marie-Francine Moens (2018), Automatic detection and correction of context-dependent dt-mistakes using neural networks, *Computational Linguistics in the Netherlands Journal* **8**, pp. 49–65.

Kloppenburg, Lennart and Malvina Nissim (2016), Native-data models for detecting and correcting errors in learners’ Dutch, *Computational Linguistics in the Netherlands Journal* **6**, pp. 39–55.

Koehn, Philipp (2005), Europarl: A parallel corpus for statistical machine translation, *MT summit*, Vol. 5, Citeseer, pp. 79–86.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013), Distributed representations of words and phrases and their compositionality, *Advances in Neural Information Processing Systems*, pp. 3111–3119.

Oostdijk, Nelleke, Martin Reynaert, Véronique Hoste, and Ineke Schuurman (2013), The construction of a 500-million-word reference corpus of contemporary written Dutch, *in* Spyns, Peter and Jan Odijk, editors, *Essential Speech and Language Technology for Dutch: Results by the STEVIN programme*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 219–247.

Schmid, Helmut (1994), Probabilistic part-of-speech tagging using decision trees, intl, *Conference on New Methods in Language Processing. Manchester, UK.*

Sevens, Leen, Tom Vanallemeersch, Ineke Schuurman, Vincent Vandeghinste, and Frank Van Eynde (2016), Automated spelling correction for Dutch internet users with intellectual disabilities, Schuurman, Ineke, Paris; ELRA, pp. 11–19.

Stehouwer, Herman and Antal van den Bosch (2008), Putting the t where it belongs: Solving a confusion problem in Dutch, *the 18th CLIN Meeting, 2007*, LOT, pp. 21–36.

Taaltelefoon (2020), deze/die. https://www.taaltelefoon.be/deze-die [Accessed: 11 May 2020].

Tijhuis, Lars (2014), *Context-based spelling correction for the Dutch language: Applied on spelling errors extracted from the Dutch wikipedia revision history*, Master's thesis, TU Delft. Electrical Engineering, Mathematics and Computer Science, Delft.

van den Bosch, Antal and Peter Berck (2012), Memory-based text correction for preposition and determiner errors, *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pp. 289–294.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017), Attention is all you need, *Advances in neural information processing systems*, pp. 5998–6008.

VRT Taal (2020), deze/die/dit/dat. https://vrttaal.net/taaladvies-taalkwestie/deze-die-dit-dat [Accessed: 11 May 2020].