Computational Linguistics in the Netherlands Journal 11 (2021) 281-306 Submitted 10/2021; Published 12/2021

Automatic Detection and Annotation of Spelling Errors and Orthographic Properties in the Dutch BasiScript Corpus

Wieke Noa Harmsen [*] WIEKE.HARMSEN2@RU.NL
Catia Cucchiarini [*] CATIA.CUCCHIARINI@RU.NL
Helmer Strik [*] HELMER.STRIK@RU.NL

[*] *Centre for Language and Speech Technology, Nijmegen, The Netherlands*

Abstract

Learning to spell in Dutch is a difficult task that children cannot learn autonomously. Essential for developing good spelling skills is direct instruction of the spelling principles, enough practice, and feedback. With respect to direct instruction, a qualitative overview of which spelling errors are made by which type of children at which point in time can be very useful to design effective spelling lessons. In addition, such an overview would enable large-scale, quantitative, research on children's spelling development. In an earlier study we presented an algorithm that can automatically detect and annotate spelling errors in BasiScript, a corpus containing texts written by primary school children. In the present study, we extended the functionality of this algorithm to make it capable of annotating written words with their orthographic properties. In practice, this means that correctly spelled letters are detected and annotated with the spelling principle that is applied correctly. These additional annotations allow us to compute relative scores, which show how often a spelling principle is applied incorrectly with respect to the total occurrence frequency of that spelling principle.

Using this relative frequency measure, we found that spelling principles from the syntax and semantics category are more problematic to learn for Dutch primary school children between second and sixth grade than phoneme-to-grapheme conversion, context and morphology spelling principles. Primary school children are especially bad at applying spelling principles concerning capital letter use, the writing of present participles, and past participles ending in "d", while you hear a /t/ sound.

In this paper, we first describe the implementation of the algorithm, its evaluation, and the computation of the relative frequency in more detail. We then discuss the results and possible limitations of our study and address future avenues of research.

1. Introduction

1.1 Spelling in Dutch

The ability to express oneself in writing is an important skill that everybody should acquire to fully participate in society. Essential for good writing skills is a good command of spelling. In an alphabetical language like Dutch, this involves choosing the right letters (graphemes) to represent the speech sounds (phonemes) a word consists of. However, knowledge of only phoneme-to-grapheme conversion spelling principles (Nunn 1998), also known as "unmarked" spelling principles (Horbach-Kleijnen 1992), is not sufficient to write flawless Dutch. Since Dutch is only a relatively transparent language (Borgwaldt et al. 2004), a frequently occurring problem is that some phonemes can be written in multiple ways. In these cases, a set of autonomous (Nunn 1998) or "marked" spelling principles (Horbach-Kleijnen 1992) is necessary to obtain the correct spelling of a word. These marked spelling principles are based on the context (the context of a letter or sequence of letters determines how it is written), morphology (morphemes with the same meaning are written in the same way), syntax (the function of a word determines the spelling of the suffix), and semantics (the meaning of a word influences the spelling) of a word. Table 1 represents for each of these four categories an example where the application of marked spelling principles is necessary.

Table 1: Application of marked spelling principles is necessary for the spelling of some words. There are four categories of marked spelling principles. This table shows for each category an example. The computer phonetic alphabet used to write the phonemes is the same as used in the Spoken Dutch Corpus 2.0 (Taalunie 2014).

Marked Category	**Example**			**Explanation**
	Phonemes	Graphemes	English translation	
Context	m a k @ n	(wij) maken	(we) make	While the /a/ is long, it is written as “a” instead of “aa”, because the vowel occurs at the end of a syllable. The context of a vowel determines the spelling.
Morphology	k A s j @	kastje	little cupboard	While the “t” is not pronounced, it is written, since the word “kastje” contains the morpheme “kast” (cupboard).
Syntax	v E r b r A n t	(ik) verbrand, (hij) verbrandt, (ik ben) verbrand	(I) burn, (he) burns, (I am) burned	“verbrand” and “verbrandt” are both pronounced in exactly the same way. The function of the verb (syntax) in the sentence determines the spelling.
Semantics	n EI m e x @ n	Nijmegen	Nijmegen	“Nijmegen” and “nijmegen” are both pronounced exactly the same. The semantics of the word (Nijmegen is a city) determines that it is written with a capital letter.

1.2 Dutch spelling errors

In this study, we mark a word as spelled incorrectly, when it is written with incorrect orthography. In addition, we define a spelling error as a Phoneme-Corresponding Unit (PCU) that is inserted, deleted or substituted when aligning the current, incorrect spelling of a word with its target spelling. The term PCU was introduced by Laarmann-Quante (2016) in her research on automatic detection and annotation of German spelling errors and refers to a sequence of graphemes that corresponds to one phoneme. In Dutch, this can be a sequence of only one grapheme (e.g., “k” or “p”), but also of two (e.g., “oe” or “ch”) or even three graphemes (e.g., “sch”). An example of how the word “automatisch” (automatic) is split into PCUs is visible in Table 2.

Table 2: Segmentation of the word “automatisch” (automatic) into PCUs. The phonemes are written using the computer phonetic alphabet from Corpus Spoken Dutch 2.0 (Taalunie 2014).

Phonemes	AU	t	o	m	a	t	i	s
PCUs	au	t	o	m	a	t	i	sch

There are three types of PCUs: PCUs that are written using only consonants (e.g., “k”, “ph”, “sch”), PCUs that are written using only vowels (e.g., “e”, “oe”, “eau”) and PCUs that are written using a combination of vowels and consonants (e.g., “qu”, “ij”). In the PCU alignment, it is possible to align vowel PCUs with vowel PCUs and consonant PCUs with consonant PCUs, even if they do not correspond to the same phoneme. However, vowel PCUs cannot be aligned with consonant PCUs and vice versa. In addition, for the combination PCUs that consist of both a vowel and consonant, it depends on the phoneme they correspond to whether they are treated as a vowel PCU or consonant PCU. For example, "ij" corresponds to the phoneme /ei/, so this combination PCU is treated as a vowel PCU. In Table 3 some alignment examples are shown.

Table 3: Four examples in which the PCUs of the target spelling (correct spelling) are aligned with the PCUs of the original spelling (incorrect spelling). Each spelling error is marked as insertion error ("i"), deletion error (" d") or substitution error ("s").

Table 3a: The word "liep" written incorrectly as "loopte" (walked).

PCUs target	l	ie	p	-	-
PCUs original	l	oo	p	t	e
Error type		s		i	i

Table 3b: The word "keek" written incorrectly as "kijkte" (watched).

PCUs target	k	ee	k	-	-
PCUs original	k	ij	k	t	e
Error type		s		i	i

Table 3c: The word "lachte" written incorrectly as "lachde" (laughed).

PCUs target	l	a	ch	t	e
PCUs original	l	a	ch	d	e
Error type				s	

Table 3d: The word "chique" written incorrectly as "chic" (chic).

PCUs target	ch	i	qu	e
PCUs original	ch	i	c	-
Error type			s	d

1.3 Learning to spell in Dutch

Writing flawless Dutch requires knowledge of both marked and unmarked spelling principles (Horbach-Kleijnen 1992, Nunn 1998, Schijf 2009). Since children are not very good in discovering underlying spelling principles by themselves (Assink 1986), spelling instruction is required in which spelling principles are explicitly taught. In these lessons, children also practice writing, which appears to be essential for developing good spelling skills. As a matter of fact, Leerdam et al. (1998) found that by only reading words and not writing them, it is not possible to develop good spelling skills. In addition, Cordewener et al. (2016) have shown that it is important to write complete words, because writing only parts of words or single letters does not support the development of good spelling skills. As to feedback on spelling, children need to know what they did right and wrong, so that they know where they can improve their spelling. Feedback can be given in many forms and Harward et al. (1994) showed that immediate feedback on each written word is more effective for developing good spelling skills than delayed feedback given after a complete list of words has been written.

1.4 The present study

Thus, learning to spell is a difficult task that children cannot learn autonomously. They need help from a teacher who teaches spelling principles, stimulates them to practice, and corrects their work. Quantitative knowledge about which spelling errors are made by which type of children at which point in time can be very useful to design effective spelling lessons and enable quantitative research on children's spelling ability. So far, Dutch children's spelling ability has only been research qualitatively using dictations (Horbach-Kleijnen 1997, Keuning and Verhoeven 2008, Schijf 2009).

In a previous study, we reported on the development of an algorithm that makes automatic quantitative analysis of spelling errors possible (Harmsen et al. 2021). This algorithm has the functionality to automatically detect incorrectly written PCUs and to annotate them indicating which spelling principle was violated. We applied this algorithm to the BasiScript corpus (Tellings et al. 2018), a large corpus of texts written by primary school children, and found that most spelling errors resulted from incorrect application of unmarked spelling principles, which are also called phoneme-to-grapheme conversion principles. However, an important caveat to this result is that unmarked spelling principles are also much more frequently applied than spelling principles from other categories. For example, writing the word "hond" (dog) involves three applications of a phoneme-to-grapheme conversion rule (for writing "h", "o" and "n") and one application of the morphology rule "Final Devoicing" (for writing "d").

The first aim of the current study is to overcome this limitation by expanding the spelling error detection and annotation algorithm presented earlier (Harmsen et al. 2021), so that it is also capable to annotate correctly written PCUs with their orthographic property. This property indicates which spelling principle was applied correctly. The second aim of the study is to use both orthographic property and spelling error annotations to compute relative scores that express how often a spelling principle is applied incorrectly with respect to how often it is used in total (both spelled correctly and incorrectly). This yields quantitative results that describe how well children have mastered specific spelling principles, and thus how problematic to learn they are. The following research questions describe these two aims:

1. Can we extend the rule-based algorithm developed by Harmsen et al. (2021) to also be able to annotate each PCU from a correctly written Dutch word indicating which spelling principle should be applied to spell this PCU correctly?

