The Influence of Spelling Errors on Content Scoring Performance

Andrea Horbach, Yuning Ding, Torsten Zesch
Language Technology Lab,
Department of Computer Science and Applied Cognitive Science,
University of Duisburg-Essen, Germany
{andrea.horbach|torsten.zesch}@uni-due.de
yuning.ding@stud.uni-due.de

Abstract

Spelling errors occur frequently in educational settings, but their influence on automatic scoring is largely unknown. We therefore investigate the influence of spelling errors on content scoring performance using the example of the short answer data set of the Automated Student Assessment Prize (ASAP). We conduct an annotation study on the nature of spelling errors in the ASAP dataset and utilize these finding in machine learning experiments that measure the influence of spelling errors on automatic content scoring. Our main finding is that scoring methods using both token and character n-gram features are robust against spelling errors up to the error frequency seen in ASAP.

1 Introduction

Spelling errors occur frequently in educational assessment situations, not only in language learning scenarios, but also with native speakers, especially when answers are written without the help of a spell-checker.[1] In automatic content scoring for short answer questions, a model is learnt about which content needs to be present in a correct answer. Spelling mistakes interfere with this process, as they should be mostly ignored for content scoring. It is still largely unknown how severe the problem is in a practical setting.

Consider the following answer to the first prompt of the short answer data set of the Automated Student Assessment Prize (ASAP):[2]

(1) *Some additional information you will need are the material. You also need to know the size of the **contaneir** to measure how the acid rain **effected** it. You need to know how much **vineager** is used for each sample. Another thing that would help is to know how big the sample stones are by **measureing** the best possible way.*

In this answer, three non-word spelling errors (printed in bold) occur. In addition, there is also one real-word spelling error, which leads to an existing word: *effected*, which should be *affected*.

While a teacher who is manually scoring learner answers can simply try to ignore spelling mistakes as far as possible, automatic scoring methods must include a spell-checking component to normalize an occurrence of *vineager* to *vinegar*. Thus spell-checking components are also a part of some content scoring systems, such as the top two performing systems in the ASAP challenge (Tandalla, 2012; Zbontar, 2012). However, it is unclear what impact spelling errors really have on the performance of content scoring systems.

Many systems in the ASAP challenge, as well as some participating systems in the SemEval 2013 Student Response Analysis Task (Heilman and Madnani, 2013; Levy et al., 2013), used shallow features such as token n-grams (Zbontar, 2012; Conort, 2012). If a token in the test data is misspelled, then there is no way of knowing that it has the same meaning as the correct spelling of the word in the training data. At the same time, individual spelling error instances are often not occurring uniquely in a dataset: Depending on factors such as the learner group (for example native speakers or language learners with a certain native language) or the data collection method (handwriting vs. typing) some spelling errors will occur frequently while others will be rare. The mis-

[1]Note that we do not distinguish between the terms *error* and *mistake* used by Ellis (1994) to denote competence and performance errors respectively. We use the two terms interchangeably.

[2]https://www.kaggle.com/c/asap-sas

spelled form *vineger*, for example, might be frequent enough that an occurrence feature for the misspelled version provides valuable information which a classifier can learn. Whether this observation mitigates the effect of spelling errors depends on the frequency of individual errors and therefore also on the shape of error distributions.

Contributions In this paper, we investigate how the presence or absence of spelling errors influences the task of content scoring, taking the afore-mentioned influence criteria of error frequency and error distribution into account. We conduct our analyses and experiments on the frequently used ASAP content scoring dataset (Higgins et al., 2014). The dataset contains 10 different prompts about different topics ranging from sciences over biology to literature questions. Each prompt comes with 2,200 answers on average. Although this dataset has been used in a lot of studies concerning content scoring, much about the spelling errors in the dataset is still unknown. Our manual annotations and corpus analyses will therefore also provide insight on the number, the nature and the distribution of spelling errors in this dataset.

