You can't suggest that?!
Comparisons and improvements of speller error models

Heiki-Jaan Kaalep, Flammie Pirinen, Sjur Nørstebø Moshagen
Tartu ülikool (Kaalep), UiT Norgga árktalaš universitehta (Pirinen, Moshagen)

Abstract

In this article, we study correction of spelling errors, specifically on how the spelling errors are made and how can we model them computationally in order to fix them. The article describes two different approaches to generating spelling correction suggestions for three Uralic languages: Estonian, North Sámi and South Sámi. The first approach of modelling spelling errors is rule-based, where experts write rules that describe the kind of errors that are made, and these are compiled into a finite-state automaton that models the errors. The second is data driven, where we show a machine learning algorithm a list of errors that humans have made, and it creates a neural network that can model the errors. Both approaches require collections of misspelling lists and understanding its contents; therefore, we also describe the actual errors we have seen in detail. We find that while both approaches create error correction systems, with current resources the expert-built systems are still more reliable.

Keywords: Spell-Checking, rule-based, fsa, machine learning, sámi languages, estonian

1. Introduction

The ultimate speller only accepts correct words, finds all spelling errors, and always gives the one and only relevant suggestion. This speller will never exist, but it is the ultimate speller we strive to achieve. In this article we explore a few ideas in that direction, and apply them to three languages found in the *GiellaLT* infrastructure[1]: North Sámi, South Sámi and Estonian. More precisely, this article looks at the error model, and how to improve the suggestions given.

To that end, our goal is to reduce the noise level (increase precision) by generating as few irrelevant suggestions as possible, and when in doubt, give no suggestion at all rather than risk giving irrelevant suggestions; this is in contrast with e.g. Hunspell[2] (Trón et al. (2005)) and the rest of the Xspell family (Ispell, Aspell[3], Myspell, nuspell[4], etc). While pursuing this goal, we try to understand the reasons behind mistyping, and assume that classifying the errors will give us some insight. Having this insight, it might be possible to find ways for increasing recall as well.

An attempt to find regularities in misspellings naturally invokes the idea that one might try machine learning for this purpose; one should use all tools available for achieving one's goal.

The approaches that will be investigated are the following:

- hand-crafted regex error model
- machine-learned error model

The work described in this article says nothing about coverage, i.e. how many words flagged by the speller are real errors and how many are actually correct words, missing from the speller's vocabulary; or how many misspelled words are falsely recognized as correct. We limit ourselves to real misspellings.

The article is organized as follows: first, there is a short overview of earlier work. Following that, we'll describe the methods used for developing new error models. We then describe the misspelling lists used for development, testing and evaluation. After that we say a few words about the types of errors in these lists, followed by a short description of the main features of the languages and their orthography, focusing on the parts relevant to this paper. We then describe the new error models in detail, starting with a short overview of our baseline error model, after which we evaluate the performance of the new error models. Finally, there is a discussion on the outcome, and a conclusion.

[1]https://giellalt.github.io/
[2]https://hunspell.github.io/
[3]http://aspell.net
[4]https://nuspell.github.io

2. Earlier work

A lot of work has been done on spelling corrections—we give an overview of the literature here—although most of it looks at English and closely or typologically related languages. See e.g. Kukich (1992), Hládek et al. (2020). Working with languages with a complex morphology and phonology does offer some additional challenges, and minority and indigenous languages with a recent writing culture adds to that challenge, also, not a lot of work has been done in this area.

Finite-state language models have been used in spell-checking and correction for a while, one of the most recent approaches that is the basis of our system as well is Pirinen et al. (2014). Within the Sámi language context, the work has been done from Gaup et al. (2005) onwards.

Substantial work on analysing North Sámi spelling errors was done in Antonsen (2013), and the insights gained were important for the work done with the North Sámi speller in this article. To the best of our knowledge, no other Sámi languages have been analysed with regard to spelling errors, their classification and frequency.

Estonian spelling errors, that emerge while typing on a computer keyboard, have not been described in publications. However, the Estonian spellers that were created by Filosoft Ltd. in the beginning of the 1990ies (e.g. for Microsoft Word) contain a suggestion module, and since their *C*-language source code has been made public[5], it has been possible to re-implement it as an FST.

There is some prior work done on the general problem of error-correction using neural networks and this is often suggested as the state-of-the-art currently, so we have chosen to experiment on this approach as well. In Li et al. (2020) the authors use a neural model to determine the context of the word, resulting in a better guess as to what was the word that the author wanted to use.

One of our central themes in this article lies in the usage and importance of a public error corpus and/or list; an elaborate model for ordering correction candidates: c.f. Flor et al. (2019). Different sources have different types of errors, thus different strategies should be used, and different recall-precision figures are expected: Beeksma et al. (2018).

The GiellaLT framework (Moshagen et al. 2013) originated from the initial work on proofing tools and morphological analysers for the Sámi languages, where Trond Trosterud has been a major driving force (see e.g. Moshagen and Trosterud (2005) and Trosterud and Wiechetek (2007)). The framework itself is language independent, but favours rule-based technologies suitable for morphology rich, complex, and low-resource languages. The overall goal is to support all language technology needs of indigenous and minority languages, from text input to speech technology. It is constantly being developed, and is the home for keyboards for 50 languages, and language models for more than 130 languages. Many languages and keyboards are in daily use, and is core to the digital life of several indigenous and minority language communities.

3. Methods

In this article we study two approaches to error-correction, a rule-based method using two-level *finite-state transducers* (FST) (Pirinen et al. 2014), and data-driven *neural network-based* (NN) (Hochreiter and Schmidhuber 1997, Bollmann and Søgaard 2016) *language models*. We call a method that corrects incorrect word-forms into correct ones *an error model*.

3.1. FST methods

The finite-state spelling correction follows the model described in Pirinen and Hardwick (2012): a transducer that modifies the erroneous string is composed with the speller transducer, which accepts only valid wordforms. As a result, the suggestion transducer presents only modifications that are also valid wordforms to the user.

Ideally, there would be only one suggestion, and this would be the right one. The more suggestions there are, and the lower down the ranked list the correct one is, the worse for the user; and the worst case is a long list of suggestions without the correct one amongst them. So the suggestion transducer has a dual goal: keep the number of the suggestions low, and rank them correctly. One may ask whether it is better to provide no suggestion at all than to present the correct one ranked as 9th, for example. Presently, we have no answer to this question. What are the psychologically comfortable

[5]`https://github.com/Filosoft/vabamorf`

number and way of ranking, is a question for future research on user studies; presently we just notice that this aspect has to be taken into consideration.