2. Can we use the developed algorithm to discover which spelling principles are relatively most problematic to learn for Dutch primary school children from grade two (7-8 years old) to grade six (11-12 years old)?

1.5 Annotating correctly written PCUs with spelling principles

For German, two algorithms have been developed that can detect and annotate both correctly and incorrectly written PCUs indicating the spelling principle that is violated (Berkling and Lavalley 2015, Laarmann-Quante 2016). Because German is just as Dutch an alphabetical and relatively transparent language (Borgwaldt et al. 2004), our hypothesis with respect to the first research question is that it should be possible to extend the current spelling error detection and annotation algorithm, so that the detection and annotation of correctly spelled PCUs is also possible.

Both German algorithms use a different method for the detection and annotation of correctly and incorrectly written PCUs. Berkling and Lavalley (2015) first automatically derive the phonetic transcription of both the original spelling (what the child wrote, including spelling errors) and target spelling (a corrected version of the original text). After that, they align these two phonetic strings using MARY (Schrödel and Trouvain 2003), an algorithm based on the articulatory features of phonemes. Using this phonetic alignment, the graphemes of the original and target spelling can also be aligned with each other. In this alignment, errors are detected as substitutions, insertions, or deletions in the alignment. Subsequently, both the incorrectly and correctly spelled letters are annotated so that both the spelling principles that are violated and those that are applied correctly are shown using a set of rules. The spelling error annotations are saved in the "error" layer and the annotations of correctly written letters in the "basis" layer.

In the second German spelling error study by Laarmann-Quante (2016), alignment of original and target spelling is carried out differently. She first generates a lexicon containing all possible misspellings of a target word that occur when one or more spelling principles are applied incorrectly. For each possible misspelling, both the alignment with the target transcription and the violated spelling principle that causes the error are saved in the lexicon. The lexicon also contains all correct spellings of words, and the spelling principles that need to be applied to obtain this spelling. After creation of this lexicon, each (misspelled) word achieved from an input text can be detected and annotated by searching the words in the lexicon. When a match is found, alignment with the target spelling, correctly applied spelling principles ("orthographic properties"), and violated spelling principle(s) ("spelling errors") can be derived from this lexicon. In comparison to the algorithm by Berkling and Lavalley (2015), this method yields better alignment of target and original words, because a complete search is done. However, this approach is computationally inefficient and time-consuming. Since the algorithm by Berkling and Lavalley (2015) also yields good detection and annotation results, we take this approach as the starting point in the current study.

1.6 The most problematic Dutch spelling principles

To investigate which spelling principles are relatively most problematic to learn for Dutch primary school children (the second research question), we will compute for each spelling principle a relative score. This score represents how often a spelling principle is applied incorrectly in relation to how often this spelling principle is or should be applied.

We hypothesize that spelling principles from the syntax category are relatively most problematic to learn, since verb spelling is part of this category and primary school children's verb spelling performance is found to be relatively low, only just above chance level (Bosman 2005). In addition, Schijf (2009) found that first-year secondary school students (12-13 years old) make more errors in applying verb spelling principles than in applying spelling principles from other categories. Bosman (2005) stated that even experienced spellers (i.e., university students) still make many verb spelling errors.

One reason why verb spelling is difficult is that many Dutch verb forms are homophones (Assink 1985, Sandra 2010, Schmitz et al. 2018). For example: the verbs "beloofd" (promised) and "belooft" (promises) are homophones, and their spelling differs depending on the function of the verb in the sentence. To obtain the right spelling, the retrieval procedure (Sandra et al. 1999, Verhaert et al. 2016), in which spellers obtain the right spelling from their mental lexicon, is not sufficient and use of the computational procedure (Sandra and Abbenyen 2009), in which spellers actively apply spelling principles, is necessary.

Assink (1985) found that writers are more likely to write a verb incorrectly if its homophone counterpart is more frequent. In addition, the syntactical context of a verb influences the spelling. This means that if the subject of the verb was written next to the verb (facilitating context), the verb is more likely to be written correctly than when the subject and verb are separated by a few other words (neutral context) (Assink 1985).

2. Methodology

In this section, we describe the algorithm that is capable of annotating both correctly and incorrectly written Phoneme-Corresponding Units (PCUs) indicating which spelling principles are violated or applied correctly. The developed algorithm consists of three parts: preprocessing of the texts, detection of PCUs and spelling errors, and annotation of PCUs displaying (in)correctly applied spelling principles. These three steps result in a multilayered analysis of each BasiScript word. Figure 1 presents a schematic overview of these three steps together with their inputs and outputs. In Table 4, on the next page, is an example visible of the word "scholen" (schools) and all its annotation layers. The goal of the algorithm is to obtain these annotation layers for every analyzed word. The complete algorithm was implemented and evaluated using Python 3 (Van Rossum and Drake 2009). The code is made publicly available on GitHub (Harmsen 2022).

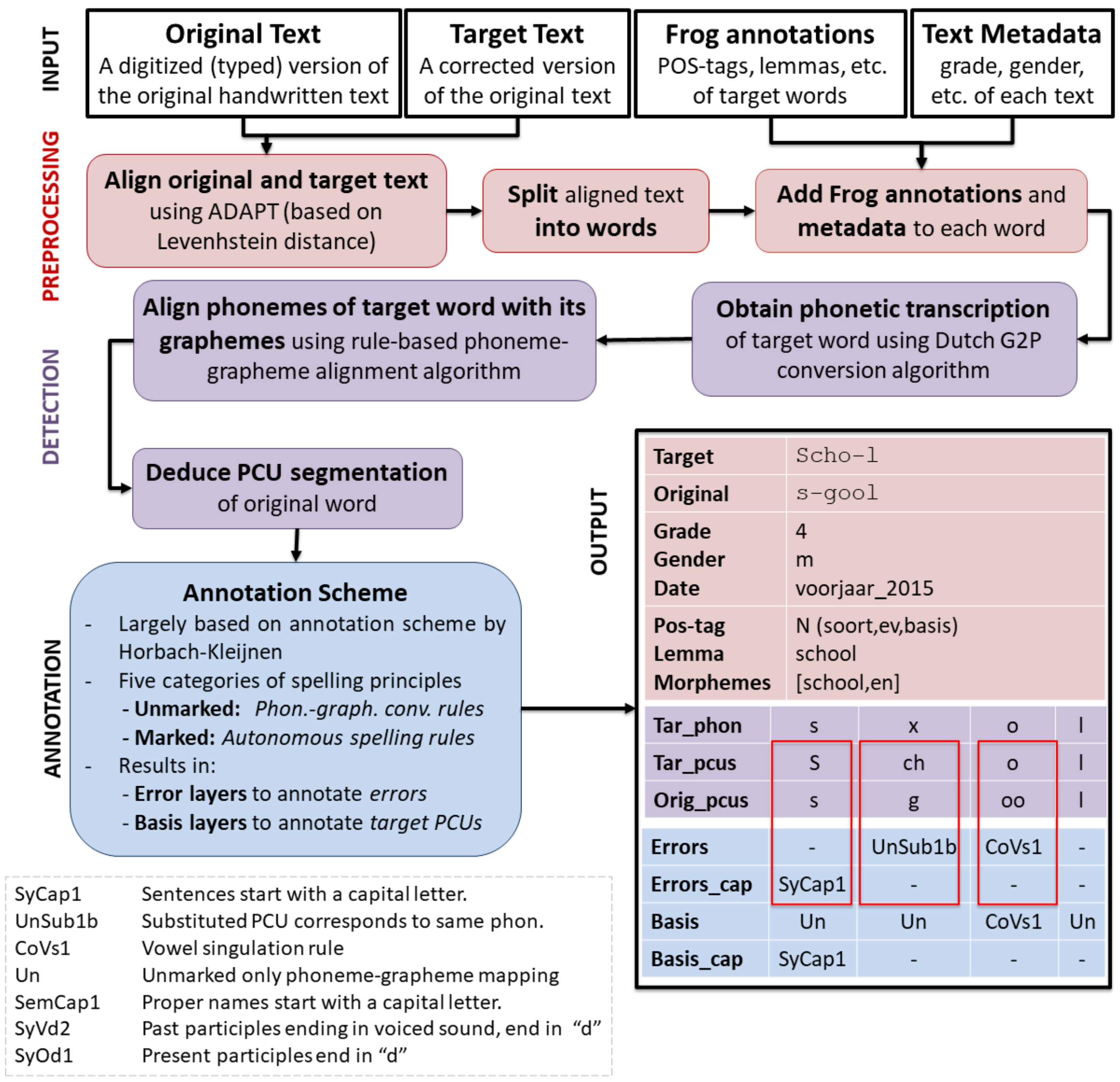
Table 4: The goal of the algorithm is to obtain for each analyzed input word a multilayered annotation scheme. This table shows an example of the annotations of the misspelled word "sgoole".

LAYER	VALUE					
word	Scholen					
target	`Scho-len`					
original	`s-goole-`					
author	JurC					
gender	j (jongen, boy)					
grade	6					
date	najaar_2014 (autumn 2014)					
theme	ThemaRarewoorden (Theme weird words)					
fileName	d389055.xml					
morphemes	[school, en]					
lemma	school					
pos-tag	N(soort,mv,basis)					
punctEndSentence	False					
capitalBeginSentence	True					
homophones	True					
phon_target	s	x	o	l	@	-
pcus_target	S	ch	o	l	e	n
pcus_original	s	g	oo	l	e	-
error	-	UnSub1	CoVs1	-	-	MoEndN1
error_capital	SyCap1	-	-	-	-	-
basic	Un	Un	CoVs1	Un	Un	MoEndN1
basic_capital	SyCap1	-	-	-	-	-

2.1 The input: texts from the BasiScript corpus

The written data used in this study are texts from the BasiScript corpus (Tellings et al. 2018). These texts are handwritten by Dutch primary school children from grade 2 (7-8 years old) to grade 6 (11-12 years old). The data was collected in six consecutive rounds. Spread over three school years, there was a data collection round every autumn and spring. In each round, the participating children wrote maximally four texts with pen on paper: two texts about a given theme and two free-themed texts. However, not all four items were written by all children and not all schools participated in all data collection rounds they were invited to. The average length of the texts, measured in words, increases over the grades, from 44.85 words on average in grade 2 to 109.97 words on average in grade 6.