First, we present an analysis of the frequency and distribution of non-word spelling errors in the ASAP corpus and compare several spelling dictionaries. We provide a gold-standard correction for the non-word errors found automatically by a spell checker in the test section of the data. We compare error correction methods based on phonetic and edit distance and extend them with a domain-specific method that prefers suggestions occurring in the material for a specific prompt.

Next, we investigate the effect of manipulating the number and distribution of spelling errors on the performance of an automatic content scoring system. We experiment with two ways of regulating the number of misspellings. We automatically and manually spell check the corpus to replace non-word spelling errors by their corrected version. This only allows us to decrease the number of errors. To increase the amount of spelling errors further, we also introduce errors artificially in two conditions: (i) adding random noise as a worst-case scenario, and (ii) adding mistakes according to the error distribution in the test data.

We find that token and character n-gram scoring features are largely robust against spelling errors with a frequency present in our data. Character n-gram features are contributing towards this robustness. When introducing more errors, we see a substantial drop in performance, such that the importance of spell-checking components in content scoring depends on the frequency of errors in the data.

2 Annotating Spelling Errors

In order to evaluate the influence of spelling errors on content scoring, we need an error-annotated corpus. However, a full manual annotation of the complete dataset, which contains around one million tokens, was beyond our means. Instead, we decided to annotate a representative sample of the ASAP corpus which we utilize to evaluate the performance of spelling error detection methods. This allows us to estimate whether we can draw reliable conclusions from applying existing spell checking methods to the full dataset.

We manually annotated the first 20 answers in each prompt using WebAnno (Yimam et al., 2013). In order to facilitate the annotation process, we automatically pre-annotate potential spelling errors using the Jazzy spelling dictionary.[3] Two annotators (non-native speakers and two of the authors of this paper) reviewed the error candidates and either accepted or rejected them, but could also mark additional spelling errors which were not detected automatically.

In this manual annotation process, we distinguish between non-word and real-word spelling errors. We annotate a mistake as real-word error if another word with a different root is clearly intended in the context, such as "*Their* are two samples". We do not distinguish between spelling errors and grammatical errors among the non-word errors, i.e., we do not filter out non-words that could originate from grammatical errors such as incorrect 3rd person forms like *dryed* instead of *dried*. We do not mark grammatical errors that lead to a real-word error, such as wrong prepositions. Equally, we do not mark lexically unsuitable words which are morphologically possible, but do not fit in the context, such as *counter partner* in a context where *counter part* was clearly intended.

In total, we annotated 9,995 tokens and reach an inter-annotator agreement of 0.87 Cohen's kappa (Cohen, 1960) on the binary decision whether a

[3]`https://sourceforge.net/projects/jazzy/files/Dictionaries/English/english.0.zip/download`

word is a spelling mistake or not. Main sources of disagreement were (i) misses of real word spelling errors not marked in the pre-annotation (e.g. *koala beer*), and (ii) disagreements as to whether compounds may be written as one word or not (e.g. *micro debris* vs. *microdebris*), a decision which is often ambiguous. In case of disagreement between the annotators, the final decision is made through adjudication by both annotators.

The resulting dataset contains 297 spelling errors, including 48 real-word errors which will not be considered for further evaluations and experiments. The resulting ratio of spelling errors in the dataset is about 3%, which is in line with the expected frequency of spelling errors in human-typed text (Kukich, 1992). However, it would be an interesting follow-up work to determine the frequency of spelling errors in other content scoring datasets.

2.1 Evaluating Automatic Spell-checking

Using our annotated answers, we now evaluate different spell-checking dictionaries. Note that the size and quality of those dictionaries influence the trade-off between precision and recall of error detection. For example, a very small dictionary yields almost perfect recall, as most spelling errors are not in the dictionary and flagged accordingly. However, precision would be rather low as some of the detected errors are perfectly valid words that are just missing from the dictionary. On the other hand, a very large dictionary lowers recall as some words that are definitely a spelling error in the context of the writing task in this dataset might be valid words in some very special context. For example, the HunSpell dictionary contains the abbreviation *AZT*, standing among others for ‘azidothymidine’ and ‘Azerbaijan time’. In the context of our learner answers, the string would likely never occur as a valid word, but would be counted as a non-word misspelling.