Limiting the number of suggestions can be achieved by either allowing fewer modifications of the erroneous form, limiting the recognizable vocabulary of the speller, or both. As an example: fewer modifications might mean that only edit distance one is allowed, and limited speller vocabulary might mean that only simplex words are allowed, while productively formed compounds are prohibited as suggested corrections[6].

With weighted transducers, we may attach different weights to different edit operations and recognized word-forms. For example, interchanging *d* with *t* adds a certain weight, and every component of a compound word adds another weight. Suggestion ranking will follow from adding up all these weights, and limiting their number may be based on cutting the list either above some absolute weight, or above some absolute number of candidates. However, it is not obvious how one should determine the right final weights and cutting points. This article concentrates on modifications of the erroneous wordforms: what kind of modifications should be made, and whether we can argue for attaching certain weights to these modifications, in order to signal their likelihood.

Weights from the speller lexicon are also used: if two candidates result from modifications with the same weight, then the one which gets smaller weight from the speller is ranked first. We achieve this by having the modification weights surpass the speller weights by a large margin; it is the modification which is important, not the likelihood of the wordform itself. The speller lexicon weights are partly based on frequency of words either in a corpus or by linguistic intuition, and partly on expert-decided likelihood of the morphological tags; more elaborate weighting schemes can be imagined, but that is outside the scope of this article.

3.2. NN methods

For neural error correction modelling, we are using a neural machine translation approach. Within the neural machine translation framework, we use the incorrectly written word-forms as source language, and the corrected word-forms as target language. This logic allows us to train an error correction model with an off-the-shelf neural machine translation toolkit. For this experiment we are using OpenNMT-py[7] (Klein et al. 2017) in its default settings, i.e. a translation model following the OpenNMT tutorial on their website[8].

To limit the creativeness of neural suggestions, we restrict the corrections to word-forms that are acceptable by the dictionary of the rule-based spell-checker. That is, we take the list of n-best translations from OpenNMT-py and check it against the speller lexicon. Only the suggestions accepted by the speller are included in the final suggestion list.

4. Lists of misspellings

It is a truism that texts differ, depending on who creates them, for what purpose and for what readership. Likewise, it is only natural to expect that the errors made while writing depend on various factors. We are aware that the misspelling lists we have at hand are not representative of the "general text class" created by an "average writer"; so, in order to remain cautious when interpreting our results, here are the main characteristics of the corpora that these lists are derived from.

4.1. North Sámi

The present day North Sámi orthography is from 1979, with some smaller adjustments from 1985[9]. The present orthography is thoroughly described in Nickel and Sammallahti (2011).

As a result of the Norwegian assimilation policy towards the Sámi people throughout a major part of the 20th century, it is clear that most texts written in the modern orthography are pretty recent. Modern North Sámi literacy is correspondingly young, which is reflected in texts in the form of spelling and other grammatical errors. In the material

[6]They would still be accepted by the speller. The core idea is that one can use two different transducers or automata for the speller: one to verify the text, including productive morphology, and another, more restricted transducer, to verify suggestions.

[7]`https://opennmt.net/OpenNMT-py`

[8]`https://opennmt.net/OpenNMT-py/quickstart.html`

[9]There have been several older orthographies going back all the way to 1748.

used in Antonsen (2013) there is about 4% spelling errors, which is considerably more than in e.g. Norwegian or English texts produced by native speakers. In Flor et al. (2015), where the majority of the texts are written by non-native speakers of English at various levels of mastering the language, the average number of spelling errors is 2.74%. And for the most advanced writers contributing to the data set, the average number of misspellings is well below 1%. That is, the average number of spelling errors in North Sámi texts is considerably higher than in similar English texts. This is expected given the short history of the orthography, the sociolinguistic setting, the paucity of available text and thus written language exposure, and the minority language status of North Sámi.

The material used in developing, testing and evaluating the error models in this paper has been collected over many years while developing various language technology tools for North Sámi.[10] Misspellings found in texts have been collected in a separate text file, together with the expected correction (usually based on the incorrect word form itself, sometimes also considering the context where the misspelling was found). By the time of writing, the list of typos contains 11 706 entries. Since the focus of research described here is evaluating and developing error models, the list was filtered by removing multiword expressions, false negatives[11], and entries for which the given correction was not recognized by the speller. The filtered list consists of 10 745 entries.

Given the development history of the list of typos, the source texts for the misspellings can be assumed to be all sorts of texts, the majority of which are found in SIKOR[12]. That is, the collection of typos can be considered relatively representative of errors made by North Sámi writers of various genres.

For the machine learning experiment, the list was split in three according to the usual 80-10-10: 80% for training, and 10% each for testing and development / validation. For the regular expression experiment, no such split was used, and the list was both used to inform the developers about useful patterns, and to evaluate the resulting error model.

4.2. South Sámi

The present day South Sámi orthography was formally decided upon in 1978, although Bull and Bergsland (1974) used an early version of that orthography. South Sámi differs from most other Sámi languages and dialects due to a vast and complex system of umlaut, c.f. Bergsland (1994) and Magga and Magga (2012). Although South Sámi does not have consonant gradation as opposed to the other Sámi languages, it does have alternations in consonant clusters and surrounding vowels depending on the syllable and foot structure of the word. Various inflectional endings add zero, one or more syllables to the base form, which forces a recast of the foot structure, which can set off a chain reaction of various consonant and vowel changes. Two examples:

(1)	a.	gåetie	gåatan	gåatetje	gåatatjasse
		gåetie+N+Sg+Nom	*gåetie+N+Sg+Ill*	*gåetie+Dimin+N+Sg+Nom*	*gåetie+Dimin+N+Sg+Ill*
		‘House, into the house, little house, into the little house’			
	b.	åeruve	åerievasse	åerievadtje	åerievadtjese
		åeruve+N+Sg+Nom	*åeruve+N+Sg+Ill*	*åeruve+Dimin+N+Sg+Nom*	*åeruve+Dimin+N+Sg+Ill*
		‘Squirrel, into the squirrel, little squirrel, into the little squirrel’			

That is, the vowel of the second and third syllables changes as follows: *-ie-, -a-, -e-, -a-* for *gåetie*, and *-u-* + *-e-, -ie-* + *-a-* for *åeruve*. The default illative case ending has two forms: *-asse* and *-ese*, and the diminutive derivation also has two forms: *-etje* and *-adtje*. The form of the suffixes (illative and diminutive in example 1) are solely dependent on the syllable count, whereas some vowel changes also depend on the stem type. The umlaut of the root vowel is triggered by the underlying vowel of both case and derivational suffixes.