For each handwritten text, the BasiScript corpus contains two digitized (i.e., typed) versions: the target text (the intended text, without spelling errors) and the original text (what the child


INPUT
Original Text
A digitized (typed) version of the original handwritten text
Target Text
A corrected version of the original text
Frog annotations
POS-tags, lemmas, etc. of target words
Text Metadata
grade, gender, etc. of each text
PREPROCESSING
Align original and target text using ADAPT (based on Levenhstein distance)
Split aligned text into words
Add Frog annotations and metadata to each word
DETECTION
Align phonemes of target word with its graphemes using rule-based phoneme-grapheme alignment algorithm
Obtain phonetic transcription of target word using Dutch G2P conversion algorithm
Deduce PCU segmentation of original word
ANNOTATION
Annotation Scheme
- Largely based on annotation scheme by Horbach-Kleijnen
- Five categories of spelling principles
- Unmarked: Phon.-graph. conv. rules
- Marked: Autonomous spelling rules
- Results in:
- Error layers to annotate errors
- Basis layers to annotate target PCUs
OUTPUT

Target	Scho-l			
Original	s-gool			
Grade	4			
Gender	m			
Date	voorjaar_2015			
Pos-tag	N (soort,ev,basis)			
Lemma	school			
Morphemes	[school,en]			
Tar_phon	s	x	o	l
Tar_pcus	S	ch	o	l
Orig_pcus	s	g	oo	l
Errors	-	UnSub1b	CoVs1	-
Errors_cap	SyCap1	-	-	-
Basis	Un	Un	CoVs1	Un
Basis_cap	SyCap1	-	-	-

SyCap1	Sentences start with a capital letter.
UnSub1b	Substituted PCU corresponds to same phon.
CoVs1	Vowel singulation rule
Un	Unmarked only phoneme-grapheme mapping
SemCap1	Proper names start with a capital letter.
SyVd2	Past participles ending in voiced sound, end in "d"
SyOd1	Present participles end in "d"

Figure 1: A schematic overview of the spelling error detection and annotation algorithm. The red blocks and layers represent the preprocessing part, the purple blocks and layers the detection part and the blue blocks and layers the annotation part.

actually wrote, including crossed-out words and spelling errors). Each target text contains lemma, morphemes and part-of-speech tag annotations, that were automatically obtained using Frog 0.13, which is an advanced natural language processing suite for Dutch (Bosch et al. 2007). In addition, metadata is saved for each text. This metadata contains additional information about the text, like the name and grade of the author, the theme of the text and the date on which the text was written.

For each BasiScript text, the target text, original text, Frog annotations and metadata are saved in a FoLiA file (Gompel and Reynaert 2013). This is a practical format for XML-based linguistic annotation.

2.2 Preprocessing

Several variables from the BasiScript FoLiA XML files constitute the input to the algorithm. These variables are the original text (including spelling errors), target text (without spelling errors), metadata (e.g., grade, date, name, etc.) and additional Frog annotations (i.e., lemma, morphemes and pos-tags of each word in the target text). These variables are first preprocessed, so that the data is presented in the right format to detect PCUs and spelling errors.

2.2.1 Clean and align original and target texts

Each original text (i.e., the text with spelling errors) contains annotations to represent crossed-out tokens that are still readable, letters written in mirror image, personal information, and unreadable tokens. In the first preprocessing step, the algorithm removes all these tags, because they are not written in the target text and would therefore complicate alignment of the original and target text.

After that, the algorithm aligns each target text with the corresponding original text using the ADAPT algorithm (Elffers et al. 2013). This is a dynamic algorithm based on articulatory distances between Dutch speech sounds that was originally developed for the alignment of phoneme strings (Cucchiarini 1993, Cucchiarini 1996). However, the current algorithm employs an altered version of ADAPT in which the feature matrices of the algorithm have been modified, so that alignment of grapheme strings is possible. These modifications included extension of the two feature matrices that were originally present in the ADAPT-algorithm: accented vowels (e.g., ë, é, and è) were added to the vowel feature matrix and accented consonants (e.g. ñ and ç) to the consonant matrix. In addition, the new version of the ADAPT algorithm now contains a third and fourth feature matrix, one for numbers and one for punctuation marks. In this way, all feature matrices together contain all possible characters that occur in the BasiScript texts.

The output of ADAPT includes an aligned target and original text. In the final alignment, dashes ("-") have been added to the target text to represent inserted symbols and to the original text to represent deleted symbols.

Before the ADAPT algorithm was applied to the text data, the data had to be converted into the right format. In the first place, hyphens that occur naturally in the texts, for example in words like "tv-toestel" (TV set) or "vrijdag- en zaterdagmiddag" (Friday and Saturday afternoon), were replaced by the hashtag symbol ("#"). This was done to avoid confusion with the dashes in the output of the ADAPT alignment. The hashtag symbol was hardly used by the children in their written texts, therefore this symbol was chosen. Secondly, all spaces were replaced by pipe symbols ("|"), because the ADAPT-algorithm interprets pipes as spaces. Thirdly, each text was split into shorter strings of approximately 150 characters. These shorter strings served subsequently as input to the ADAPT algorithm. This last step was necessary, because ADAPT is a dynamic algorithm. This means that it is not capable of handling large strings of symbols, like whole texts, because this costs too much memory. To make sure that the original and target texts were both split at a corresponding location, the algorithm checked whether the five characters before and after the split were the same.

2.2.2 Split aligned text into words

After these short strings of approximately 150 characters have been aligned with each other, the algorithm splits them into tokens (i.e., words, digits, and punctuation marks). The spaces in the target string are used to determine where a new token starts.

2.2.3 Add metadata and Frog-annotations to each word

Subsequently, for each token, the corresponding metadata was saved in a set of layers. The metadata consists of the pseudonymized name of the author, the gender of the author, the grade that the author is in, the date the text was written, the theme of the text, and the original file name.

The BasiScript corpus also contains lemma, morpheme and part-of-speech tag annotations of each token that the target transcription consists of. These annotations have been automatically obtained using Frog 0.13 (Bosch et al. 2007), an advanced natural language processing suite for Dutch. The algorithm matched each token with the corresponding Frog annotations.

Unfortunately, it was in some cases not possible to match the Frog annotations correctly. This is because some texts were split incorrectly into tokens in the previous step, which created non-existing tokens (e.g., the token "!Hallo" does not exist). To avoid problems with missing annotations, texts that contain non-existing tokens were removed from the analyzed data.

Additionally, because the tokens were still in the right order, it was possible to obtain information about whether a capital letter is at the beginning of a sentence, whether a punctuation mark is at the end of a sentence and whether two words are homophones of each other. For each token, we added three layers that represent the value of these variables.

2.2.4 Remove digits and punctuation marks from data

In the last preprocessing step of the algorithm, the complete set containing all annotated tokens was split into three subsets: one containing only words, one with punctuation marks, and one containing digits. Because the current study was not focused on punctuation errors, only the set containing words was selected for further analysis. In addition, the set with digits was not included, because the use of ADAPT for aligning a digit representation of a number (e.g., 16) with a text representation of a number (e.g., sixteen) yielded incorrect alignment. Finally, words with an empty target or original transcription were removed from the data. These words were a result of incorrect alignment.

2.3 Detection of PCUs and spelling errors

In the second part of the algorithm, spelling errors were detected at Phoneme-Corresponding Unit (PCU) level. A PCU is a sequence of graphemes that corresponds to one phoneme (Laarmann-Quante 2016).

2.3.1 Obtain phonetic transcriptions

The algorithm starts this second part by creating a list of all unique target words from the BasiScript corpus. The words in this list were cleaned and preprocessed by removing the dashes that represent insertions and by converting the words to lowercase. This last step will reduce the size of the list. For each word in this list, the phonetic transcription was automatically obtained using a Dutch grapheme-to-phoneme converter webservice (Bosch n.d.). The phonetic transcriptions were represented in a computer phonetic alphabet that was also used in the Spoken Dutch Corpus 2.0 (Taalunie 2014). The target words together with their phonetic transcriptions form a lexicon.

2.3.2 Phoneme-grapheme alignment

Subsequently, each target word in the lexicon was aligned with its phonetic transcription. To perform this task, a rule-based sub-algorithm was developed. This sub-algorithm contains rules that describe which combinations of graphemes can be used to write one phoneme. This is an example of two of these rules: in Dutch, the phoneme /a/ can be written as "aa" as in "maan" (moon), or "a" as in "maken" (to make). The resulting phoneme-grapheme alignments provide information about which sequence of graphemes corresponds to each phoneme, these are the PCU-segmentations of each target word. The algorithm uses this sub-algorithm to extend the lexicon with the PCU-segmentation of each unique target word.

2.3.3 Deduce PCU segmentation

Next, the algorithm uses the created lexicon to derive the PCU-segmentation of the aligned original and target word. Therefore, the PCU-segmentation of the target word is first obtained by selecting the PCU-segmentation of the target word from the lexicon. Next, dashes are inserted to the PCU-segmentation to represent inserted PCUs. The position of these dashes is derived from the target alignment. After that, the PCU-segmentation of the original word is deduced from the PCU-segmentation of the target word and the alignment of the original with the target word. This is done by copying the PCU-segmentation boundaries from the target PCU-segmentation to the original word. In this final alignment of target and original PCUs, spelling errors are detected as inserted, deleted, or substituted PCUs.

2.4 Annotation of PCUs with spelling principles

In the third part of the algorithm, the PCUs are annotated with spelling principles. The goal of this part is to annotate each incorrectly spelled PCU showing which spelling principle was violated (error and error_capital layers) and to annotate each correctly spelled target PCU with the spelling principles that should have been applied to write that PCU correctly (basic and basic_capital layers). In the resulting multilayered annotation scheme, the error layers represent information about which spelling principles are violated in the spelling errors, and the basic layers make it possible to put this information into context.

