Soft on errors? The correcting mechanism of a Skolt Sami speller

Trond Trosterud and Sjur Nørstebø Moshagen

UiT The Arctic University of Norway
https://uit.no
{trond.trosterud, sjur.n.moshagen}@uit.no

Abstract. The article discusses correcting spelling errors due to erroneous use of the so-called *soft sign* in Skolt Sami, one of the most common orthographic symbols, and the most common source of typographic errors. The discussion is based upon the suggestion mechanism of an existing open source Skolt Sami speller. The discussion shows that with an improved suggestion mechanism, the speller is able to restore a single soft sign error in over 97 % of the cases, and remove a hypercorrect soft sign as first correction in 90 % of the cases. Allowing the target form to be within top-5, the correction performance is well above 99 %. Improving the suggestion mechanism also had a positive impact of its overall performance, rising the percentage of target forms within top-5 from 74.1 % to 84.7 %.

Keywords: Skolt Sami · Proofing · Spelling · Orthography · Palatalisation.

1 Introduction

The article discusses how the **soft mark**, one of the more common letter symbols in Skolt Sami orthography, is treated by the current Skolt Sami spellchecker[1]. The soft mark indicates palatalisation of its surrounding letters, and thus plays a more abstract role of marking a suprasegmental feature in Skolt Sami orthography. The soft mark is also not part of the orthography of Finnish, the majority language for the Skolt Samis. We thus expect the soft mark to be a challenge for the writers. The present article investigates to what extent the Skolt Sami spellchecker is able to detect palatalisation errors, both omissions of the soft mark and hypercorrect use of it.

[1] This spellchecker was made within the GiellaLT infrastructure for language technology [2]. The source code for the speller engine is available at github.com/divvun/divvunspell. The source code for the Skolt Sami transducer at github.com/giellalt/lang-sms. The principal architect behind the Skolt Sami transducer is Jack Rueter, cf. [1].

In Hämäläinen, M., Partanen, N., Alnajjar, K. (eds.) Multilingual Facilitation (2021), pages 197–207.

2 Background

2.1 Skolt Sami orthography

Linguistically, the Skolt Sami orthography has a one-to-one correspondence between phonemes and graphemes. The consequence of this is a large repertoire of letters (44 letter symbols, as compared to 29 for e.g. Finnish)[2].

Three of the 44 symbols are not letters, but diacritic marks: the **soft mark** ˊ (U+0289), the **hard mark** ʹ (U+02B9) and the **length mark** ˈ (U+02C8). The length mark is only used between identical consonants. It is also only used in dictionaries, not in the standard orthography. The hard mark is used to show lack of palatalisation in contexts where it would be expected.

The soft mark is the most common of the three. In a corpus of 20028 words it was found 7162 times (in 35 % of the words), as opposed to 506 (2,5 % of the words) and 0 for the hard and length mark, respectively[3]. This makes the soft mark the tenth most common letter in Skolt Sami, after **t, i, e, a, l, õ, m, s, u**, and in this article we will focus on the soft mark only.

2.2 The soft mark as a spelling problem

Skolt Sami orthography is phonemic, thus in compliance with what is usually seen as a good orthography, with one letter for each phoneme.

The soft mark differs from the other symbols in two respects: First, it is not a letter, but a diacritical mark. Second, it does not represent a linguistic segment (vowel or consonant), but a **suprasegmental** feature, palatalisation, a feature which is realised over the whole coda of the syllable. The soft mark is put between the last vowel and the first consonant of the coda. Looking at actual data, we find that 1/3 of the typos in our baseline typos corpus shown in table 1 actually are soft mark typos, about evenly distributed between omission and hypercorrect errors (54.1 % vs. 45.9 %) . It is thus clear that the soft mark is an important source of error in Skolt Sami text.