The South Sámi language community is just a fraction of the North Sámi, and with correspondingly less production and exposure to the written language. Also, a considerable portion of the population is in practice L2 speakers. This is reflected in the misspelling list used for testing as a number of errors relating to mixing vowel and inflectional endings, essentially miscounting the syllables and thus applying the wrong suffix; an example of this taken from the list can be seen in (2). (2) also contains other errors, like using *ø* for correct *ö*, and mixing *s* and *sj*. Identifying each and every such case reliably is not trivial, identifying the proportion of these errors to the rest is left as a topic for future research.

[10]Source code at: `https://github.com/giellalt/lang-sme`
[11]misspellings accepted by the speller as valid words.
[12]Giellatekno and Divvun (2021)

(2) a. *Vyøhkesadtibie
vyöhkesjadtedh+V+IV+Ind+Prs+Pl1
'We help each other' (wrong syllabification and thus suffix form)

b. Vyöhkesjadtebe
vyöhkesjadtedh+V+IV+Ind+Prs+Pl1
'We help each other' (correct syllabification and suffix)

Identifying the syllabic structure is not made easier by historic processes leading to exceptions, so that instead of the regular pattern $2 + 2 + \cdots n \cdots + 2/3$, you get $3 + 2$, or $2 + 1$, instead of the expected $2 + 3$, and 3. Examples of these can be seen in (3).

(3) a. dåerie•dieh
dåeriedidh+V+TV+Ind+Prs+Pl3
'They are following' (syllable structure: 2 + 1)

b. dåerede•minie
dåeriedidh+V+TV+Ger
'(In the process of) following' (syllable structure: 3 + 2)

Complicating the issue further are loan words: how should their syllables be counted and fit into the foot structure of South Sámi phonotactics? An example of this can be seen in (4), with the misspelled form in (4a), and the correct form in (4b). It is very clear that the misspelling of the case suffix is caused by applying a wrong foot structure to the word form.

(4) a. Wikipe•dij:ese
wikipedije+N+Sg+Ill
'Into Wikipedia' (wrong syllable structure: 3 + 3, and thus wrong suffix form)

b. Wiki•pedi•jasse
wikipedije+N+Sg+Ill
'Into Wikipedia' (Correct syllable structure: 2 + 2 + 2)

Finally, the South Sámi orthographic rules recommend that one uses Norwegian *æ* and Swedish *ö*. Up until recently, following these rules require that one knows how to produce the vowel letter from the other side of the border, and it also requires an extra key press: AltGr + the standard vowel. In practice, most people didn't care, and the South Sámi list is full of Norwegian *ø*'s and Swedish *ä*'s. These are considered misspellings by the spelling checker, and they also contribute to the complexity of correcting South Sámi. It is not uncommon to find spelling errors with an editing distance of four and more; in the test list of typos 48 such cases are found, ≈4.2% of the corpus.

As was the case with North Sámi, the list of typos for South Sámi is collected while developing the morphological analyser, based on material that is mostly found in SIKOR (Giellatekno and Divvun (2021)). The cleaned version of that manually built list mentioned above contains only 1 154 entries. A separate list of typo-correction pairs was extracted from a manually marked up corpus of gold-standard text. That token list contains 8 325 non-unique entries, and was used for training a machine learning model, testing and evaluation, using the common 80-10-10 split. This list, extracted from the gold standard corpus, was not used when building the manually crafted regex error model.

4.3. Estonian

Estonian orthography in its present form was adopted during the third quarter of the 19th century. It is modelled after Finnish orthography; the proposal was made by Adolf Ivar Arwidsson (1822). Prior to this, Estonian orthography was modelled after High German, but uneducated Estonian peasants spontaneously tended towards the Finnish style orthography (Kask 1970:p. 204)

The main difference from the previous orthography lies in the simplicity of the rules for marking phone length: nowadays, the rule of thumb is that a short phone is marked by one letter, a long (and extra-long) phone by two letters, and every consonant in a cluster is marked with one letter, even if it is pronounced long or extra long. As an exception, *k*, *p* and *t* are written as *g*, *b*, *d* when short, *k*, *p*, *t* when long, and *kk*, *pp*, *tt* when extra long. Also, when adjacent to a nonsonorous consonant, *g*, *b*, *d* are also written as *k*, *p*, *t*. In addition to indeterminacy in differentiating between long and extra long phones (except for *k*, *p*, *t*), and between short and long ones in consonant clusters, palatalisation is also not marked. There have been numerous propositions to improve the Estonian orthography, in order to make it even more phonetic, e.g. by allowing double letters in consonant clusters, and three letters for extra long phones, but these propositions have not been adopted. Very succinct hearing and marking of phone lengths is difficult to implement in practice, given the various co-articulation effects in real speech.

In addition to the principle of phone length and letter correspondence, the Estonian orthography also to some extent follows the principle of keeping the traditional form of words (even if it deviates from the current pronunciation), and the principle of retaining the form of morphemes while inflecting the word (Erelt et al. 2007). Orthography errors tend to happen when these two additional principles collide with the phonemic principle.

The Estonian list of 3000 misspelled words originates from journalists' texts. About one third of it dates from the 1980-1990ies: 1) a re-typed-in Corpus of Estonian Literary Language[13], containing 1 million words from 1983–1988, and 2) texts from the news agency Baltic News Service, from one month in 1996 (about 250 000 words). The errors were gathered by running an Estonian morphological analyser on the corpus; and then manually picking misspellings from the set of unanalysed words (by Heili Orav and Leho Paldre). Another two thirds date from 2000-2010ies, gathered by Kairit Sirts from a newspaper corpus in an ad hoc manner, according to her own words.