2.4.1 Annotation scheme

To annotate the PCUs with spelling principles, an annotation scheme is necessary. The annotation scheme adopted in the present study is largely based on the one by Horbach-Kleijnen (1992), because this is a linguistic scheme that describes Dutch orthography. The complete annotation scheme used in this study can be found in the Appendix. The annotation scheme consists of 38 different spelling principles to annotate errors in the error layers ("error spelling principles") and 33 spelling principles to annotate orthographic properties in the basic layers ("basic spelling principles"). The majority of basic spelling principles (32/33) is also used as error spelling principle. The spelling principles are divided over five categories: Unmarked, Marked by Context, Marked by Morphology, Marked by Syntax and Marked by Semantics. Table 5 shows the number of error and basic spelling principles in each category, together with an example.

The spelling principles in the annotation scheme are named using a naming strategy. The first two/three letters always represent the category, which is Context (Co), Morphology (Mo), Syntax (Sy), Semantics (Sem) or Unmarked (Un). After that, an abbreviation of the theme is written down, for example Apostrophe (Ap) or Consonant Doubling (Cd). This subcategory is followed by a number (i.e., 1, 2, 3, etc.), which indicates the different spelling principles within a subcategory.

In some cases, it is possible to split a spelling principle into several sub-principles. These sub-principles are marked with the suffix a, b, c, etc. For example, spelling principle MoFd1 is the first final devoicing (Fd) spelling principle from the morphology (Mo) category (see Appendix). This principle can be split into MoFd1a (voiced obstruent "d" is written as unvoiced "t") and MoFd1b (voiced obstruent "b" is written as unvoiced "p"). The sub-principles always depend on the value of the target PCU and original PCU, therefore they cannot be used as basis spelling principles. The focus of this research is on the spelling principles, and not on the sub-principles. However, the sub-principles are often used to detect whether a PCU should be annotated with a spelling principle.

There are six error spelling principles that have a different basic spelling principle name. In case of UnDel1, UnSub1, UnSub2, and UnSub3 it is only possible to annotate PCUs in the basic layer with the main category ("Un"), since the value of the original PCU is necessary to determine whether the error is a deletion or specific type of substitution. For CoAc2, this also holds, because this rule checks the insertion of an accent on a vowel (e.g. writing "opticien" (optician) as "opticiën"). Since this can literally happen on any vowel, we annotate PCUs where this happens in the error layer as "Un" in the basic layer. So, "Un" actually means application of a basic phoneme-grapheme conversion rule.

The error spelling principle "UnIns1" is used when the incorrectly written original PCU is aligned with an empty target PCU, so the writer inserted a PCU. It is not possible to use this annotation without the original spelling. Therefore, the corresponding basic error spelling principle in "Ins". This annotation only serves as a placeholder in this case, and is only used in combination with "UnIns1" in the error annotation layer.

Table 5: This table represents an overview of the annotation scheme used to annotate both incorrectly spelled PCUs (in the error layer) and correctly spelled PCUs (in the basic layer). The spelling principles in the annotation scheme can be subdivided over five categories. For each category is represented how many spelling principles it contains, and one spelling principle used to annotate errors is shown as example.

Category	**Number of Spelling Principles**		**Example Spelling Principle (used for error annotation)**		
	Error	Basic	Name	Description	Example (original/ target)
Marked by Context	9	8	CoCd1	Consonant Doubling: A consonant is doubled if it is written after a short vowel (excluding sjwa) and if it is not at the end of the word.	jo**k**en / jo**kk**en
Marked by Syntax	13	13	SyVt1	When the stem of a verb ends in an unvoiced sound (the sounds in "'t exkofschip"), the past tense suffix starts with an unvoiced /t/ sound ("t").	hij kras**d**e / hij kras**t**e
Marked by Morphology	10	10	MoFd1	Final devoicing: voiced consonants are pronounced as unvoiced at the end of a word.	hon**t** / hon**d**
Marked by Semantics	1	1	SemCap1	Every proper name, title and some abbreviations start with a capital letter.	**n**ijmegen / **N**ijmegen
Unmarked	5	1	UnSub2	A PCU is substituted with another PCU that does not correspond to the same phoneme.	**k**iefde / **l**iefde
Total	38	33			

2.4.2 PCU annotation process

For each spelling principle, the algorithm contains a Boolean rule that gets as input the target PCU, original PCU, and some extra information from the annotation layers (e.g., lemma, morpheme(s), and the position of the PCU in the word). Using these inputs, each rule is able to determine whether the spelling principle it represents was violated (in case of error annotations) or should have been applied (in case of basic annotations). The annotation scheme contains two spelling principles that involve capital letters. These are SemCap1 (every proper name starts with a capital letter) and SyCap1 (every sentence starts with a capital letter). When one of these two annotations are used, they are written down in separate layers, called error_capital and basic_capital. This was done because a PCU can contain both a capital letter error and a lowercase letter error (e.g., compare "America" and "aamerica").

3. Results

3.1 Evaluation of the algorithm

After we applied the algorithm to texts from the BasiScript corpus, we evaluated the developed algorithm on two aspects. In the first place, we verified that the original and target texts had been properly aligned and split into words and PCUs. Therefore, 1000 incorrectly spelled words were generated (0,016% of all correctly and incorrectly spelled words from the BasiScript corpus) and checked on these aspects. The results show that from this sample of 1000 words from the BasiScript corpus, 97.9% of the words were aligned and split correctly into words and PCUs. An overview of the most frequent alignment errors was presented in an earlier study in which we reported about this algorithm (Harmsen et al. 2021).

The second aspect on which the algorithm is evaluated is the correctness of spelling principle annotations with which the PCUs have been labeled. All target PCUs should have been labeled with a basic annotation that indicates the spelling principle that was applied correctly, and all incorrect original PCUs (i.e., spelling errors) should have been labeled with error annotations that indicate the spelling principle that was violated. To verify whether the target PCUs and incorrect original PCUs had been annotated correctly, for each of the 38 spelling principles used for error annotation and the 33 spelling principles used for basic annotation, 20 random words were generated that contain at least one PCU that is annotated with this principle (respectively in the error layer or in the basic layer). In these generated random words, it was checked whether the spelling principle annotation was used correctly to label a PCU. Subsequently, the precision (see Equation 1) was computed for each basic and error spelling principle by dividing the number of correctly used annotations by the total number of annotations (=20). This yields a number between 0 and 1 that indicates to what extent the spelling principle was used correctly to label PCUs. The higher the precision, the better the algorithm is in annotating PCUs with that spelling principle.

$$\text{Precision(X)} = \frac{\text{True Positives}}{\text{True Positives + False Positives}} = \frac{\text{PCUs truely labeled with X}}{\text{PCUs truely labeled with X + PCUS falsely labeled with X}} \quad (1)$$

In the first place, we found that for 32 out of 38 spelling principles used for error annotation, the precision was 1.0. For the other six error spelling principles, the precision scores were between 0.6 and 0.95 (see Table 6). Secondly, the computed precision of the basic spelling principles was 1.0 for 28 out of 33 basic spelling principles. The five remaining basic spelling principles had a precision between 0.2 and 0.95. Table 6 shows an overview of all spelling principles with a precision that is not equal to one. What stands out from this table is that spelling principles with low precision for basic annotation also have a low precision for error annotation. In addition, SyCoN1 has a really low basic precision, which is only 0.2 and MoAs1 has a low basic and error precision (respectively, 0.55 and 0.6).

3.2 Most problematic spelling principles

3.2.1 Relative Spelling Error Frequency

Spelling principles of some categories are much more frequent than others. For example, rule SyCap1 (every sentence starts with a capital letter) is applied once in every sentence, while rule SemCap1 (every proper name is written with a capital letter) is used less frequently, since not every sentence contains a proper name. In this study, we will refer to the frequency of incorrect applications of a spelling principle in relation to the frequency of both incorrect and correct applications using the Relative Spelling Error Frequency (RSEF). The RSEF is computed for each spelling principle X and grade G as follows:

Table 6: Precision scores for the six error spelling principles that have a precision lower than 1.0. Five of them also have a precision lower than 1.0 when used as basic annotation. See the Appendix for more examples of the spelling principles.

Spelling Principle			**Precision**	
Name	**Description**	**Example (original/target)**	**Error**	**Basic**
SyCoN1	Between -n needs to be written between two morphemes in some composition words.	bijekorf / bijenkorf	0.8	0.2
MoAs1	Assimilation of stem (unvoiced consonant is pronounced as voiced or vice versa).	zeldsame / zeldzame	0.6	0.55
MoMi1	Miniaturization	achtien / achttien	0.95	0.7
MoAsMi1	Assimilation of stem followed by miniaturization.	opod / opbod	0.8	1.0
MoCoS1	You write a “between s” between two parts of a composition word if you hear that “s”.	dorpstraat / dorpsstraat	0.95	0.95
MoCoS2	If you don’t hear a “s” between two parts of a composition word, you don’t write one.	hoofdsstraat / hoofdstraat	0.75	0.75

$$\text{RSEF(G, X)} = \frac{\text{Nr. of PCUs in grade G labeled with } \textbf{error} \text{ spelling principle X}}{\text{Nr. of PCUs in grade G labeled with } \textbf{basic} \text{ spelling principle X}} \times 100\% \qquad (2)$$

In Equation 2, variable X can also represent a category instead of a spelling principle. The RSEF can only be computed on the basic spelling principles, which is a subset of the error spelling principles.