Other potential sources of errors should certainly not be overlooked when building proofing tools for Skolt Sami, but tentative investigations did not reveal problematic cases on the scale of typos connected with the use of the soft mark. A priori one may think that the large inventory of letters would be problematic for Skolt Sami writers, but tests did not reveal obvious classes of e.g. k/ǩ or g/ǧ errors. Another possible error source would be Skolt Sami's quite complex morphophonological alternations, such as consonant gradation and compulsory lengthening. Confusing strong and weak grade or morphophonological short and long vowel alternation does however more often than not result in real-word errors, hence an error type out of scope for the present paper.

[2] The main principles of Skolt Sami orthography was drawn up in [3]. The current orthography is presented in a recent school grammar [4], and phonological aspects of it are discussed in [5].

[3] The corpus used for the present article is a frequency-based sort of Skolt Sami wordforms, at gtsvn.uit.no/langtech/trunk/words/lists/sms/2017-06-22_sms_wf.freq.

We thus confine the discussion of the present paper to errors involving omission and hypercorrect use of the soft mark.

2.3 The speller engines: hfst-ospell vs divvunspell

There are presently two speller engines available to Skolt Sami users, the engines `hfst-ospell` and `divvunspell`. In fact, they are very close to each other, as `divvunspell` is a re-implementation of `hfst-ospell` in Rust. For various technical reasons the Rust implementation is substantially faster. `hfst-ospell` is presently used in Linux and for the LibreOffice speller, `divvunspell` is used elsewhere.

There is another difference between the two that is of interest here: The `divvunspell` engine contains an additional weighting system, where edits of initial and final characters are given higher penalty. As documented in e.g. [11] and [12], the distribution of spelling errors within a word is such that errors in the beginning or at the end of a word is less likely than in the middle of the word. This is captured by the extra penalty system in `divvunspell`, and this affects the overall performance of the speller in terms of suggestion quality. The extra penalty system is presently hard-coded.

Both speller engines use the exact same speller file - there are no differences on the linguistic side. This makes it interesting to see how the difference in the penalty system plays out in various versions of the error model, and below we will present results for both speller engines with regard to correcting errors in the use of the soft mark sign.

For this article we just observe that there is an interesting difference in speller behavior between the two engines, and that there is no clear winner. Improving the suggestion mechanism with respect to the word internal distribution of errors will be a topic for future research.

2.4 Related work

The handling of typographic errors is a vexed topic to follow academically, since the most wide-spread systems are maintained by commercial companies who do not necessarily publish neither methodology nor results. Most of the last years' publications on correcting mistakes have been using neural networks, cf. for example [6].

The main difference between these approaches and ours is that they do not anchor the target forms in an explicit norm. There is no way to say that the target forms represent a **norm** as defined by a normative body. Rather, the language model – morphology and lexicon – is based upon corpora of existing text. Even though the normative body of Skolt Sami and other Sami languages take usage data into consideration, they also state that a large percentage of the wordforms in actual texts are outside of the norm, and want the spellchecker to correct them. In a situation where erroneous forms may outnumber the correct ones, using language models trained on existing texts becomes problematic. It is reasonable to assume that the situation will be similar for most minority and indigenous languages, as text production in these languages is far outnumbered by the majority language production. We also know that text production in general is hard for minority language writers, see e.g. [7], especially chapter 6.

Using neural networks to improve the *correction mechanism* seems more promising. One challenge for this approach is that it needs large amounts of corrected error data, data that often are not there. This is usually solved by creating error data by randomly creating errors (cf. e.g. [8], [9], although they also use real error data). The problem here is the underlying assumption that these random alternations would be representative for actual typos. In our experience, typos are of two types: they either arise from *imperfect mastering of the norm*, or they are due to *accidental typing errors*, hitting the neighbour key or hitting left and right hand keys in the wrong order. Creating typos by random changes will miss both these error types, thereby somewhat undermining the results arising from this methodology.

As shown in the next section, the present article also uses artificially created typos. These forms are not randomly created, though. We do not address accidental typos at all, rather we look at errors due to imperfect mastering of the norm and our procedure for creating the typos is linguistically motivated. This being said we would expect interesting results from machine learning work on correction suggestions in the future.