5. Error types

An ideal error typology would reflect what went wrong in the chain of actions of the writer, and/or what was the likely cause, not just count the edit operations. However, we have not been able to reach this ideal yet. It seems though that one potential distractor might be the current set of conventions for writing the language, i.e. its orthography.

The full list of registered typos was run through a semi-automatic classification system, and tagged according to identified class. The resulting classification combines edit distance with character classes that are involved and is summarized in Table 1. In cases where subclasses are identified, the figures for those are listed to the left in each column, the total to the right.

Accented letter errors are easy to correct: there are very few alternatives one should offer, and the reasoning behind the suggestions is transparent, making it easy for the writer to decide whether to accept or not. An example for Estonian would be **tshempion* - *tšempion*. For North Sámi, this type of errors is very frequent - one third of misspellings belong to this class, and we can even identify subclasses: vowel *á* vs *a* (e.g. **Amerihka* - *Amerihká*), or consonants *č*, *đ*, *ŋ*, *š*, *ŧ*, *ž* vs *c*, *d*, *n*, *s*, *t*, *z* (e.g. **Cuovvovaccat* - *Čuovvovaččat*, **Sámediggerádi* - *Sámediggeráđi*, **CD-singel* - *CD-siŋgel*, **oktašas* - *oktasaš*, **olbmoŧ* - *olbmot*, **gazaldaga* - *gažaldaga*). In fact, *a*-*á* confusion is the single most frequent spelling error in North Sámi texts, around 40% in general according to (Antonsen 2013:p 24)[14]. The source of these errors in North Sámi is likely several. One is lack of keyboard support that makes it hard to type the correct letter. That was a major issue in social media texts investigated by Antonsen op.cit., but for several years now there has been available a North Sámi keyboard app for mobile phones, so this is less of a problem today. Another possible source is insecurity in the correct spelling, often in combination with dialectal variation. The *a*-*á* confusion can at least partly be attributed to the fact that the orthography does not follow the phonology in various dialects, the variation is greater and more complex than the orthography reflects. Also final *ŧ* instead of final *t* is most likely based on pronunciation: in some dialects, the plosive *t* is reduced to a pure fricative *h* sound when followed by a word beginning with a vowel. As almost all misspellings of *ŧ* for correct *t* can be found in this position, it is very likely that phonology plays a role. For a more detailed analysis of spelling errors in North Sámi, see (Antonsen 2013).

Accented letters in South Sámi covers only three pairs: *i* vs *ï* (e.g. **jih* - *jïh*, **hïjven* - *hijven*), **ø* vs *ö* (e.g. **bøøremes* - *bööremes*), and **ä* vs *æ* (e.g. **nännoste* - *nænnoste*). But they cover more than half of all misspellings in our test data. Out of a total set of 8 325 misspelling instances, 4 285 – or 51.5% – are errors of this type. The conjuction *jïh* (=*and*) alone counts for more than 10% (884 occurrences) of all misspellings. The three pairs fall into two categories, one purely orthographic, and one phonological. The **ø/ö* and **ä/æ* pairs are purely orthographic: as South Sámi is spoken in both Sweden and Norway, the idea is to make a compromise such that one sound is written using a Swedish letter (*ö*) and one using a Norwegian letter *æ*. Due to the lack of a South Sámi keyboard, people have usually fallen back to using either a Norwegian or a Swedish keyboard, disregarding the orthographic norm. In the case

[13] `https://www.cl.ut.ee`

[14] she includes real-word errors, which we do not, which probably explains the difference in relative size for this error type in her investigation compared to our findings.

Main error class	Subclass	Estonian	North S	South S
	á vs *a*		25	
	čđŋšŧž vs *cdnstz*		8	
Only accented letter errors		2	= 33	5
	double or diphthong	7	13	
	other	37	7	
Delete 1		= 44	= 20	18
	double or diphthong	7	11	
	other	16	4	
Add 1		= 23	= 15	14
Substitute 1		13	17	11
	1 to 2 or 2 to 1		3	7
	adjacent		2	4
Substitute 2		0	= 5	= 11
Transposition		10	2	2
Repetition; South S=suffix		2	0	3
Other		6	8	36
Total		100%	100%	100%

Table 1: Error types, percentage of all errors.

of *i* vs *ï* it is a real phonological opposition, although the distinction was not made in early versions of the South Sámi orthography. The distinction is also not clear to all speakers.

As seen above, the error type **accented letters** is a heterogenous class, with various properties across the languages. It still makes sense to treat them as one with respect to modelling errors, as they stand out from other misspellings both in frequency and often simplicity of correction.

Deleting (or omitting) a letter is a very frequent error. It may be caused by failing to hit a key, or by failing a phone-to-letter mapping rule. A suggestion to correct this error by doubling a letter, or (in case of North Sámi) by creating a diphthong, e.g. **departementa* - *departemeanta*, might seem more plausible than a suggestion to insert a letter in some random position of the same word. Thus, it makes sense to identify this subclass of deletions.

If the misspelling means that an extra letter has been **added**, we also identify a subclass of resulting doubles or diphthongs, the classification thus being similar to the deletion errors.

Substitution errors are relatively more frequent in the Sámi corpora than in Estonian. They also involve cases where one letter is substituted by two (e.g. North Sámi **direktora* - *direktevra*), or two by one (e.g. North Sámi **Osllu* - *Oslo*), or two adjacent letters by two different ones, as in consonant gradation mix-ups (e.g. **Sámedikkeválgii* - *Sámediggeválgii*).

In Estonian, the main source of errors is the typing process, as evidenced by the relatively high proportion of **transpositions** (e.g. **komapnii* - *kompanii*) and repetitions (e.g. **poliititika* - *poliitika*). Errors relating to incorrectly writing phones are relatively few. In North Sámi, the main source of errors is the phone-to-letter process, i.e. applying rules of orthography. Many substitution errors may be blamed on it. This is also documented and discussed by (Antonsen 2013).

In South Sámi as well, the main source of errors is the phone-to-letter process, i.e. applying rules of orthography. In addition, another major source of error is the morphophonology of the language, especially as related to syllable structure and its consequences for **suffix** realisation, as exemplified by **edtjibie* vs *edtjebe*. But the biggest class of errors in South Sámi is the unclassified **other** group — these are typos that are not easily classified by the means used in this work.

6. Error models

The error models we study are: the baseline, a new regex model, and a machine learned model. The baseline model is a general edit distance 2 model built from the alphabet of the language, with some language-specific tweaks described below, whereas the regex model focuses on documented and generalisable error types for the language in question.