In order to find a suitable dictionary for our task, we evaluate the following setups: As baseline dictionary, we use the one that comes with the **Jazzy** spell checker.[4] It is relatively small (about 47,000 entries) and does not contain inflected forms (such as third person singular). Thus, we also use the English **HunSpell** dictionary with more than 120,000 entries.[5] Both are general

Dictionary	**P**	**R**	**F**
Jazzy	.25	.98	.39
HunSpell	.63	.89	.74
HunSpell -abbr	.63	.95	.76
HunSpell +prompt	**.88**	.88	.88
HunSpell -abbr +prompt	.86	**.94**	**.90**

Table 1: Evaluation of different error detection dictionaries

purpose dictionaries, which we can adapt in order to get better performance. First, we remove all-uppercase abbreviations from the dictionary (**-abbr**), as they can lead to the above-mentioned problem.[6] Second, we extend the dictionary with prompt-specific lexical material (**+prompt**), which we extract automatically from the reading material and scoring rubrics associated with each prompt. This step adds about 600 tokens to the dictionary. Third, we combine both strategies (**-abbr +prompt**), keeping a word if it is contained both in the list of abbreviations and in the prompt material.

After checking the first results, we found that some artifacts influence the results. First, the tokenizer splits words like *can't* into two tokens *ca* and *n't*, which are then detected as spelling errors. Second, the learner answers often contain bullets used in lists, such as *a)*, *b)*, etc. For the final results, we do not count these cases as spelling errors.

Results Table 1 gives an overview of the results. As expected, the rather small Jazzy dictionary has very high recall as many words are not found in the dictionary, including almost all spelling errors but also a lot of valid words, which results in low precision. Using the larger HunSpell dictionary lowers recall a bit, but dramatically improves precision. Excluding abbreviations has less effect than expected. It might even hurt a bit, if words such as *DNA* are removed, which are frequently used in the biology prompts. Extending the dictionary with the small number of prompt-specific terms brings large boosts in detection precision with an – in comparison – moderate drop of recall. Excluding the abbreviations before adding the prompt-specific terms recovers most of the lost

[4]`http://jazzy.sourceforge.net`

[5]`https://sourceforge.net/projects/hunspell/files/Spelling%20dictionaries/en_US/`

[6]Note that in our experiments, we lowercase all material before the comparison so that we factor out capitalization problems.

recall and trades in some precision, but yields the best F-score.

It might be surprising that we do not reach a perfect recall in error detection, i.e. that there are words in the dictionary which we mark as incorrect. These include tokens from the prompt material which were annotated as incorrect (such as *microdebris* instead of *micro debris*), as well as erroneously tokenized words such as *a ddition* where both parts have been marked as part of a non-word spelling error although *a* appears in the dictionary.

Hunspell -abbr +prompt gives us overall the best performance and is thus used in the following experiments.

3 Annotating Error Corrections

In order to evaluate the performance of error correction methods, we manually correct part of the data. We use the **Hunspell -abbr +prompt** dictionary with the Jazzy spell-checker to detect and correct all errors in the test data part of ASAP. A list of these errors and their corrections are then presented to two human annotators (the same as in the previous annotation task). We showed each instance within a 20 character window to the left and right to allow for a decision in context, but annotators could inspect the full learner answer if that window was not sufficient. Annotators performed two consecutive tasks: First they accepted or rejected a word as a spelling error, thus sorting out words that should not have been flagged as an error in the first place. Next, they either accepted a proposed correction or changed it to a different one.