3.2.2 Categories of spelling principles

In total, we labeled 22,220,072 PCUs (of which 896,636 PCUs are written incorrectly). These PCUs form together 6,028,023 words and are extracted from 70,593 different BasiScript texts written by children from grade 2 to grade 6. To determine which category of spelling principles is most problematic to learn, we compute the Relative Spelling Error Frequency (RSEF; see formula 2 in section 3.2.1). This is a measure that represents the error frequency of a spelling principle in relation to how often that spelling principle should be applied. The RSEF scores of each grade and category (Figure 2), show that spelling principles belonging to the Marked by Semantics category were relatively most frequently violated and appear to be most problematic to master in all grades. Secondly, spelling principles from the Marked by Syntax category were problematic. In all categories, the RSEF score decreased with increasing grade. The smallest decrease is found for spelling principles from the Marked by Syntax category.

3.2.3 Individual spelling principles

The Marked by Semantics category only consists of one spelling principle: SemCap1 (every proper name, title, and some abbreviations should start with a capital letter). Thus, the RSEF score of this spelling principle is represented by the Marked by Semantics category in Figure 2. This figure shows that there was almost no awareness of SemCap1 in grade 2, because the principle was applied incorrectly 88.89% of the times. This percentage decreased over the grades, but in sixth grade, the rule is still applied incorrectly in more than one third (36.51%) of the times that sixth graders have to use this rule.

After that, the RSEF scores of the individual spelling principles from the Marked by Syntax category were inspected (Figure 3). In the first place, these scores show that children became better

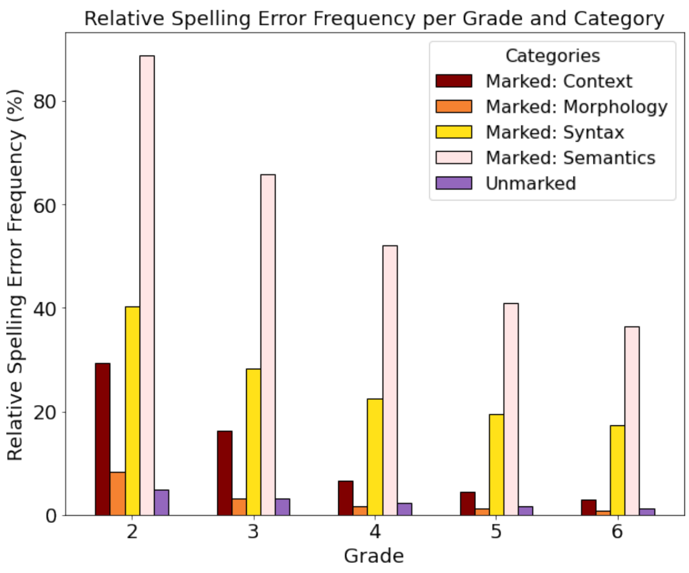


Figure 2: The Relative Spelling Error Frequency (RSEF) per category and grade. The higher the score of a grade, the more problematic to learn the spelling principles belonging to that category are. The bars for the category Marked by Semantics also represent the RSEF scores for spelling principle SemCap1, because this is the only spelling principle that the Marked by Semantics category consists of.

in applying all these spelling principles, because the RSEF scores decreased when children got older. Additionally, there were three spelling principles that had in all grades the highest RSEF score and were still applied incorrectly in more than 30% of the times that they should have been applied in sixth grade. These spelling principles are SyVd2 (past participles ending in a voiced PCU should get “d” as a suffix, and not “t”, while the suffix is pronounced with a /t/ sound), SyOd1 (every present participle should end in a “d”, while a /t/ sound is heard) and SyCap1 (every sentence should start with a capital letter). Thus, these three spelling principles were most problematic to learn for the children. In Figure 3, the spelling principles are addressed using their abbreviated names. A description of the spelling principles can be found in Table 10 and 11 of the Appendix.

The necessity to apply a spelling principle is for some spelling principles more frequent than for others. For example, the principle “each sentence starts with a capital letter” (SyCap1) should be applied every sentence, while the principle “write every proper name with a capital letter” (SemCap1) is less often necessary to apply, since not every sentence contains a proper name. Therefore, we also computed the normalized absolute error frequencies of the four spelling principles with the highest RSEF, which are SemCap1, SyVd2, SyOd1 and SyCap1 (see Figure 4). To compute this absolute score, we count for each grade the total number of times a specific spelling principle is applied incorrectly. We normalize this absolute score by dividing this score by the total number of PCUs (both correct and incorrect) in the respective grade. From the absolute scores in Figure 4 we can conclude that not having mastered the principles SyCap1 and SemCap1 results in more errors than not having mastered SyVd2 and SyOd1.

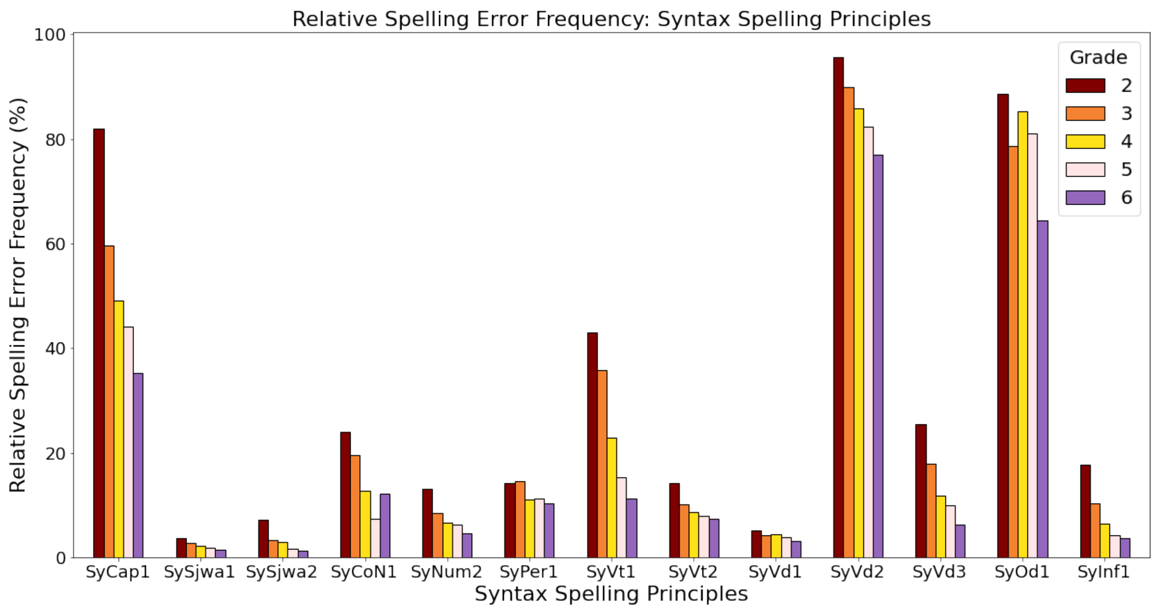


Figure 3: The Relative Spelling Error Frequencies (RSEFs) of the Marked by Syntax spelling principles.

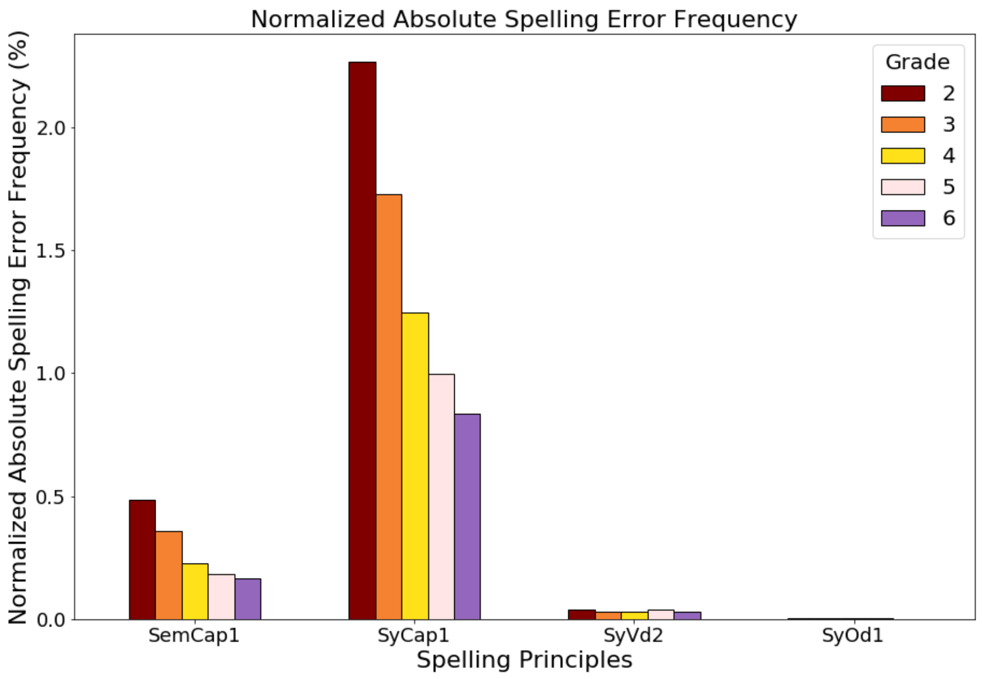


Figure 4: The normalized absolute spelling error frequencies of spelling principles SemCap1, SyCap1, SyVd2 and SyOd1.

4. Discussion

The first aim of this research was to investigate whether it is possible to extend the spelling error detection and annotation algorithm that we presented in an earlier study (Harmsen et al. 2021) so that it is also capable of annotating each PCU of a correctly written Dutch word indicating which spelling principle should be applied to spell this PCU correctly (i.e., orthographic property annotation). We found that this is possible, which confirmed our hypothesis for the first research question.

We found that by extending the algorithm, it is possible to determine for each spelling principle how often it is violated with respect to how often it could potentially be applied (i.e., the relative frequency). This is a measure to express how well children have mastered specific spelling principles, and thus to determine how problematic to learn they are. We found that spelling principles from the semantics and syntax category have the highest relative frequency in all grades of primary school and are therefore most problematic to learn.