3 Correcting soft sign errors

3.1 Setting up the test sets

In order to test the speller's treatment of palatalisation, we first built three test sets, one with one omitted soft mark (`nopal`), one with two or more omitted soft marks (`nopal2`) and one with hypercorrect use of soft mark (`hypal`). The test sets were built from all recognised words in the list of 33479 frequency-sorted wordforms referred to in footnote 3, 14108 of which contained the soft mark. We took all words containing one soft mark, and removed it, making the `nopal` test set, and correspondingly for words containing more soft marks for the `nopal2` test set. Making two test sets for soft mark omission was done to be able to evaluate speller behavior independently in the two cases – we expect the speller to perform better for single errors than for multiple ones. For the `hypal` test set, where we wanted to test hypercorrect use of the soft mark, we took mono- and bisyllabic words without a soft mark from the corpus, and added soft mark between the V and C in the coda of the first syllable.

10.35 % of the words with their soft mark removed and 17.25 % of the words with an inserted soft mark resulted in existing words, so-called *real-word errors*. Such errors fall outside of the scope of the present article, as correcting them would require a grammar checker rather than a spellchecker. Since the topic under scrutiny is the suggestion mechanism, the pairs containing real-word errors were removed, so that the `nopal`, `nopal2` and `hypal` test sets[4] were restricted to pairs where the error form was indeed unknown to the speller, and the correct form was known. The task at hand was thus to see to what extent the suggestion mechanism was able to return the intended form, and if so, in what position the expected suggestion would be among all suggestions given.

3.2 Establishing the baseline

In table 1 we show the overall performance of the speller at the start of the development described in this article[5]. There are three speller configurations: For all three configurations, suggestions are ranked according to the Levenshtein distance from the misspelled word ([10]). One configuration added a manually made set of replacement values, favouring certain letter or string substitutions higher than the general Levenshtein distance. The most elaborate configuration gives priority to common wordforms, based upon their frequency weight in a reference corpus, in addition to the manual replacement values. In the tables 1 through 4 these configurations are referred to as *Levenshtein, no corpus and w/corpus*, respectively. Each configuration was tested with both speller engines presented in section 2.3. Overall, the suggestion performance is acceptable, but not very good.

Table 1. Overall performance by February 1. Percentages of suggestions in 1st position, within top-5, anywhere, no suggestions, and only wrong suggestions. N = 324. Tested three variants of the speller: 1) with a corpus for frequency-based weighting of suggestions, and a number of additional replace rules; 2) no corpus, with additional replace rules; and 3) only plain Levenshtein distance 2 edits. Variant 1) was the one available to users February 1.

Suggestion mechanism	1. suggestion	In top-5	In top-10	No suggestions	Only wrong
hfst-ospell w/ corpus	63.43 %	**74.11 %**	74.76 %	4.85 %	20.39 %
hfst-ospell no corpus	61.17 %	73.46 %	74.43 %	4.85 %	20.71 %
hfst-ospell Levenshtein	54.69 %	66.02 %	68.28 %	7.12 %	24.60 %
divvunspell w/ corpus	**64.81 %**	74.07 %	**75.31 %**	4.94 %	19.75 %
divvunspell no corpus	62.65 %	73.77 %	75.00 %	4.94 %	20.06 %
divvunspell Levenshtein	54.94 %	66.98 %	69.14 %	7.10 %	23.77 %

It must be noted that the test sample is very small, only 324 typos. Thus the overall, general performance will not necessarily be as described above on a

[4] The test data used in this work will be made available in the regular github.com/giellalt/lang-sms repository at publication time, in the `test/data/` directory.

[5] The source code can be inspected at the github page for the Skolt Sami speller. The baseline manual replacement rules referred to as *no corpus* in the tables can be found at github.com/giellalt/lang-sms/tree/develop/tools/spellcheckers, the versions for tables 1 through 4 are the ones for February 1st, 2021.

larger set of test data. In comparison, our test bench for the other Sami languages have a general typos list of several thousands, ranging all the way up to more than 10 000 collected, real typos for North Sami.