This approach was much less time consuming than performing manual error detection and correction on raw data, and allowed us to annotate the complete test data section of the corpus with 6,400 proposed error candidates. As the recall of the error detection approach is expected to be around 90-95% (see evaluation results in Table 1), we will miss some errors. However, we decided that an imperfect annotation of the full test data section is more useful than an almost perfect annotation of only parts of the answers.

For prompts 1 and 2, two annotators checked all instances on the training as well as on the test data. On these items, annotators reached an inter-annotator agreement of κ=0.90 on the decision whether a word should be considered an error. For those candidates considered an error by both annotators, they found the same correction in 86% of all cases. In addition, the test data for all 10 prompts was annotated by one annotator each. On this data, out of 6,400 error candidates, about 5,200 were accepted as errors. The resulting error detection precision of 81% is close to the values shown in Table 1.

There was a surprisingly high number of errors for which it was not possible to annotate a correction, because the answer was so garbled that annotators could not find a target hypothesis. An example for such a sentence is the following: *[. . .] but they don;t tell me to subtract the end mass from the* ***slurhnf***. When looking at the wider context, it is somewhat plausible that *slurhnf* should be something along the lines of *start mass*, it remains unclear what the student meant. We checked some of these candidates with a native speaker, but still remained with almost 3% of uncorrectable errors. We ignore these cases when we evaluate different spell-checkers.

3.1 Evaluating Automatic Error Correction

We compare four setups of error correction methods. As a baseline, we use the original Jazzy spellchecker[7], which only generates candidates with the same phonetic code using **Metaphone** encoding. If there is more than one candidate, we select one randomly.

As a variant to random error candidate selection, we additionally use prompt knowledge (**+prompt**), i.e. we make use of the prompt material and of the frequency of words and bigrams in all answers for a prompt. We prefer material occurring in the prompt of an answer, if there are several candidates, we take the one occurring most frequently in the data. We observed in our annotations that a number of identified errors (1,065) are the result of tokenization errors on the side of the students, i.e. they often omit whitespace between two words. Often these errors evolve around punctuation marks (e.g. *content.they*), in which cases they are easy to detect and correct. In cases without punctuation showing the token boundary, we check whether an unknown word can be split into two in-dictionary words and accept the candidate if the resulting bigram also occurs in the answers for that prompt.

We also built our own spellchecker, which does not only take phonetically identical candidates

[7]http://sourceforge.net/projects/jazzy

into account, but all candidates up to a certain **Levenshtein** distance (3 was an optimal value in our case) using LibLevenshtein[8] for an efficient implementation, but prefers candidates with a shorter distance if possible. In analogy to the *Metaphone* setup, we test (i) a basic version where a candidate is randomly selected should several occur and (ii) a prompt-specific version.

Table 2 shows correction accuracy of the different methods on the annotated gold-standard, i.e. we check how often the correction method found the same correction as annotated, ignoring capitalization and ignoring words we could not manually correct. We also show coverage values, which specify for how many gold standard errors the respective method was able to provide a correction candidate at all. For our following experiments, we select the best-performing **Levenshtein +prompt** method.

Method	**Variant**	**acc**	**coverage**
Jazzy	Metaphone	.51	.85
Jazzy	Metaphone +prompt	.55	.82
Our	Levenshtein	.46	.96
Our	Levenshtein +prompt	**.69**	.95

Table 2: Performance of different error correction methods

4 Dataset Analysis

To get a better understanding of the nature of spelling errors, we provide additional analyses on our annotations.

4.1 Error Detection Analysis

For the error detection annotations, we compare the length of a token in characters to its likelihood of being misspelled and find that longer words have higher chances to be misspellings (see Figure 1).

To further drill down on the nature of errors we compute the probability of spelling errors across different coarse-grained POS tags. We map the Penn Treebank tagset to 12 coarse-grained tags as described in (Petrov et al., 2011). Table 3 shows that error occur mainly in content words (only 17 out of 255 annotated errors occur in function words).

