Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 1167–1177
Marseille, 20-25 June 2022

Unmasking the Myth of Effortless Big Data — Making an Open Source Multilingual Infrastructure and Building Language Resources from Scratch

Linda Wiechetek, Katri Hiovain-Asikainen, Inga Lill Sigga Mikkelsen, Sjur N. Moshagen, Flammie A. Pirinen, Trond Trosterud, Børre Gaup
Department of Language and Culture, UiT the Arctic University of Norway
giellalt@uit.no

Abstract

Machine learning (ML) approaches have dominated *Natural Language Processing* (NLP) during the last two decades. From machine translation and speech technology, machine learning tools are now also in use for spellchecking and grammar checking, with a blurry distinction between the two. We unmask the myth of effortless big data by illuminating the efforts and time that lay behind building a multi-purpose corpus with regard to collecting, marking up and building from scratch. We also discuss what kind of language technology tools minority language communities actually need, and to what extent the dominating paradigm has been able to deliver these tools. In this context we present our alternative to corpus-based language technology – knowledge-based language technology – and we show how this approach can provide language technology solutions for languages being outside the reach of machine learning procedures. We present a stable and mature infrastructure (*GiellaLT*) containing more than hundred languages and building a number of language technology tools that are useful for language communities.

Keywords: infrastructure, corpus, text processing, minority languages, finite state technology, knowledge-based nlp, grammar checking, TTS, ASR, speech technology, spellchecking

1. Introduction

During the last two decades, i.e. early 2000s, machine learning approaches have dominated the field of *natural language processing* (NLP). The philosophy has been to have machines learn behaviour from large corpora, thereby offering speech technology, machine translation and spelling and grammar checking.

For at least 3/4 of the world's languages, this is bad news. They have less than 20,000 speakers, and even with a school system teaching mother tongue literacy and a language policy encouraging publishing, such small populations would have a hard time producing enough text to be able to model the language in a reliable way using the dominating NLP paradigm. On top of that, weak literary traditions give rise to corpora too unreliable to be able to function as a model for NLP tools.

From the reactions we get on our work in the current academic NLP community contexts, we experience an absence of understanding for why we choose to work rule-based, despite our explanations of unavailable noise-free data. There seems to be the idea that big data comes for free. In addition, since most of the work within the dominating methodology is done on morphologically simple languages, there seems to be a general lack of understanding of the methodologies and technologies needed to work efficiently with polysynthetic and other morphologically complex languages.

This article has a clear answer to the big data question — corpus data does not come for free, as illustrated in the three test cases we present. Building corpora requires proficient writers or speakers of the language. In addition, it requires proof-readers, translators and experts who can mark up the texts with regard to errors or other traits of the text. While these may be available for some languages before developing language tools, many languages need to build them from scratch.

However, there is a way to produce the NLP tools needed by the language community without Big Data. We show how a set of knowledge-based methods is able to analyse and generate morphologically complex languages reliably, as well as a language-independent infrastructure that makes it possible to share developmental costs among the languages involved.

We have within 20 years built language resources for several Sámi languages from scratch – this includes lexica, morphological and syntactic analysers. Even though we had a number of non-digital resources available, these were far from exhaustive. This means that our work also included normative discussions, requests and suggestions to the language normative organs, error classifications, grammatical descriptions of phenomena not included in grammar books. In several cases, these phenomena needed traditional linguistic research.

We argue that both a functioning literacy and thereby also corpus-based approaches are dependent upon rule-based tools of the type produced via our approach.

2. Background

2.1. What does the language community need?

At present, several minority language communities with a weak literary tradition try to strengthen the position of the language in society. In doing so, they find themselves in a situation lacking the infrastructure needed to do so, infrastructure that majority language speakers take for granted.

Looking at the writing process, there must be a keyboard layout. Most orthographies contain letters outside the A-Z (or Cyrillic А-Я) range, these letters must be placed on the keyboard in ergonomically sound positions. Layouts should be easy to install on the operating systems on both computers and mobile devices.

In order to produce text with a consistent spelling in an efficient way, the language community will need a spelling checker. Since there are few proofread texts, a spelling checker can not be made by learning from an existing corpus. Not only will the existing corpora be too small, they will typically also be too unreliable. With a weak literary tradition, text often deviates from the norm, and the existing body of text thus can not be taken as a substitute for an explicit norm.

Language communities wanting to strengthen their language do not start from scratch. In practice, the languages of such communities have already been described to a certain extent. Given the present state of linguistic typology research, there will in all relevant cases be scholars devoted to the language, who have made dictionaries covering the core vocabulary and a basic grammar. In order to teach the language to new speakers and to publish texts, the language will still need a practical orthography, an orthography with design principles radically different from the ones used by philologists and typologists. Making a good practical orthography and rewriting the philologists' grammars and dictionaries by using this orthography is at the core of the language planning process.