Within the syntax and semantics category, there are four spelling principles that are still applied incorrectly by sixth graders in more than 30% of the times that they are used. Two of these spelling principles concern capital letter use and the other two concern verb spelling (i.e., the writing of present participles and the writing of past participles ending in "d", while you hear a /t/ sound). Thus, the hypothesis of research question 2 is partially confirmed, since not only verb spelling is found to be problematic (a finding in line with Bosman (2005) and Schijf (2009)), but also capital letter spelling.

Primary school children do not use present participles (SyOd1), like "dansend" (dancing) or "werkend" (working), and proper names and titles (SemCap1), like "Nijmegen" (Nijmegen), very frequently. This can be deduced from the fact that words of which at least one PCU is marked with "SyOd1" or "SemCap1" in the basic layer are very infrequent in the analyzed texts from the BasiScript corpus. The infrequent necessity to apply these principles can be a possible explanation for the fact that these spelling principles have a high relative frequency. Writers forget about these principles or don't learn to apply them correctly, because they almost never write words for which they need to apply them (Assink 1985). Since the rules that describe capital letter use and present participle writing are quite straightforward and do not have many exceptions, we believe that more attention to direct instruction of these spelling principles during spelling lessons could considerably improve the spelling proficiency of Dutch children.

4.1 Limitations

4.1.1 Limitations to the algorithm design

First, three limitations with respect to the design of the algorithm will be discussed. A problem that occurred while applying the algorithm to the BasiScript texts is that it could not align the target text and original text of some of the files correctly, which resulted in the fact that splitting the aligned target and original texts into tokens (i.e., words and punctuation marks) was not always successful. The result is that some original tokens are aligned with empty target tokens, and thus cannot be matched with the lemmas, morphemes, and POS-tag annotations that each BasiScript file contains at word level. This problem occurs for example often when aligning uncommon sequences of punctuation marks, like "!?!". To solve this problem, all texts in which this phenomenon occurred were removed from the data. This is a substantial loss of data (16,075 texts; 18.5% of all texts). A better way to overcome this problem would involve changing the method of splitting texts into words, for example by using a Dutch word and punctuation mark tokenizer function provided by the Python package UCTO, which is also incorporated in Frog (Bosch et al. 2007).

Another limitation with respect to the design of the algorithm concerns the grapheme-phoneme alignment process, in which the graphemes of the target word are aligned with their phonemes, which results in a PCU-segmentation. The problem is that this phoneme-grapheme alignment algorithm is not perfect. For example, some rules are missing, such as “b” is only defined with a /b/ sound, and not with a /p/ sound. Moreover, the algorithm has problems with the alignment of some foreign loan words, like “Youtube” and “Wiien”, in which graphemes are pronounced with sounds that are not common in Dutch (and thus are not defined as rules in the algorithm). In the current study, all words for which a correct alignment of the graphemes and phonemes of the target word was not possible were deleted. A better way to overcome this problem would be to extend the phoneme-grapheme alignment algorithm with extra rules.

The third limitation concerns the low precision scores for six spelling principles used for error annotation, of which five also have low precision for annotation of target PCUs. This means that some spelling errors or target PCUs are labeled incorrectly with these spelling principles. The algorithm can be improved by changing the functions that recognize these errors and target PCUs in the data, so that a higher precision is achieved.

4.1.2 Limitations to the algorithm evaluation method

There are also two limitations with respect to the method that was chosen to evaluate the algorithm. The major concern is that the automatic annotations are not compared with manual annotations. This comparison would allow computation of the recall metric next to the precision metric. The recall metric is useful, because it provides insight into the number of false negatives, which are spelling errors that are annotated with an unmarked spelling principle, while they should actually be annotated with a marked spelling principle. False negatives are currently defined in this way, because the algorithm first checks for all marked spelling principles whether they are present, and if this is not the case, the algorithm uses an unmarked annotation. This means that all spelling errors that do not fit into a marked category are annotated with an unmarked spelling principle.

In addition, the currently computed precision scores are for each spelling principle based on a sample of only twenty words, containing at least one PCU that is annotated with the currently selected spelling principle. This is a small sample, especially for high-frequent spelling errors, and makes it possible that certain incorrect annotations are not detected. To increase the reliability of the algorithm in a follow-up study, it is recommended that evaluation of the algorithm encompasses the comparison of automatic annotations with manual annotations and the computation of both precision and recall.

4.2 Possible future directions

4.2.1 Improvement and extension of the algorithm

The current algorithm already performs well on detecting and annotating both correctly and incorrectly written PCUs in the BasiScript corpus, but the performance can still be improved on several aspects, so that the earlier described limitations are overcome. These improvements involve that all texts are split correctly into words, the grapheme-phoneme alignment algorithm is also able to align words with foreign roots, and all spelling principles have a high precision and recall, when used as error annotation, but also when used as basic annotation.

The developed algorithm contains 38 spelling principles to annotate spelling errors and 33 spelling principles to annotate target PCUs. They cover most spelling principles, but not all. For example, there are currently no spelling principles in the annotation scheme concerning the spelling of strong verbs in the past tense. For example, if someone writes “loopte” instead op “liep” (walked), this is currently recognized as an “oo”-“ie” substitution (UnSub2) and insertion of “t” (UnIns1) and “e” (UnIns1). Adding these spelling principles would make the algorithm more detailed and better able to recognize all types of errors.

The annotation scheme also does not contain spelling principles to annotate punctuation marks. Therefore, punctuation marks are currently not analyzed. The same applies to tokens that contain or are digits. In a future study, the annotation scheme can be extended, so that punctuation use can be analyzed automatically. This can yield very interesting insights, because this has not been done before on such a large scale for Dutch writing by children.

Another way to extend the algorithm is by changing the output format. In the current algorithm, the output containing over six million annotated words is saved in one csv file. A disadvantage is that this file is very large. A better output format would be to add new spelling error annotation layers to the BasiScript FoLiA files (Gompel and Reynaert 2013) of each text.

4.2.2 Further analysis of the BasiScript corpus

This study yields a multilayered analysis of more than six million words extracted from the BasiScript corpus. These layers include information about the child that wrote the word, but also about each word's orthographic properties, phonemes, PCUs, and spelling errors. Using these layers, the spelling errors in the BasiScript corpus can be explored in more detail than was done in this study.

In a possible future study, the spelling development can, for example, be analyzed in greater detail by involving information from the layer "date". This layer states whether the text was written in spring or autumn, which makes it possible to split each grade into two grades, for example "grade 5 autumn" and "grade 5 spring".

In addition, the spelling errors in the BasiScript corpus could be analyzed in greater detail. For example, several studies (Sandra et al. 1999, Bosman 2005, Schmitz et al. 2018) have analyzed d/t-errors in Dutch verb spelling and found that spellers are more likely to write "d" instead of "t" at the end of a verb than writing "t" instead of a "d". These studies were based on adults or older children and the results observed appear to be an effect of frequency and exposure. In future research, it would be interesting to investigate whether we find this preference in children in primary school who, by definition, have received less exposure.

Another direction for further exploration of the BasiScript corpus is research into the development of individual spelling performance. Such a longitudinal study is possible, because many children that wrote the BasiScript texts participated in subsequent data collection rounds.

In addition, it is possible to extract statistical information from the BasiScript corpus, like the most frequently misspelled words and PCUs and why these words and PCUs are misspelled. Especially in combination with metadata, like gender and grade of the author, the part-of-speech tag of a word, and the theme of the text the word belongs to, it is possible to improve the spelling lessons, so that they better fit the needs and perceptions of the children.

Another corpus, BasiLex (Tellings et al. 2014), consists of Dutch texts that are specifically written for children. From these texts, word frequencies can be obtained of all words that children from a specific grade are exposed to. Using these word frequencies, it is possible to investigate the relationship between how often a child is exposed to a word and the spelling errors made in that word. This is another route that could be explored in a future study.

4.2.3 Machine learning applications

Using the multi-layered annotations of each word, it is also possible to extract certain features from each text, like the number of errors in a specific category, the rate at which nouns are used, or the occurrence of certain phonemes. These text-based features enable the training of machine learning algorithms on texts written by children.

Possible supervised applications of these algorithms are authorship identification or grade prediction. In addition, a possible unsupervised application is to cluster the texts to discover similarities and differences between texts that tell something about the spelling performance of the writer and whether the writer is a weak, normal or strong speller.

4.2.4 Other applications of the algorithm

Next to automatic analysis of written, digitized texts, like the texts in the BasiScript corpus, the spelling error detection and annotation algorithm can also be used in other applications. For example, in combination with an automatic spelling error correction system, the algorithm can be implemented in a computer application that provides detailed online, real-time feedback on writing. Next to marking the PCU that is written incorrectly, the computer application can also explain why the PCU is written incorrectly by giving information about the violated spelling principle. Because direct instruction of the spelling principles of the Dutch orthographic system Assink (1986) and immediate, not-delayed, feedback on writing Harward et al. (1994) are essential for developing good spelling skills, it is expected that this computer application could have great potential for improving the writing skills of Dutch children.

Another possible application of the algorithm would be in research on children with spelling problems, like dyslexia. The algorithm can for example help to design tests in which freely-written texts of children are analyzed, instead of dictations. These new tests can, for example, support the measurement of children's spelling proficiency, so that the effectiveness of certain treatments can be assessed more easily. Another advantage of these tests is that they can help to identify children with dyslexia at an early stage. This is necessary, because a recent study on the spread of dyslexia in the Netherlands has found that in 2019, 7.5% of the children were in possession of a dyslexia statement after the last year of primary school and 14% after the last year of high school (Inspectie van Onderwijs 2019). These numbers show that dyslexia is a common disorder among Dutch children, and that almost half of the children suffering from it are not diagnosed during primary school. This means that the spelling problems of many children are recognized too late, which makes it harder to help them.

5. Conclusions

The present research shows that it is possible to develop an algorithm that automatically detects spelling errors in the BasiScript corpus and annotates them with the spelling principle that is violated. In addition, the algorithm is able to annotate correctly written PCUs with the spelling principle that was applied correctly. This algorithm, applied to the BasiScript corpus, facilitates automatic quantitative research to spelling errors made by Dutch primary school children in freely written texts.