Next, we investigate, how many errors are automatically detected in ASAP using the best per-

[8]https://github.com/universal-automata/liblevenshtein-java

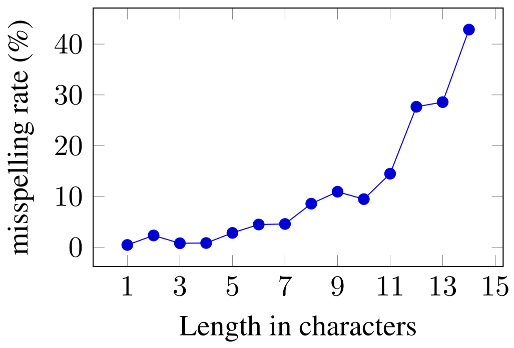

Figure 1: Probability for words of a certain length to be misspelled in our annotated data

POS	**# instances**	**P(error|POS)**
.	1	0.1
ADJ	26	4.2
ADP	5	0.5
ADV	27	4.8
CONJ	-	-
DET	3	0.3
NOUN	140	6.2
NUM	-	-
PRON	3	0.4
PRT	1	0.3
VERB	45	2.2
X	4	0.2

Table 3: Probability for tokens from a certain POS class to be misspelled.

forming dictionary. Table 4 provides an overview of the frequency of errors for each prompt, as well as the type-token-ratio for error tokens. We see that many errors occur more than once, which might have consequences for content scoring if a model is able to associate frequent misspellings with a certain label. Table 5 shows as an example the top 10 most frequent misspellings for prompt 2. We see that there are a few very frequent misspellings centered around important vocabulary for that prompt and a long tail of infrequent misspellings (not shown in the table).

We also check whether there is a correlation between the number of spelling errors in an answer and the content score assigned by a teacher. We normalize by the number of tokens in the answer to avoid length artifacts and find no significant correlation. This is in line with our general assumption that spelling errors are ignored by teachers when scoring a learner answer.

4.2 Error Correction Analysis

In order to understand the nature of spelling mistakes better, we perform additional analyses on the

prompt	# errors	TTR
1	1.2	.69
2	1.2	.64
3	1.0	.62
4	2.0	.51
5	7.7	.43
6	5.8	.62
7	1.5	.63
8	2.3	.42
9	2.1	.59
10	2.2	.43
∅	2.3	.56

Table 4: Average number of spelling mistakes per 100 tokens (punctuation excluded) and type-token-ratio for errors for the individual ASAP prompts.

Misspelling	#
streched	117
strech	31
strechable	18
nt	16
streching	15
expirement	13
streached	10
expiriment	9
strechiest	8
streches	6

Table 5: Top-10 most frequent misspellings for prompt 2

corrected test data. First, we categorized errors according to the Levenshtein distance between an error and its corrected version (see Figure 2). A number of instances with very high distances originate from errors involving tokenization, e.g. several words concatenated without a whitespace. To avoid such artifacts in the analysis, we counted only cases where both the original token and its corrected version did not include any whitespace.

There is still a surprisingly high number of words with a Levenshtein distance greater than 1. An example for a word with a high Levenshtein distance would be *satalight* instead of *satellite*. This shows that finding the right correction can be a challenging task, as correction candidates with a lower distance often exist. For example, in the answer *The students could have impared the experiment by (. . .)*, it becomes clear from the question context (*Describe two ways the student could have improved the experimental design*) that the right correction for *impared* is *improved* and not *impaired*.

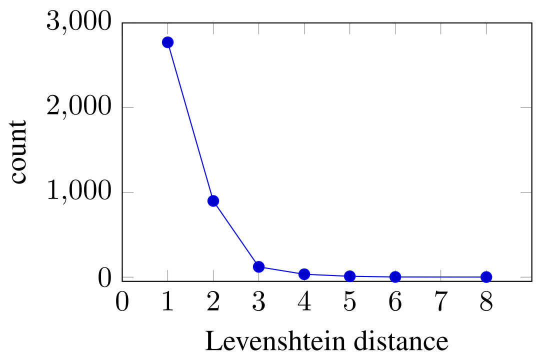

Figure 2: Distribution of errors across different Levenshtein distances.