Still, the test sample does give us an idea of the general performance, and it serves as a good baseline to compare our work against. At the end we will test our reworked suggestion mechanism using the same data, to evaluate what impact our targeted changes have had on the overall performance (cf. table 9 below).

We also tested the baseline speller from table 1 with the palatalisation test data described above. Test results are shown in tables 2, 3 and 4.

Table 2. Same speller as in table 1, but this time using data for testing correction of single soft mark omission errors, the **nopal** test set. N=5599.

Suggestion mechanism	1. suggestion	In top-5	In top-10	No suggestions	Only wrong
hfst-ospell w/ corpus	83.41 %	**87.52 %**	**87.78 %**	2.32 %	9.89 %
hfst-ospell no corpus	81.05 %	86.87 %	87.73 %	2.34 %	9.93 %
hfst-ospell Levenshtein	81.05 %	86.87 %	87.73 %	2.34 %	9.93 %
divvunspell w/ corpus	**85.23 %**	86.78 %	86.78 %	2.89 %	10.32 %
divvunspell no corpus	84.14 %	86.71 %	86.78 %	2.91 %	10.31 %
divvunspell Levenshtein	84.14 %	86.71 %	86.78 %	2.91 %	10.31 %

Single soft mark omission errors all have a Levenshtein distance of 1, and are thus much easier to correct than the general error corpus of table 1, as can be seen from the 20 % increase in the values of table 2. Note that the manual replace rules of the baseline speller have no impact on soft mark omission errors, the performance of the *Levenshtein* and *no corpus* models are the same. Adding corpus weight gives a marginal 1.1 % improvement for 1. position suggestion, and slightly less for the top-5.

Table 3. Same speller as in table 1, but this time using data for testing correction of hypercorrect use of the soft mark, the **hypal** test set. N=945.

Suggestion mechanism	1. suggestion	In top-5	In top-10	No suggestions	Only wrong
hfst-ospell w/ corpus	77.04 %	94.07 %	95.13 %	0.11 %	4.76 %
hfst-ospell no corpus	74.50 %	93.02 %	94.60 %	0.11 %	5.29 %
hfst-ospell Levenshtein	74.50 %	93.02 %	94.60 %	0.11 %	5.29 %
divvunspell w/ corpus	**85.93 %**	**95.13 %**	**95.24 %**	0.11 %	4.66 %
divvunspell no corpus	83.07 %	95.03 %	95.13 %	0.11 %	4.76 %
divvunspell Levenshtein	83.07 %	95.03 %	95.13 %	0.11 %	4.76 %

The picture for correcting hypercorrect use of the soft sign (table 3) is somewhat similar. Also for this corpus, there is an edit distance of 1, and the overall performance is around 20 % above the baseline results. Again, the existing manual replacement rules gave no effect, and adding corpus weights gave an almost 3 % increase in performance for correct 1st suggestion, but almost no effect for the top-5 result.

Table 4. Same speller as in table 1, but this time using data for testing correction of omission of two or more soft mark errors in the same word, the **nopal2** test set. N=1301.

Suggestion mechanism	1. suggestion	In top-5	In top-10	No suggestions	Only wrong
hfst-ospell w/ corpus	**60.49 %**	72.56 %	**73.10 %**	18.52 %	8.38 %
hfst-ospell no corpus	58.57 %	**72.87 %**	**73.10 %**	18.52 %	8.38 %
hfst-ospell Levenshtein	58.57 %	**72.87 %**	**73.10 %**	18.52 %	8.38 %
divvunspell w/ corpus	57.96 %	72.56 %	72.56 %	18.75 %	8.69 %
divvunspell no corpus	57.65 %	72.56 %	72.56 %	18.75 %	8.69 %
divvunspell Levenshtein	57.65 %	72.56 %	72.56 %	18.75 %	8.69 %

We also looked at cases where the target form contained two or more soft signs, i.e. cases with an edit distance of 2 or more. Here, the results dropped below the baseline results for the general test sample. As expected, especially the results for the first suggestion were much worse, since here restoring one soft sign often resulted in a grammatical word, and thus gave a lower Levenshtein distance than the target form, which always would require two editing operations, adding two soft signs. Otherwise, the results was as for the previous tests, also here with no effect from the existing manual rules.