6.1. Baseline error models for South and North Sámi

The baseline error models for North and South Sámi are the ones used in production[15]. They are both built following the same structure, and as such the models will be described only once. A general description of the production error model can be found online[16].

The starting point is a Levenshtein edit distance (Levenshtein et al. 1966) error model based on the alphabets of the language, with an editing distance of two. It is possible to adjust the weight of specific edits in the edit distance 2 error model. Adjacent swaps are not enabled by default (they are computationally quite expensive in the present implementation).

Parallel to the default Levenshtein error model, there is a separate set of string edits, handwritten based on identified and frequent error patterns in the languages. The string edits are single FST operations, although each string can be arbitrarily long, thus allowing for much more complex edits than the default model. The string edits are applied as many timed as the default error model, that is, up to twice for both North and South Sámi.

Another extension to the default model is one of suffix edits. That is, a simple transducer mapping input strings to output strings, as the string edits described above, but now restricted to the end of the word. As described above, errors in suffixes are relatively common in especially South Sámi, and this module is meant to target such errors.

Finally, there is a whole-word string replacement module, but that one is utilized very rarely, and does not impact the performance very much. It is also applied to the new regex models described below, mainly because it would be more work to avoid using it.

For Estonian, the regex model is the first one implemented in FST. It is based on the earlier work by Filosoft; no earlier baseline models have been developed for Estonian.

6.2. Rule-based error models

The *regular expressions* (regexes) are grouped according to our assumptions about the nature and likelihood of different types of spelling errors. Also, although guided by the principle that when ranking, one should prefer suggestions with fewer modifications, ours is not based directly on Levenshtein distance. The reasoning is that when calculating the amount of difference between two words, one should view them not as mere symbol strings, but as the traces of a series of mental and physical actions. A change in one action may result in multiple changes in the letter sequence, but it should still be counted as one error.

- Keyboard and orthography (mis)matches. In addition to the Latin letters that form the core of the alphabet, languages typically need some (usually accented) modifications of some of these letters, corresponding to the phones not covered by the core alphabet. These accented letters tend to be positioned in the periphery of the standard keyboard, and/or need key combinations to be used for appearing in the text. It is to be expected that such letters also tend to be mistyped. Also, an accent on a letter may indicate a minor pronunciation subtlety which the speakers need not pay much attention to, so mixing similarly looking and sounding letters would be easy.

  For Estonian, the misspelling list indicates that in case the keyboard does not provide a convenient way to type the accented letters, users may come up with an alternative orthography, e.g. use *sh* or *s^* instead of the correct *š*. If this is the case, then one may expect unlimited substitutions of this kind in a wordform (in addition to other errors). Nordic letters that are not part of the Sámi alphabets, and *á* which is notoriously difficult for North Sámi writers to use correctly, also belong to this class of errors. Correcting them is weighted lightly, and the number of such edit operations is not limited.

[15] https://divvun.no
[16] https://giellalt.uit.no/proof/TheSpellerErrorModel.html

- Keyboard errors, like transposition of letters and repetition of letter sequences, happen so likely in Estonian that regexes for them are needed, while in Sámi, they are highly unlikely. Encoding a context-dependent regex (like one that is needed for repetition) is very costly in terms of FST memory, thus they are not used in the Sámi FSTs.
- Morphology errors, i.e. violating the rules that govern how a word is modified when it is inflected or compounded. These errors are corrected by highly specialised regexes containing string pairs, e.g. a pair of inflectional suffixes.
- Orthography, i.e. the convention of writing phones and their combinations. Letters and combinations that sound similar, like *i* and *j*, belong to this group. For Estonian, the set of orthography-related regexes is smaller than for the Sámi languages, reflecting the proportion of this type of errors in the misspelling list. Also, it is rather common for a Sámi word to contain more than one orthography error (as defined currently); it is possible that a better understanding of the errors will allow us to see in the future how they might really be the manifestation of single errors in the mental process of the writer.

There are different ways to write and combine regexes to yield an FST that converts an input string into another. It is common knowledge among programmers that every existing program can be turned into one that either 1) is smaller when compiled, 2) runs faster, or 3) is more readable, but it is not possible to achieve all these three goals simultaneously. The same is true for FST's. It is well known that an obvious and simple (for the human eye) set of regular expressions may well result in a huge transducer. Aiming at a smaller transducer, one must note that as a rule of thumb, a simpler and smaller transducer puts fewer restrictions on the language it accepts, in other words, the set of possible string pairs passing through a simpler typo modification transducer is larger, thus resulting in more time the speller FST has to spend checking them. Consequently, the number of possible modifications must be controlled, and this forces one to either complicate the regexes or allow the transducer to grow in size.

Appendix A presents a selected set of regex examples showing solutions to some specific problems.

6.3. Machine learned error models

Error modelling in the neural framework is based on imagining the problem as a question similar to machine translation, or just a sequence to sequence character string mapping. Instead of e.g. learning a mapping of e.g. English to French we make the model learn the mapping of misspelled to correct word-forms, and instead of a sentence of words as a context, we have the letters in a word-form. The idea is that if we have enough such mappings, the neural model will learn to translate the misspelled strings into correctly spelt ones, as long as the word list is representative of the errors that are being made. The error correction models that are learnt are character-based, so in principle a representative sample should have some examples of each substitution, deletion and insertion in various enough contexts, so it will learn to make them exactly in the places needed. As is usual with machine learning, the modelling is data-hungry, which means that for ideal usable models we need hundreds of thousands of examples, something that we cannot easily deliver with a low-resource languages. However, in recent years the requirement of the amount of data has been getting smaller, which has made it more plausible to perform these experiments in real low-resource settings.

7. Evaluation

The data in Table 2 gives an overview of the performance of the various error models, for a number of parameters:

- **Spelling error list size:** Number of spelling errors in test corpus for rule-based model, number of training samples / validation + testing for neural network.
- **Average position of correct suggestion:** Ideally this should be 1, ie the correct suggestion is always on top.
- **Average number of suggestions per misspelling:** Ideally this should also be 1, ie there should be no other suggestions than the correct one. That is, the higher the number, the higher the noise level.
- **Top 1/5/all positions:** How many of the misspellings have a correct suggestion in the top position, among the top 5 suggestions, or anywhere among the suggestions
- **No suggestion:** How many of the misspellings have no suggestions; neural models will generally always generate suggestions
- **Only bad suggestions:** How many of the misspellings get only wrong suggestions?