5 Influence of Spelling Errors on Scoring

In the following experiments, we vary the amount of spelling errors in the data systematically. We use automatic and manual spell-checking to decrease the error rate and add artificial errors for a higher number of errors.

5.1 Experimental Setup

In our experimental studies, we examine the influence of spelling deviations and spell-checking on content scoring. We train one classification model for each of the ten ASAP prompts, using the published data split into training data and "public leaderbord" data for testing. We preprocess the data using the ClearNLP segmenter and POS tagger provided through DKPro Core (Eckart de Castilho and Gurevych, 2014). We use a standard feature set often used in content scoring (Higgins et al., 2014) and extract token 1–3 grams and character 2–4 grams using the top 10,000 most frequent n-grams in each feature group. We then train a SVM classifier (Hall et al., 2009) with default parameter settings provided through DKPro TC (Daxenberger et al., 2014). We evaluate using both accuracy and quadratically weighted kappa (QWK, Cohen (1968)), as proposed in the Kaggle competition for this dataset and present results averaged across all 10 prompts.

One important property of this feature setup is that the character n-gram features could be able to cover useful information from misspelled words. If the word *experiment* is, for example, misspelled as *expirment*, there are n-grams shared between these two versions, such as the character trigrams *exp*, *men* or *ent*. Therefore, we also use a reduced feature set, where we only work with token n-gram features, in order to quantify the size of this effect.

	tokens & chars		tokens only	
	QWK	acc	QWK	acc
baseline	.68	.71	.66	.70
spell-check train	.66	.70	.66	.70
spell-check test	.68	.70	.66	.70
spell-check both	.68	.70	.66	.70
gold test	.68	.70	.66	.70

Table 6: Scoring performance on ASAP with and without spell checking

# errors	# test items	QWK	acc
≥ 0	522	.66	.70
≥ 1	269	.65	.69
≥ 2	132	.63	.68
≥ 3	52	.62	.70

Table 7: Scoring performance on ASAP when using only answers with a certain minimal number of errors for testing.

5.2 Experiment 1 – Decreased Error Frequency

In a first machine learning experiment, we investigate the influence of spell-checking on the performance of content scoring. We systematically vary three sets of influence factors: First, we use either our automatically corrected learner answers as a realistic spell checking scenario or the corrected gold-standard version of the learner answers. We consider the latter an oracle condition to estimate an upper bound of improvement that filters out noise introduced by the spell checker. Second, we use two different feature sets: either the full feature set covering both token and character n-grams or the reduced feature set with only token n-grams. Third, we vary which part of the data is spell-checked. We either correct both the training and the test data or only training or only test data. In the oracle condition, we have only annotations for the test set, so that we use only the test condition here.

Table 6 shows the results. We see that it makes little difference whether we spell-check the data (be it automatically or manually). One possible explanation for the very small difference is that there are many answers without any spelling mistakes at all. Thus, we also comparing the performance for answers with different minimal number of errors. Table 7 shows the breakdown of the results per number of errors. We observe a reduced performance in kappa for answers with more spelling errors, but do not see that repeated in the accuracy, i.e. misclassified answers with more errors have a higher tendency to be completely misclassified.

5.3 Experiment 2 – Simulating Increased Error Frequencies

As we have seen in the above experimental study, there is little difference between the scoring quality on original data vs. spell-checked data. We already ruled out that this might be due to a noisy spell-checker by also evaluating on the annotated gold standard. Another potential reason for our findings is that the amount of errors present in the data is just not large enough to make a difference. To check that, we artificially introduce different amounts of new errors into the learner answers. In this way, we can also simulate corpora with different properties, so that practitioners can check the average amount of spelling errors in their data and can get an estimation of what performance drop to expect.