A systematic evaluation of the difference between the two speller engines is outside the scope of the present paper, but we note that `hfst-ospell` performs better when edit distance = 1, whereas `divvunspell` fares better for edit distance = 2.

The next section will discuss the effect of an improved set of edit distance rules. The Levenshtein results, which showed results for a speller without such rules, will thus not be referred to in the next section.

3.3 Test results

In order to improve the speller's treatment of soft mark errors, we thoroughly revised the manual suggestion mechanisms. We improved both the palatalisation rules and to a certain extent also the general rules. For the single-symbol replacement rules, we gave voiced for unvoiced consonant replacement 0.6 % of standard edit distance and unvoiced for voiced as 0.4 %. For the string replacement rules, we added soft mark addition and removal as 0.2 % of standard edit distance for soft mark following diphthongs. For the final string replacement rules we gave the value 0.0 % of standard edit distance for all attested two-consonant codas, and 0.2 % of standard edit distance for the attested three-consonant codas.

We then tested the pairs in our test bench. The results are shown for the same artificial error sets as referred to in the previous section, table 5 (omission of a single soft sign), table 6 (hypercorrect use of the soft sign) and table 7 (omission of two or more soft signs). In all cases, the tables measure the speller's ability to restore the missing or remove the erroneous soft sign. This time the spellers are slightly different: The *+pal, +corp, +letter* setup gives both the revised soft sign rules (*+pal* and the wordform frequency corpus (*+corp*). In order to investigate the effect of the manual replacement rules referring to the soft sign we did a run without the soft sign rules (*-pal*). We also did a run without the corpus weighting (*-corp*) in order to investigate the effect of the frequency of the correct word forms, as well as a run omitting both to compare against. We did not test for omission of the general replacement rules (*-letter*), as this would not be relevant to the present topic.

Table 5. Correcting a **single** missing soft mark. Percentages of suggestion in 1st position, within top-5, the **nopal** test set. N = 5599

Suggestion mechanism	First suggestion	Within top-5	Within top-10
+pal, +corp, +letter (ospell)	94.48 %	**99.36 %**	**99.75 %**
-pal, +corp, +letter (ospell)	94.12 %	99.34 %	**99.75 %**
+pal, -corp, +letter (ospell)	91.55 %	98.79 %	99.70 %
-pal, -corp, +letter (ospell)	90.93 %	98.61 %	99.68 %
+pal, +corp, +letter (divvunsp)	**97.07 %**	98.48 %	98.48 %
-pal, +corp, +letter (divvunsp)	96.64 %	98.48 %	98.48 %
+pal, -corp, +letter (divvunsp)	95.96 %	98.41 %	98.48 %
-pal, -corp, +letter (divvunsp)	95.37 %	98.41 %	98.48 %

Compared to the baseline result, this is an improvement of more than 10 %. Since the weight corpus is the same as for the baseline tests, the main improvement is due to the improvement of the letter rules. The improved soft mark rules give a small improvement for the first suggestion (0.62 % for `hfst-ospell` and 0.57 % for `divvunspell`), for the top-5 results the soft mark rules have only marginal effect. Weighting suggestions for corpus frequency has a slightly larger impact for the first suggestion, 3.79 % for `hfst-ospell` and 1.27 % for `divvunspell`, also here the top-5 and top-10 results were less affected.

We then tested **hypal**, the test set with hypercorrect use of the soft mark. The result is shown in table 6.

Here, the effect of the general improvement rules was marginal, especially for the `hfst-ospell` engine. Also introducing the quite extensive set of soft mark rules actually led to worse results, with a worsening of approximately 1 % for the first suggestion, and a marginal effect for the top-5 result. Introducing corpus weighting did help, with an improvement of 6-7 % for both speller engines, and again the soft mark rules gave worse results.