Analysis of spelling errors in more than six million words from the BasiScript corpus shows that spelling principles concerning knowledge about Dutch syntax and semantics are the most problematic to learn. These quantitative results are very useful to improve direct spelling instruction, which is essential for developing good spelling skills (Assink 1986).

In future studies, further analysis of spelling errors in the BasiScript corpus can provide more detailed insights into which spelling errors are made by which type of children. In addition, it is possible to employ the algorithm in a computer application that provides support on spelling. Using the algorithm together with an automatic spelling checker, the application can provide immediate feedback on written text, indicating not only which letters are written incorrectly, but also what the underlying reason is why these letters are incorrect by displaying the violated spelling principle. Such an application can support direct and immediate feedback on writing, which is essential for becoming a proficient speller (Harward et al. 1994).

References

Assink, E. M. H. (1985), Assessing spelling strategies for the orthography of Dutch verbs, *British Journal of Psychology* **76**, pp. 353–363.

Assink, E. M. H. (1986), Verkennen kinderen spontaan orthografische regels?, *Tijdschrift voor Taalbeheersing* **8**, pp. 106–118.

Berkling, K. and R. Lavalley (2015), WISE: A web-interface for spelling error recognition for German: A description and evaluation of the underlying algorithm, *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology* **40A**, pp. 87–96.

Borgwaldt, S.R., F. Hellwig, and A.M.B. de Groot (2004), Word-initial entropy in five languages: Letter to sound and sound to letter, *Written Language and Literacy* **7**, pp. 165–184.

Bosch, A. van den, B. Busser, S. Canisius, and W. Daelemans (2007), An efficient memory-based morphosyntactic tagger and parser for Dutch, *Computational linguistics in the Netherlands: Selected papers from the seventeenth CLIN Meeting* pp. 99–114.

Bosch, L. ten (n.d.), Grapheme to phoneme converter. Available at: https://webservices.cls.ru.nl/g2pservice/ (Accessed: 19 Jan. 2021).

Bosman, A. (2005), Development of rule-based verb spelling in Dutch students, *Written Language and Literacy* **8**, pp. 1–18.

Cordewener, K. A. H., L. Verhoeven, and A. M. T. Bosman (2016), Improving spelling performance and spelling consciousness, *The Journal of Experimental Education* **84**, pp. 48–74.

Cucchiarini, C. (1993), *Phonetic transcription: a methodological and empirical study*, Nijmegen, The Netherlands.

Cucchiarini, C. (1996), Assessing transcription agreement: Methodological aspects, *Clinical Linguistics and Phonetics* **10**, pp. 131–155.

Elffers, B., C. van Bael, and H. Strik (2013), *ADAPT: Algorithm for Dynamic Alignment of Phonetic Transcriptions*, Nijmegen, The Netherlands.

Gompel, M. van and M. Reynaert (2013), FoLiA: A practical xml format for linguistic annotation: a descriptive and comparative study, *Computational Linguistics in the Netherlands Journal* **3**, pp. 63–81.

Harmsen, Wieke (2022), Algorithm for Automatic Detection and Annotation of Spelling Errors in Dutch Texts. https://github.com/WiekeHarmsen/SpellingErrorsBasiScript.

Harmsen, W.N., C. Cucchiarini, and H. Strik (2021), Automatic quantitative analysis of spelling errors in texts written by sixth graders, *Proceedings of the 13th International Conference on Education and New Learning Technologies* pp. 8937–8945.

Harward, S. V., R. A. Allred, and R. R. Sudweeks (1994), The effectiveness of four self-corrected spelling test methods, *Reading Psychology: An International Quarterly* **15**, pp. 245–271.

Horbach-Kleijnen, R. (1992), *Hardnekkige spellingfouten: Een taalkundige analyse*, Swets & Seitlinger, Lisse.

Horbach-Kleijnen, R. (1997), *Strategieën van zwakke lezers en spellers in het voortgezet onderwijs*, Swets & Seitlinger, Lisse.

Inspectie van Onderwijs (2019), *Dyslexieverklaringen: Verschillen tussen scholen nader bekeken.*

Keuning, J. and L. Verhoeven (2008), Spelling development throughout the elementary grades: The Dutch case, *Learning and Individual Differences* **18** (4), pp. 459–470.

Laarmann-Quante, R. (2016), Automating multi-level annotations of orthographic properties of German words and children's spelling errors, *Proceedings of the 2nd Language Teaching, Learning and Technology Workshop* pp. 14–22.

Leerdam, M. van, A. M. T. Bosman, and G. C. van Orden (1998), The ecology of spelling instruction: Effective training in first grade, *in* Reitsma, P. and L. Verhoeven, editors, *Problems and interventions in literacy development*, Kluwer Academic Publishers, Dordrecht, pp. 307–320.

Nunn, A. (1998), Dutch orthography: A systematic investigation of the spelling of Dutch words, Holland Academic Graphics, Den Haag.

Sandra, D. (2010), Homophone dominance at the whole-word and sub-word levels: Spelling errors suggest full-form storage of regularly inflected verb forms, *Language and Speech* **53** (3), pp. 405–444.

Sandra, D. and L. van Abbenyen (2009), Frequency and analogical effects in the spelling of full-form and sublexical homophonous patterns by 12 year-old children, *The Mental Lexicon* **4** (2), pp. 239–275, John Benjamins. https://www.jbe-platform.com/content/journals/10.1075/ml.4.2.04san.

Sandra, D., S. Frisson, and F. Daems (1999), Why simple verb forms can be so difficult to spell: The influence of homophone frequency and distance in Dutch, *Brain and Language* **68** (1), pp. 277–283.

Schijf, T. (2009), *Lees- en spellingvaardigheden van brugklassers*, SCO-Kohnstamm Instituut, Amsterdam.

Schmitz, T., R. Chamalaun, and M. Ernestus (2018), The Dutch verb-spelling paradox in social media: A corpus study, *Linguistics in the Netherlands* **35** (1), pp. 111–124, John Benjamins.

Schrödel, M. and J. Trouvain (2003), The German text-to-speech synthesis system MARY: A tool for research, development and teaching, *International Journal of Speech Technology* **6**, pp. 365–377.

Taalunie (2014), Corpus Gesproken Nederlands - CGN (version 2.0.3). Available at Dutch Language Institute, http://hdl.handle.net/10032/tm-a2-k6.

Tellings, A., M. Hulsbosch, A. Vermeer, and A. van den Bosch (2014), Basilex: An 11.5 million words corpus of Dutch texts written for children, *Computational Linguistics in the Netherlands* **4**, pp. 191–208.

Tellings, A., N. Oostdijk, I. Monster, F. Grootjen, and A. van den Bosch (2018), BasiScript: : A corpus of contemporary Dutch texts written by primary school children, *International Journal of Corpus Linguistics* **23** (4), pp. 494–508.

Van Rossum, Guido and Fred L. Drake (2009), *Python 3 Reference Manual*, CreateSpace, Scotts Valley, CA.

Verhaert, N., E. Danckaert, and D. Sandra (2016), The dual role of homophone dominance. why homophone intrusions on regular verb forms so often go unnoticed, *The Mental Lexicon* **11** (1), pp. 1–25.

6. Appendix

Tables 6 to 11 show the annotation scheme that is used in this study to annotate the incorrectly spelled PCUs (error layer) and correctly spelled PCUs (basic layer). Each table contains the spelling principles of one category: Unmarked (Table 6), Marked by Context (Table 7), Marked by Morphology (Table 8), Marked by Syntax (Table 9 and 10), and Marked by Semantics (Table 11).

Each spelling principle is numbered and has a unique name in the "Error" column. This tag is used to make annotations in the error layer. For annotation in the basic layer, the names in the "Basic" column are used. These are not unique for every spelling principle, because it was in some cases not possible to distinguish between the different spelling principles when only taking into account the target spelling, and not the original spelling. In total, there are 38 unique error spelling principles and 32 unique basic spelling principles.

Most spelling principles consist of several sub-spelling principles. These are printed in italics below the corresponding spelling principle. For example: UnSub1 consists of UnSub1a and UnSub1b. This means that UnSub1a and UnSub1b are used to annotate PCUs in the data set. UnSub1 is the superset of UnSub1a and UnSub1b. So each spelling principle annotated with UnSub1a or Unsub1b is also annotated with UnSub1.

Table 7: Spelling principles from the Unmarked category

	Theme	Name		Description of spelling error	Examples	
		Error	Basic		Target	Original
1	Deletion	UnDel1	Un	Deletion of PCU	straa**t**	straa
2	Insertion	UnIns1	Ins	Insertion of PCU	school	sch**r**ool
3	Substitution	UnSub1	Un	Substitution of PCU with another PCU that maps to the same phoneme		
		UnSub1a	*Un*	*A consonant is not doubled if it's written after a long vowel and if it's not at the end of the word*	ko**k**en	ko**kk**en
		UnSub1b	*Un*	*Substitution of PCU with another PCU that maps to the same phoneme, which is not UnSub1a*	p**au**w	p**ou**w
4		UnSub2	Un	Substitution of PCU with another PCU that maps to another phoneme.		
		UnSub2a	*Un*	*Target PCU is reversed original PCU.*	kl**ei**n	kl**ie**n
		UnSub2b	*Un*	*Graphemes of PCU are partly deleted*	r**eu**s	r**e**s
		UnSub2c	*Un*	*Graphemes of PCU are partly inserted*	b**i**nnen	b**ui**nen
		UnSub2d	*Un*	*Complete substitution of PCU*	b**ui**ten	b**oe**ten
5		UnSub3	Un	Substitution involving capital letter PCUs		
		UnSub3a	*Un*	*Every not-first letter of a sentence and non-name is written with a lowercase letter*	**k**at	**K**at
		UnSub3b	*Un*	*Substitution of capital letter with lowercase letter that is not SemCap1 or SyCap1*	**k**at	**K**at