Generating Spelling Errors In order to generate additional spelling errors, we use two different models:

Random Error Generation introduces errors by either adding, deleting, or substituting a letter or by swapping two letters in randomly selected words. This error generation process is a worst case experiment in the sense that there is no predictable pattern in the produced errors.

Informed Error Generation produces errors according to the distribution of errors in the data. This means we introduce errors only to words that were misspelled in our annotated gold-standard, and we introduce errors by using the misspelled version actually occurring in the data considering the error distribution. In this way, there are chances that – like in real life – some errors will be more frequent than others such that a classifier might be able to learn from them.

We use both models with different configurations of the experimental setup. We vary whether the errors are added to all words (**all**) or only to content words (**cw**). This is because we observed that mainly longer and content words are misspelled. We argue that these words can be more important in content scoring than small function words. Therefore, those realistic errors might do more harm than random errors. In both conditions,

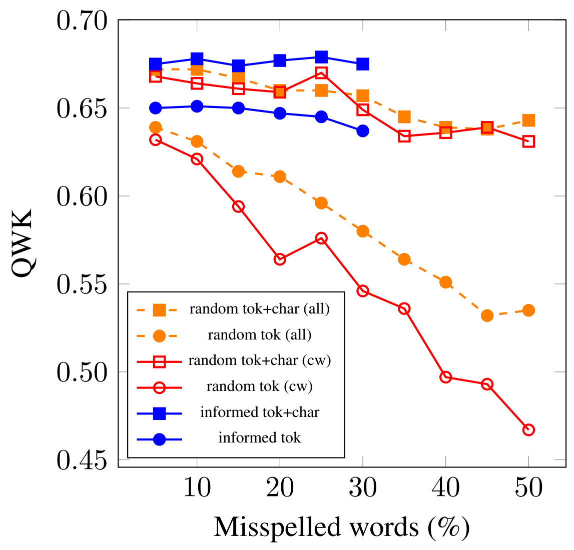

Figure 3: Scoring performance on ASAP with increasing amounts of additional introduced errors.

we make sure that the overall error rate across all tokens matches the desired percentage. Additionally, we use either only token features (**tok**), that will be more sensitive towards spelling errors, or also include the character features (**tok+char**), which we know to be more robust.

Figure 3 shows the performance of the two error generation models in their different variants for different amounts of artificial errors. Note that there was a natural upper bound for the amount of errors which can be introduced using the informed error generator. As expected, we see that content words are more important for scoring than function words as introducing errors to only content words yields a larger performance drop. We see for both generation models that a scoring model using character n-grams is largely robust against spelling mistakes while a model using only information on the token level is not. We also see that a more realistic error generation process is not as detrimental for the scoring performance as random errors. Of course, our error model might be slightly over-optimistic and in real life with such a high number of errors we might see new orthographic variants for individual words that were not covered in our annotations. We therefore believe the realistic curve to be somewhere between the informed and the random model.

6 Related Work

We are not aware of other works studying the impact of spelling errors on content scoring performance.

In general, spell checking is often used as a preprocessing step in educational applications, especially those dealing with input written by learners, either non-natives or natives. In some works the influence of spell-checking is explicitly addressed. Pilan et al. (2016) predict the proficiency level of language learners using textbook material as training data and find that spell-checking improves classification performance. Keiper et al. (2016) show that normalizing reading comprehension answers written by language learners is beneficial for POS tagging accuracy.

In some areas however, spelling errors can also be a useful source of information: In the related domain of native language identification, a discipline also dealing with learner texts, Chen et al. (2017) found that spelling errors provide valuable information when determining the native language of an essay's author.

7 Conclusions

We presented a corpus study on spelling errors in the ASAP dataset and provide gold-standard annotations for error detection and correction on large parts of the data.

Next, we examined the influence of spelling errors on content scoring performance. Surprisingly, we found very little influence of spelling mistakes on grading performance for our model and on the ASAP dataset. In our setup, spellchecking seems negligible.