We also looked at the set of wordforms with two or more soft marks, shown in table 7. In the same way as for the results in table 4, also here the results were poorer than results for word forms with one soft mark missing. Here the soft mark rules had a positive effect, improving the results with approximately 2 %

Table 6. Correcting **hypercorrect** soft mark. Percentages of suggestion in 1st position, within top-5, the **hypal** test set. N = 945

Suggestion mechanism	First suggestion	Within top-5	Within top-10
+pal, +corp, +letter (ospell)	77.88 %	98.41 %	99.68 %
-pal, +corp, +letter (ospell)	79.79 %	98.73 %	99.79 %
+pal, -corp, +letter (ospell)	71.43 %	96.08 %	99.15 %
-pal, -corp, +letter (ospell)	72.38 %	96.40 %	99.15 %
+pal, +corp, +letter (divvunsp)	88.47 %	**99.79 %**	**99.89 %**
-pal, +corp, +letter (divvunsp)	**89.95 %**	**99.79 %**	**99.89 %**
+pal, -corp, +letter (divvunsp)	82.01 %	99.58 %	99.79 %
-pal, -corp, +letter (divvunsp)	82.86 %	99.68 %	99.79 %

compared to a suggestion mechanism without soft sign rules. Systems without corpus weighting performed better than the ones without soft sign rules, but here the best results always were the ones with both cues included.

Table 7. Correcting **multiple** missing soft marks. Percentages of suggestion in 1st position, within top-5, , the **nopal2** test set. N = 1301

Suggestion mechanism	First suggestion	Within top-5	Within top-10
+pal, +corp, +letter (ospell)	**77.86 %**	**92.54 %**	**93.08 %**
-pal, +corp, +letter (ospell)	75.10 %	90.16 %	90.85 %
+pal, -corp, +letter (ospell)	75.48 %	**92.54 %**	**93.08 %**
-pal, -corp, +letter (ospell)	72.71 %	90.16 %	90.85 %
+pal, +corp, +letter (divvunsp)	73.41 %	92.08 %	92.24 %
-pal, +corp, +letter (divvunsp)	70.71 %	89.93 %	90.08 %
+pal, -corp, +letter (divvunsp)	73.10 %	92.08 %	92.24 %
-pal, -corp, +letter (divvunsp)	70.41 %	89.93 %	90.08 %

4 Conclusion: The perfect speller does not exist

The performance of the original suggestion mechanism as measured against the general set of typos was shown in table 1. After the revision of the suggestion mechanism that was made in order to improve the suggestions for soft mark errors, we got the results shown in table 9. The baseline results are also repeated in table 8 for easy comparison.

As shown in table 9, when one throws in other types of errors the performance drops substantially. This is expected, as the additional rules are targeted at soft mark errors. It is also a common issue with spellers: the perfect speller does not exist, it is always a sum of many different compromises and conflicting interests.

It still shows considerable improvement over the baseline, going from about 64 % correct in first position, to 70 % when using a corpus (frequency) weighted lexicon. Compared with a basic Levenshtein model (54.9 %), the improvement

Table 8. Baseline test results, repeated here for convenience.

Suggestion mechanism	1. suggestion	In top-5	In top-10	No suggestions	Only wrong
hfst-ospell w/ corpus	63.43 %	**74.11 %**	74.76 %	4.85 %	20.39 %
hfst-ospell no corpus	61.17 %	73.46 %	74.43 %	4.85 %	20.71 %
hfst-ospell Levenshtein	54.69 %	66.02 %	68.28 %	7.12 %	24.60 %
divvunspell w/ corpus	**64.81 %**	74.07 %	**75.31 %**	4.94 %	19.75 %
divvunspell no corpus	62.65 %	73.77 %	75.00 %	4.94 %	20.06 %
divvunspell Levenshtein	54.94 %	66.98 %	69.14 %	7.10 %	23.77 %

Table 9. Overall performance after our modifications. Percentages of suggestion in first position, within top 5, within top 10, no suggestions, and only wrong suggestions. N = 324

Suggestion mechanism	1. suggestion	In top-5	In top-10	No suggestions	Only wrong
hfst-ospell w/ corpus	**70.33 %**	**84.67 %**	85.33 %	**2.33 %**	12.33 %
hfst-ospell no corpus	68.33 %	83.33 %	**85.67 %**	**2.33 %**	**12.00 %**
divvunspell w/ corpus	69.75 %	83.95 %	85.19 %	2.47 %	12.35 %
divvunspell no corpus	67.90 %	83.95 %	85.19 %	2.47 %	12.35 %

is even larger. If we look at the top five suggestions, we find the correct one in almost 85 % of the cases, about 10 % better than before the changes.