	Estonian		North Sámi			South Sámi		
	RGX	ML	BL	RGX	ML	BL	RGX	ML
Spelling error list size, in thousands	3.0	2.4/0.6		8.5	10.0/1.1		1.1	6.6/0.8
Average position of correct suggestion	1.31	1.97	1.33	1.36	1.99	1.45	1.37	1.49
Average number of suggestions per typo	5.47	6.0	4.00	7.80	6.30	9.30	7.43	6.06
Top 1 positions, %	76.81	13.35	65.03	75.92	46.64	71.32	69.06	34.32
Top 5 positions, %	93.71	28.01	77.53	89.68	63.82	89.43	84.23	46.26
All positions, %	94.46	28.01	78.55	91.30	64.62	91.16	86.05	47.01
No suggestion, %	1.94	0	8.99	2.09	0	1.04	3.55	0
Only bad suggestions, %	3.60	71.98	12.46	6.60	35.38	7.80	10.40	52.99
Speed, words/second	11.05	85.51	31.77	69.34	17.09	14.12	35.69	38.66
FST/ NMT error model size, Mb	13	38	30	31	38	7.9	17	38

Table 2: FST / machine learnt performance; BL = baseline, RGX = handmade regex, ML = machine learnt. ML spelling error list size is specified as training data size / test & evaluation data size.

- **Speed, words/second:** This number is relative, and is provided only to compare between the models and languages. The speed tests were run on the same computer, with as similar conditions as possible; the neural models used a single GPU core and the FST a single CPU core[17]
- **Error model size in megabytes:** FST size is provided in the table to compare the models. The FST size is directly proportional to the use of regexes that consider longer context, like when checking letter pair, triple etc. repetitions, or counting the allowed number of edit operations. The neural model size is the size of the neural network and dependent on the hyperparameters used.

The performance numbers are not directly comparable between languages, due to the different nature of source texts of the misspellings: the orthographic conventions, text creation agents (fast-typing journalists vs a heterogeneous group of Sámi writers), and age of literacy and literary traditions. Keeping these differences in mind, there are still interesting observations to be made.

First, the machine learning models are not able to compete with any of the FST models, not the baseline model, and not the handwritten regex error models. North Sámi had the largest training material, a bit over 8000 typo-correction pairs used for training, and that is clearly not enough to achieve a useful error model. Having established that as a fact is very helpful when guiding future work on minority and indigenous languages. This provides strong evidence that the GiellaLT philosophy is correct: for languages with little to none electronic resources, rule-based is the only option, and we clearly establish that this includes the error models in spelling checkers. If machine learning methods are not currently viable for North Sámi, the language in this experiment with the largest resources, then clearly it will not work for lesser resourced languages.

Upon closer inspection of the ML model, it also became clear that it was mostly simple edit distance one errors that got correct suggestions, which means that it can't contribute meaningfully in a hybrid setup either — the errors it can correct are errors that the rule based model has no problem correcting.

Second, building hand-tuned regular expressions is an exercise worth undertaking, but requires thorough analysis of the error landscape. The Estonian error model performs really well, and the North Sámi handwritten regex model (labelled RGX in Table 2) makes a major leap forward in both recall and precision, making 10+ percentage jumps compared to the baseline. Admittedly, the baseline error model was not that good in the first place, but with the handwritten regex, North Sámi is close behind the Estonian model in performance.

South Sámi, on the other hand, had a good baseline error model to begin with, and with the most complex and varying error typology, it was harder to write a regex that would improve upon the baseline. The regex model is still not very far behind, and with some more analysis and fine-tuning it should be possible to surpass the baseline. The South Sámi spelling error lists have a large portion of errors due to pure orthographic conventions, mixing **ø* for *ö*, and to a

[17]an intel Core i7 CPU and an nVidia Quadro T1000

less degree **ä* for *æ*. Since these errors are frequent, and at the same time very easy to correct, any speller would be expected to perform relatively well. To really improve the South Sámi speller one would have to focus on the remaining classes of misspellings, but as shown above, those are quite heterogeneous. And finally, the misspelling data available to regex development and testing was quite small, and that may influence the numbers. We used another data source for ML training and testing, hence the very different dataset sizes for ML and RGX.

The reasons for the difference between the North and South Sámi baseline models can probably be attributed to several factors. One possible factor is the differences in orthographic principles, where North Sámi often uses accented letters and is thus increasing the size of the alphabet, where South Sámi uses consonant clusters to express the same sound. Another factor is the very different morphophonologies: North Sámi has a very rich and complex consonant gradation system, South Sámi does not have consonant gradation, and South Sámi on the other hand has an elaborate umlaut system, where North Sámi has a very limited set of vowel alternations. But these are just educated guesses on what the cause of the difference could be. A more thorough explanation would require a separate study, and is outside the scope of this article.

We did not use a list of unseen misspellings when evaluating the BL and RGX models. We are aware that when writing regexes, it is possible to come up with specific ones for a selected set of words, i.e. end up with overfitting to the data. We tried to avoid this, and interested readers might check the source code[18]. However, in case of a corpus with different parameters than we had (e.g. a different type of text, different types of text producers), we expect the precision figures to be different.

While having a real held out error corpus for evaluation would be ideal, it is not easily attainable in the context of lesser resourced languages, where we often want to use all available data during development phase for practical reasons.

8. Discussions and speculations, conclusion

Given that for all three languages, one can take an FST speller, baseline or new RGX, that has a recall of over 90%, the main remaining task is to improve precision. That can be achieved in two ways: by even more targeted hand-crafted regexes, or by looking at the context of the error word to either filter or rerank the suggestions. Handcrafting could target the hypothesis that some co-occurring errors are actually inter-dependent (and the co-occurring problem of long-distance dependencies), and/or fine tune the weights and thus the ordering of the suggestions. Since most speller API's do not give any context information, one should try to improve the ordering independently of the context. This is an area for future research.