There are several explanations for that: First, we found that we observe a drop in performance if we artificially increase the number of spelling errors. This drop is especially pronounced, if (a) no character n-gram information is used for scoring and (b) if errors follow no specific pattern. If a dataset is expected to contain a higher percentage of spelling errors it will therefore be helpful to correct errors automatically and/or to mitigate the effect of misspelled word by the usage of character n-gram features.

Second, our scoring models relied on shallow features. We assume that scoring models using higher linguistic processing such as dependency triples might suffer more substantially, a question that will be pursued further in future work.

Our gold standard annotations are available under `https://github.com/ltl-ude/asap-spelling` in order to encourage more work on spell-checking in the educational domain.

Acknowledgments

This work is funded by the German Federal Ministry of Education and Research under grant no. FKZ 01PL16075.

References

Lingzhen Chen, Carlo Strapparava, and Vivi Nastase. 2017. Improving native language identification by using spelling errors. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 542–546. http://aclweb.org/anthology/P17-2086.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement* 20(1):37–46.

Jacob Cohen. 1968. Weighted kappa - nominal scale agreement with provision for scaled disagreement or partial credit 70:213–20.

X Conort. 2012. Short answer scoring: Explanation of gxav solution. *ASAP Short Answer Scoring Competition System Description. Retrieved July* 28:2014.

Johannes Daxenberger, Oliver Ferschke, Iryna Gurevych, Torsten Zesch, et al. 2014. Dkpro tc: A java-based framework for supervised learning experiments on textual data. In *ACL (System Demonstrations)*. pages 61–66.

Richard Eckart de Castilho and Iryna Gurevych. 2014. A broad-coverage collection of portable nlp components for building shareable analysis pipelines. In *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT (OIAF4HLT) at COLING*. pages 1–11.

Rod Ellis. 1994. *The study of second language acquisition*. Oxford University.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The weka data mining software: an update. *ACM SIGKDD explorations newsletter* 11(1):10–18.

Michael Heilman and Nitin Madnani. 2013. Ets: Domain adaptation and stacking for short answer scoring. In *SemEval@ NAACL-HLT*. pages 275–279.

Derrick Higgins, Chris Brew, Michael Heilman, Ramon Ziai, Lei Chen, Aoife Cahill, Michael Flor, Nitin Madnani, Joel R Tetreault, Daniel Blanchard, Diane Napolitano, Chong Min Lee, and John Blackmore. 2014. Is getting the right answer just about choosing the right words? The role of syntactically-informed features in short answer scoring. *Computation and Language* http://arxiv.org/abs/1403.0801.

Lena Keiper, Andrea Horbach, and Stefan Thater. 2016. Improving pos tagging of german learner language in a reading comprehension scenario. In *LREC*.

Karen Kukich. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys (CSUR)* 24(4):377–439.

Omer Levy, Torsten Zesch, Ido Dagan, and Iryna Gurevych. 2013. Ukp-biu: Similarity and entailment metrics for student response analysis. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. volume 2, pages 285–289.

Slav Petrov, Dipanjan Das, and Ryan T. McDonald. 2011. A universal part-of-speech tagset. *CoRR* abs/1104.2086. http://arxiv.org/abs/1104.2086.

Ildiko Pilan, Elena Volodina, and Torsten Zesch. 2016. Predicting proficiency levels in learner writings by transferring a linguistic complexity model from expert-written coursebooks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan, pages 2101 – 2111.

Luis Tandalla. 2012. Scoring short answer essays. *ASAP Short Answer Scoring Competition System Description. Retrieved July* 28:2014.

Seid Muhie Yimam, Iryna Gurevych, Richard Eckart de Castilho, and Chris Biemann. 2013. Webanno: A flexible, web-based and visually supported system for distributed annotations. In *ACL (Conference System Demonstrations)*. pages 1–6.

Jure Zbontar. 2012. Short answer scoring by stacking. *ASAP Short Answer Scoring Competition System Description. Retrieved July* 28:2014.