Looking at the artificially generated errors, the speller gives the target form in 97.07 % of the cases for missing soft mark and in 89.95 % for hypercorrect soft mark. Looking at top-5, the target form of both error types is almost always included and the result is marginally better (99.79 %) for hypercorrect soft mark than for omitted soft mark (99.36 %).

Adding lexicon weight was shown to have noticeable effect, especially on the generated soft mark errors. The soft mark rules improved performance on the soft mark omission cases (tables 5, 7), but decreased performance on the hypercorrect forms (table 6). The reason for this was that the soft mark rules were designed to cover omission cases rather than hypercorrect usage. This is something future work on the suggestion mechanism will have to look into.

Overall, the revised suggestion mechanism of the Skolt Sami spellchecker is well capable of handling soft mark errors. We leave it to future work to achieve comparable results also for other types of Skolt Sami typographic errors.

References

1. Rueter, J. and Hämäläinen, M.: FST Morphology for the Endangered Skolt Sami Language, *Proceedings of the 1st Joint SLTU and CCURL Workshop (SLTU-CCURL 2020)*. Beermann, D., Besacier, L., Sakti, S. and Soria, C. (eds.). Paris: European Language Resources Association (ELRA), p. 250-257. (2020)
2. Moshagen, S. N., Rueter, J., Pirinen, T., Trosterud, T., Tyers, F. M.: Open-Source Infrastructures for Collaborative Work on Under-Resourced Languages. In: *Collaboration and Computing for Under-Resourced Languages in the Linked Open Data Era.* LREC. (2014)

3. Korhonen, M., Mosnikoff, J., Sammallahti, P.: *Koltansaamen opas.* Castreanumin toimitteita, Helsinki (1973)
4. Moshnikoff, S., Moshnikoff, J., Koponen, E.: *Koltansaamen koulukielioppi (Sää´mǩiõl ǩiõllvuä´ppes škoou´li vääras)*. Saamelaiskäräjät, Inari. (2009)
5. Feist, T.: *A Grammar of Skolt Saami.* University of Manchester School of Languages, Linguistics and Cultures, Manchester. (2010)
6. Ahmadzade, A., Malekzadeh, S.: Spell Correction for Azerbaijani Language using Deep Neural Networks. arXiv:2102.03218. (2021)
7. Outakoski, H.: Multilingual literacy among young learners of North Sámi: contexts, complexity and writing in Sápmi. *Studier i språk och litteratur från Umeå universitet*, 27. (2015)
8. Jayanthi, S. M., Pruthi, D., Neubig, G.: NeuSpell: A Neural Spelling Correction Toolkit. arXiv:2010.11085v1. (2020)
9. Ghosh, S., Kristensson, P. O.: Neural Networks for Text Correction and Completion in Keyboard Decoding. arXiv:1709.06429v1. (2017)
10. Levenshtein, V. I. = Levenštejn, V. I.: Dvoičnye kody s ispravleniem vypadenij, vstavok i zameščenj simbolov [Binary codes capable of correcting deletions, insertions, and reversals].*Doklady Akademij Nauk SSSR*, 163 (4): 845–848. (1965)
11. Machtynger, Jon and Shallice, Tim: *Normalizing serial position analyses: The Proportional Accountability algorithm.* Cognitive neuropsychology. 26. 217-22, 2009.
12. Alan M. Wing and Alan D. Baddeley: *Righting errors in writing errors: The Wing and Baddeley (1980) spelling error corpus revisited*, in Cognitive Neuropsychology, 26:2, 223-226, 2009.