On the other hand, if the context is available, e.g. via a grammar checker API, it should be possible to filter or promote specific suggestions based on the syntactic context. One such example is Pirinen et al. (2012), using a POS trigram model to promote suggestions matching the trigram model. Another approach is to use the full sentence as the context, in combination with syntactic disambiguation and parsing, for example using the VislCG3 formalism. Examples of such systems are Bick (2006) and Wiechetek et al. (2019), the last one being implemented within the GiellaLT framework, and as such principally available to all languages. That would thus be the next logical step.

For the machine learning setup, it seems that the limitation posed by the amount of training data we have available is still too dire for our use case. While this can be improved by approaches such as automated generation of synthetic misspelling lists, it is limited by the fact that the error generation algorithm should be representative of the errors that real users are likely to make; merely generating statistical noise using a Levenshtein style algorithm will only lead to a neural model that is equal to rule-based Levenshtein corrector but heavier. On the other hand, if we know enough of the nature of the real-world errors to devise an algorithm to generate a representative synthetic misspelling list, we already have an algorithm that can also solve those errors, perhaps more efficiently than the network, thus the main positive side of the neural model may be in the added robustness. All in all, it is good to know that chasing machine learning ghosts can now be put aside, as we have demonstrated that there are better alternatives with a lower environmental impact (see Strubell et al. (2019) for an evaluation of the environmental impact of modern machine learning).

[18]`https://github.com/giellalt/lang-sma/`, `https://github.com/giellalt/lang-sme/`, `https://github.com/giellalt/lang-est-x-utee/`

Acknowledgments

The work was partly supported by the Centre of Excellence in Estonian Studies (CEES, TK-145). The computations for ML model building were performed on resources provided by UNINETT Sigma2 - the National Infrastructure for High Performance Computing and Data Storage in Norway.

References

Antonsen, Lene. 2013. Čállinmeattáhusaid guorran [english summary: Tracking misspellings]. *Sámi diedalaš áigecála* 2/2013: 7–32.

Arwidsson, Adolf Ivar. 1822. Ueber die ehstniche orthographie. won einem finnländer. *Beiträge zur genauern Kenntniss der ehstnischen Sprache. Funfzehntes Heft* pp. 124–130.

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

Beesley, Kenneth R and Lauri Karttunen. 2003. Finite-state morphology: Xerox tools and techniques. *CSLI, Stanford* .

Bergsland, Knut. 1994. *Sydsamisk grammatikk*. Davvi Girji o. s., Karasjok.

Bick, Eckhard. 2006. A constraint grammar based spellchecker for danish with a special focus on dyslexics.

Bollmann, Marcel and Anders Søgaard. 2016. Improving historical spelling normalization with bi-directional LSTMs and multi-task learning. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 131–139. The COLING 2016 Organizing Committee, Osaka, Japan.

Bull, Ella Holm and Knut Bergsland. 1974. *Lohkede saemien. Sørsamisk lesebok*. Grunnskolerådet, Kirke- og undervisningsdepartementet: Universitetsforlaget, Oslo.

Erelt, Mati, Tiiu Erelt, and Kristiina Ross. 2007. *Eesti keele käsiraamat*. EKI, Tallinn.

Flor, Michael, Michael Fried, and Alla Rozovskaya. 2019. A benchmark corpus of english misspellings and a minimally-supervised model for spelling correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 76–86. https://doi.org/10.18653/v1/W19-4407.

Flor, Michael, Yoko Futagi, Melissa Lopez, and Matthew Mulholland. 2015. Patterns of misspellings in L2 and L1 English: a view from the ETS Spelling Corpus. *Bergen Language and Linguistics Studies* 6. https://doi.org/10.15845/bells.v6i0.811.

Gaup, Børre, Sjur Moshagen, Thomas Omma, Maaren Palismaa, Tomi Pieski, and Trond Trosterud. 2005. From xerox to aspell: A first prototype of a north sámi speller based on twol technology. In *International Workshop on Finite-State Methods and Natural Language Processing*, pp. 306–307. Springer. https://doi.org/10.1007/11780885_37.

Giellatekno and Divvun. 2021. SIKOR UiT Norges arktiske universitets og det norske Sametingets samiske tekstsamling, versjon 01.10.2021.

Hládek, Daniel, Ján Staš, and Matúš Pleva. 2020. Survey of automatic spelling correction. *Electronics* 9 10: 1670. https://doi.org/10.3390/electronics9101670.

Hochreiter, Sepp and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9 8: 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Kask, Arnold. 1970. *Eesti kirjakeele ajaloost*. Tartu Riiklik Ülikool, Tartu.

Klein, Guillaume, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pp. 67–72. Association for Computational Linguistics, Vancouver, Canada. https://doi.org/10.18653/v1/P17-4012.

Kukich, Karen. 1992. Techniques for automatically correcting words in text. *Acm Computing Surveys (CSUR)* 24 4: 377–439. https://doi.org/10.1145/146370.146380.

Levenshtein, Vladimir I et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, vol. 10, pp. 707–710. Soviet Union.

Li, Xiangci, Hairong Liu, and Liang Huang. 2020. Context-aware stand-alone neural spelling correction. *arXiv preprint arXiv:2011.06642* https://doi.org/10.18653/v1/2020.findings-emnlp.37.

Magga, Ole Henrik and Lajla Mattsson Magga. 2012. *Sørsamisk grammatikk*. Davvi Girji, Karasjok.

Moshagen, Sjur, Tommi A Pirinen, and Trond Trosterud. 2013. Building an open-source development infrastructure for language technology projects. In *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013)*, pp. 343–352.

Moshagen, Sjur N. and Trond Trosterud. 2005. Samisk språkteknologi. In *Nordisk Sprogteknologi 2004: Aarbog for Nordisk Sprogteknologisk Forskningsprogram 2000-2004*, edited by H. Holmboe, pp. 57–62. Museum Tusculanums Forlag, København.

Nickel, Klaus Peter and Pekka Sammallahti. 2011. *Nordsamisk grammatikk*. Davvi Girji, Karasjok, 2. hapmi = utgave, 1.deaddileapmi = opplag edn.

Pirinen, Flammie, Krister Lindén, et al. 2014. State-of-the-art in weighted finite-state spell-checking. In *Computational Linguistics and Intelligent Text Processing 15th International Conference, CICLing 2014, Kathmandu, Nepal, April 6-12, 2014, Proceedings, Part II*.