Table 8: Spelling principles from the Marked by Context category

	Theme	Name		Description of spelling error	Examples	
		Error	Basic		Target	Original
6	Vowel Singulation	CoVs1	CoVs1	A long vowel is written with one vowel symbol if it is at the end of a syllable	m**a**ken	m**aa**ken
7		CoVs2	CoVs2	Exceptions: A long vowel is written with two vowel symbols at the end of a syllable in several cases:		
		CoVs2a	*CoVs2a*	*A long vowel is written with two letters if it is followed by "ch" to avoid confusion with the sjwa*	g**oo**chelaar	g**o**chelaar
		CoVs2b	*CoVs2b*	*A long "ee" is written with two symbols at the end of a word to avoid confusion with the sjwa*	z**ee**	z**e**
		CoVs2c	*CoVs2c*	*A long "ie" is written with two* vowels at the end of a word	dr**ie**	dr**i**
		CoVs2d	*CoVs2d*	*A long vowel is written with two vowels before a diminutive suffix*	l**aa**tje	l**a**tje
8	Consonant Doubling	CoCd1	CoCd1	A consonant is doubled if it is written after a short vowel (excluding sjwa) and if it's not at the end of the word	jo**kk**en	jo**k**en
9	Special Cases	CoSc1	CoSc1	The "w" before an "r" is pronounced as a /v/	**w**reken	**v**reken
10		CoSc2	CoSc2	A /w/ or /j/ pronounced between two vowels is not written	januari, eieren	janu**w**ari, ei**j**eren
11		CoSc3	CoSc3	When a "w" is pronounced after an /ee/ or /ie/ sound, an "u" should be written before the "w"	snee**u**w, nie**u**w	sneew, nieuw
12	Accents	CoAc1	CoAc1	Some vowels need an accent to simplify pronunciation	Belgi**ë**, caf**é**	Belgi**e**, caf**e**
13		CoAc2	Un	Most vowels don't need an accent to simplify pronunciation	k**a**t	k**ä**t
14	Apostrophe	CoAp1	CoAp1	An apostrophe is written in several cases:		
		CoAp1a	*CoAp1*	*An apostrophe is written in proper names that end in an sis-sound and are used as genitive*	Frits**'**	Frits
		CoAp1b	*CoAp1*	*An apostrophe is written in plural and genitive forms that people could read wrongly with a short vowel instead of long vowel*	opa**'**s	opas
		CoAp1c	*CoAp1*	*Apostrophe is written in diminutives of words ending in a consonant + "y"*	baby**'**tje	babytje
		CoAp1d	*CoAp1*	*Apostrophe is written in one symbol words or abbreviations before suffixes of plural and genitive forms*	h.b.o.**'**er' sms**'**t	h.b.o.**-**er sms**-**t
		CoAp1e	*CoAp1*	*Apostrophe is written instead of other letters to shorten a word*	zo**'**n	zoon

Table 9: Spelling principles from the Marked by Morphology category

	Theme	Name		Description of spelling error	Examples	
		Error	Basic		Target	Original
15	Assimilation	MoAs1	MoAs1	Assimilation of stem		
		MoAs1a	*MoAs1*	*Unvoiced consonant is pronounced as voiced*	stee**d**s	stee**t**s
		MoAs1b	*MoAs1*	*Voiced consonant is pronounced as unvoiced*	zeld**z**ame	zeld**s**ame
16	Miniatu-rization	MoMi1	MoMi1	Miniaturization (Geminaatdelging)	ach**tt**ien	ach**t**ien
17	Ass. and Min.	MoAsMi1	MoAsMi1	Assimilation of stem, followed by miniaturization	o**pb**od	o**b**od
18	Final Devoicing	MoFd1	MoFd1	Final Devoicing: Voiced obstruent becomes voiceless at end of word		
		MoFd1a	*MoFd1*	*Voiced obstruent "d" is written as "t"*	hon**d**	hon**t**
		MoFd1b	*MoFd1*	*Voiced obstruent "b" is written as "p"*	clu**b**	clu**p**
19		MoFd2	MoFd2	Exception Final Devoicing: Words cannot end in a voiced obstruent		
		MoFd2a	*MoFd2*	*Words cannot end in the voiced obstruent "v"*	wer**f**	wer**v**
		MoFd2b	*MoFd2*	*Words cannot end in the voiced obstruent "z"*	mui**s**	mui**z**
20	Silent "t"	MoEndT1	MoEndT1	t is written, but /t/ is not pronounced	kas**t**je rech**t**door	kasje rechdoor
21	Silent "n"	MoEndN1	MoEndN1	n is written, but /n/ is not pronounced	binne**n** fietse**n**	binne fietse
22	Composition words	MoCoS1	MoCoS1	You write a "between s" between two parts of a composition word if you hear that "s" or when you hear it in similar words	dorp**s**weg, dorp**s**straat	dorpweg, dorpstraat
23		MoCoS2	MoCoS2	If you don't hear an "s" between two parts of a composition word, you don't write one	hoofdweg	hoofd**s**weg
24	Hyphen	MoHy1	MoHy1	Use a hyphen ...		
		MoHy1a	*MoHy1*	... *in case of word repetition*	zon**-** en feestdagen	zon en feestdagen
		MoHy1b	*MoHy1*	... *in case of word repetition*	abc**-**boek	abc boek
		MoHy1c	*MoHy1*	... *in names*	Gert**-**Jan	Gert Jan
		MoHy1d	*MoHy1*	... *in case of vowel collision*	zonne**-** energie	zonneenergie
		MoHy1e	*MoHy1*	... *in other categories*	-	-

Table 10: Spelling principles from the Marked by Syntax category (1/2)

	Theme	Name		Description of spelling error	Examples	
		Error	Basic		Target	Original
25	Number	SyNum1	SyNum1	Plural noun forms end in "s" or "n"		
		SyNum1a	*SyNum1*	*Plural noun forms end in "s"*	bureau**s**	bureau
		SyNum1b	*SyNum1*	*Plural noun forms end in "n"*	kante**n**	kante**r**
26		SyNum2	SyNum2	Plural verb forms end in -n (pv tt, pv vt, vd, od, inf)	fietse**n**	fietse**r**
27	Sjwa	SySjwa1	SySjwa1	Sjwa in non-verb suffixes: Some suffixes consist of an "e" (sjwa), or "e" (sjwa) + "n"	hel**e**	hel
28		SySjwa2	SySjwa2	Some suffixes have an "e" (sjwa) in second position after the "d" or "t"	dansend**e**	dansend
29	Composition words	SyCoN1	SyCoN1	Between -n needs to be written between two morphemes in some composition words	bije**n**korf	bijekorf
30	Person	SyPer1	SyPer1	Present 2nd/3th person singular forms have suffix "t"		
		SyPer1a	*SyPer1*	*"t" substituted with "d"*	loop**t**	loop
		SyPer1b	*SyPer1*	*"t" substituted with another character, which is not a "d"*	loop**t** vind**t**	loop vinds
31	Past Simple	SyVt1	SyVt1	When the stem of a verb ends in an unvoiced sound (the sounds in "'t exkofschip"), the suffix starts with an unvoiced /t/ sound ("t")		
		SyVt1a	*SyVt1*	*"t" substituted by "d"*	werk**t**e	werk**d**e
		SyVt1b	*SyVt1*	*"t" substituted by "tt"*	werk**t**e	werk**tt**e
		SyVt1c	*SyVt1*	*"tt" substituted by "t"*	wach**tt**e	wach**t**e
		SyVt1d	*SyVt1*	*"t" or "tt" substituted by PCU other than "d", "tt" or "t"*	werk**t**e wach**tt**en	werk**s**e wach**k**en
32		SyVt2	SyVt2	When the stem of a verb ends in an voiced sound (the sounds not in "'t exkofschip"), the suffix starts with a voiced /d/ sound ("d")		
		SyVt2a	*SyVt2*	*"d" substituted by "t"*	krab**d**e	krab**t**e
		SyVt2b	*SyVt2*	*"d" substituted by "dd"*	krab**d**e	krab**dd**e
		SyVt2c	*SyVt2*	*"dd" substituted by "d"*	bran**dd**e	bran**d**e
		SyVt2d	*SyVt2*	*"d" or "dd" substituted by other PCU than "t", "dd" or "d"*	krab**d**e bran**dd**e	krab**m**e bran**k**e

Table 11: Spelling principles from the Marked by Syntax category (2/2)

	Theme	Name		Description of spelling error	Examples	
		Error	Basic		Target	Original
33	Past Participle	SyVd1	SyVd1	If last letter of word stem in "'t exkofschip", suffix starts with "t"		
		SyVd1a	*SyVd1*	*"t" substituted by "d"*	gepak**t**	gepak**d**
		SyVd1b	*SyVd1*	*"t" substituted by other character, which is not a "d"*	gepak**t**	gepak**s**
34		SyVd2	SyVd2	If last letter of word stem not in "'t exkofschip", suffix starts with "d"		
		SyVd2a	*SyVd2*	*"d" substituted by "t"*	beloof**d**	beloof**t**
		SyVd2b	*SyVd2*	*"d" substituted by other character, which is not a "t"*	beloof**d**	beloof**b**
35		SyVd3	SyVd3	Some part participles end in -en	geroep**en**	geroep**t**
36	Present Participle	SyOd1	SyOd1	The suffix of present participle always starts with "d"		
		SyOd1a	*SyOd1*	*"d" substituted by "t"*	dansen**d**	dansen**t**
		SyOd1b	*SyOd1*	*"d" substituted by other character, which is not a "t"*	dansen**d**	dansen**s**
37	Capital	SyCap1	SyCap1	Every sentence starts with a capital letter	**H**allo	**h**allo

Table 12: Spelling principle from the Marked by Semantics category

	Theme	Name		Description of spelling error	Examples	
		Error	Basic		Target	Original
38	Capital	SemCap1	SemCap1	Every proper name, title and some abbreviations start with a capital letter	**N**ijmegen	**n**ijmegen