Pirinen, Tommi, Miikka Silfverberg, and Krister Linden. 2012. Improving finite-state spell-checker suggestions with part of speech n-grams. In *Computational Linguistics and Intelligent Text Processing*, edited by Alexander Gelbukh. International Conference on Intelligent Text Processing and Computational Linguistics ; Conference date: 11-03-2012 Through 17-03-2012.

Pirinen, Tommi A and Sam Hardwick. 2012. Effect of language and error models on efficiency of finite-state spell-checking and correction. In *Proceedings of the 10th International Workshop on Finite State Methods and Natural Language Processing*, edited by Iñaki Alegria and Mans Hulden, pp. 1–8. The Association for Computational Linguistics, United States. International Workshop on Finite State Methods and Natural Language Processing ; Conference date: 23-07-2012 Through 25-07-2012.

Strubell, Emma, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243* https://doi.org/10.18653/v1/P19-1355.

Trosterud, Trond and Linda Wiechetek. 2007. Disambiguering av homonymi i nord- og lulesamisk. In *Sámit, sánit, sátnehámit. Riepmočála Pekka Sammallahtii miessemánu 21. beaivve 2007*, edited by Ante Aikio and Jussi Ylikoski, Suomalais-Ugrilaisen Seuran Toimituksia 253, pp. 347–354. Suomalais-Ugrilainen Seura, Helsinki.

Trón, Viktor, Andras Kornai, György Gyepesi, László Németh, and Péter Halácsy. 2005. Hunmorph: Open source word analysis. In *Proceedings of the Workshop on Software. Association for Computational Linguistics*, pp. 77–85. https://doi.org/10.3115/1626315.1626321.

Wiechetek, Linda, Sjur Nørstebø Moshagen, and Kevin Brubeck Unhammer. 2019. Seeing more than whitespace — tokenisation and disambiguation in a North Sámi grammar checker. In *Proceedings of the 3rd Workshop on the Use of Computational Methods in the Study of Endangered Languages Volume 1 (Papers)*, pp. 46–55. Association for Computational Linguistics, Honolulu. https://doi.org/10.33011/computel.v1i.403.

A. Appendix. Some tips and tricks for FST

Below is a selected set of regex examples showing solutions to some specific problems.[19]

A.1. Transposition and permutation

Hitting right keys in a wrong order may result in letter transpositions, e.g. Estonian **blianss* instead of intended *bilanss*, and permutations, e.g. **skepulatsioon* instead of intended *spekulatsioon*. The task of a regex is to re-order a few letters.

Transposition may be encoded as a set of pairs of all letters of the alphabet being swapped, e.g. for adjacent letters:

```
[{ab} -> {ba}] | [{ba} -> {ab}] | [{ac} -> {ca}] | [{ca} -> {ac}] |
...
[{yz} -> {zy}] | [{zy} -> {yz}]
```

However, it can also be modelled as a process where a letter disappears from one side of some other letter, and appears on the other side, that is, becomes zero and emerges from zero. (The other letter - any letter, for that matter - is expressed by a ?-mark, denoting "any symbol".) The resulting FST will also transform a pair of identical letters to the same pair, but in our modification-plus-checking workflow this makes no real harm.

```
# transposition 12 -> 21
[a:0 ? 0:a] | [b:0 ? 0:b] |
...
[z:0 ? 0:z]
```

This way of expressing transposition makes a smaller transducer than would be the alternative with explicitly listed pairwise expressions, because using the ?-mark imposes fewer restrictions on the language than explicitly listing the transposition pairs.

In addition, defining a transposition via a letter disappearing and appearing, it makes one notice that transposition error is just a special case of permutation errors, i.e. transposition is permutation of adjacent letters. For example, the Estonian misspelling list contains a typo **proessorf*, where the error is a permutation *fessor -> essorf*. Below is an example of expressing permutation error correction.

```
# permutation 123 -> 312
[0:a ? ? a:0] | [0:b ? ? b:0] |
...
[0:z ? ? z:0]
```

A.2. Repetition

It may happen that a part of a word is mistakenly re-typed, e.g. Estonian **minimimaalne* instead of intended *minimaalne*, and the task is to delete this repeated part.

Repetition-expressing regexes are notorious for blowing the transducer's size up. In order to alleviate that, one may use the ?-mark again, under-specifying the context and thus arriving at a smaller compiled transducer.

```
# abab -> ab
[
a (->) "<COR>" || _ ? a ,,
b (->) "<COR>" || _ ? b ,,
...
z (->) "<COR>" || _ ? z
]
.o.
[ [? - "<COR>"]* "<COR>" "<COR>" [? - "<COR>"]* ]
```

[19] We use the Xerox regular expression notation, c.f.Beesley and Karttunen (2003)

The regex above ultimately deletes the first letter pair from a sequence of two similar pairs. First, any letter is optionally substituted with a `<COR>` tag, if the same letter also appears as next-to-next. This transducer is then composed with another one, which requires exactly two adjacent `<COR>` tags to be in the resulting string. This ensures that there were exactly two adjacent letters that were substituted, meaning that there was a repetition pattern like `abab` in the wordform. Once `<COR>` tags are removed, the removal of two repeated letters is finished.

A.3. Regex as linguistic abstraction

Finally, below is an example on how to express a linguistic abstraction in a regex shorthand. Sámi writers tend to be confused on how to write similarly sounding phones. For example, *k-sound* may be written in North Sámi as *g*, *gg*, *k*, *kk*, or *hk*, and the list of misspellings contains many word pairs where these letters are confused, e.g. **geatgi - geatki*, **Mákkarávjui - Máhkarávjui*, **sámedikkeválggas - sámediggeválggas*, **Johtolagii - Johtolahkii*, **Sámedigge - Sámedikke*, **ohkiin - ogiin*.

The task is to substitute a wrong letter sequence with a correct one, while keeping the intended sound sequence. It is convenient to model this orthography-related uncertainty via confusion sets. Below is an example of expressing a confusion set of 5 letter combinations that are used for writing down *k-sound*:

```
[  [ { g } | { gg } | { k } | { kk } | { hk } ] : [ { g } | { gg } | { k } | { kk } | { hk } ]  ]
```

The expression means that any of these combinations may be substituted for any other. The regex also redundantly substitutes a combination for itself, resulting in a ”modification” that is the same as the original; such modifications get discarded by the speller module, so they make no harm except waste a little extra time.