Machine translation into a minority language is of no use when the output is unreliable and the language community bilingual, thus in a position to choose the majority language original instead. Translation from the minority language is of no use if there is no monolingual text to translate.

Based upon experience with around 50 language models (cf. *giellalt.github.io*) we found that if the lexicon is electronically available and the grammatical structure reasonably clear, a grammatical model covering around 80 % of the word forms in running text can be built in a couple of months. Such a model will be able to support electronic dictionaries, lemmatise text and help insecure writers. It will not be enough for reliable spellchecking or linguistic analysis. Making a language model good enough to reliably predict what is and what is not part of the language will take years.

2.2. Related work

Several studies discuss key aspects of the infrastructure presented here, e.g. Pirinen and Tyers (2021), Moshagen et al. (2013). We are not aware of too many similar open source infrastructures that contain both the same amount of languages and different end user applications that actually work. One example is the rule-based machine-translation system Apertium (Khanna et al., 2021a) that hosts repositories for language resources for 51 machine translators. In addition to containing machine translation pairs in use in different practical applications, the language models underlying them may also be used for a wide range of applications. As a part of the *GiellaLT* infrastructure work we have made a pipeline for including our language models in the *Apertium* pipeline, our MT programs are thus made using the *Apertium* formalism.

The producers of statistical and corpus-based models have also started to gather their data and models into larger multilingual repositories. If we try to draw parallels, for example within the recent neural models, one could compare our repository to the likes of *huggingface.co* that hosts a repository of language models and APIs to access them (Wolf et al., 2020). However, they are still mainly targeting English and a handful of well-resourced languages, and while a number of lower-resourced languages are provided with models and data, the data seems to be very limited. In speech technology and corpora, there is also the common voice project from Mozilla that gathers speech resources for multiple languages (Ardila et al., 2019), and eventually provides spoken language models.

3. A multilingual infrastructure from scratch

In this article, we will assume that the basic language planning work has been done and that there is a practical orthography and access to the lexicon of the language in machine-readable format and a basic grammar. This description holds for a large group of languages that still have no language technology tools in place.

3.1. The *GiellaLT* infrastructure

The foundation for the work presented in this article is the multilingual infrastructure *GiellaLT*, which includes numerous languages — 130 altogether — that have little or no data, a rare case in the NLP world. Everything produced in the *GiellaLT* infrastructure is under free and open licences and freely available. The corpora are available with free licensing where possible. The infrastructure is split code-wise in three GitHub organisations: *GiellaLT* containing the language data for each language, *Divvun* containing language independent code for the infrastructure, and *Giellatekno* for corpus infrastructure. End user tools served by the Divvun group are at *divvun.no* & *divvun.org*, and tools served by the Giellatekno group at *giellatekno.uit.no*, both at *UiT—Norway's Arctic University*.

We build systems that include lexical data as well as rules governing morphophonology, syntax and semantics as well as a number of application specific information, e.g. grammatical rules for grammar checking, phonetic rules for *Text-To-Speech* (TTS) and so forth.

The language-independent work is currently done within the infrastructure, the language-independent features and updates that are relevant to all languages are semi-automatically merged as they are developed. To ensure that language independent and common features and updates do not destroy existing language data or use case, we enforce a rigorous continuous integration based testing regime. The current system for testing is a combination of our long-term investment in testing within the infrastructure locally for developers—combined with modern automatic testing currently supplied by GitHub actions.

Another part of the *GiellaLT* philosophy is that of reusable and multi-purposeful resources, cf. Antonsen et al. (2010). This is true for all of our work, from corpus collection to cross-lingual cooperation.

Despite the lack of data, there are high-level tools in *GiellaLT* such as machine translation, TTS, spelling and grammar checkers and more, that have been very well received in the language communities. This would not have been possible without first developing basic tools such as keyboards, morphological analysers and spelling checkers.

3.2. Keyboards

To be able to type and write a language, you need a keyboard. Using the tool `kbdgen`, one can easily specify a keyboard layout in a *YAML* file, mimicking the actual layout of the keyboard. The listing below shows the definition of the Android mobile keyboard layout for Lule Sámi. The `kbdgen` tool takes this definition and a bit of metadata, combines it with code for an Android keyboard app, compiles everything, signs the built artefact and uploads it to the Google Play Store, ready for testing.

```
modes:
  android:
    default: |
      á w e r t y u i o p å
      a s d f g h j k l ø æ
         z x c v b n m ŋ
```

The tool supports generating keyboard apps or installer packages for Android, iOS, macOS, Windows, Linux (X11 and m17n) and Chrome OS. There is also experimental support for generating *Common Language Data Repository* (CLDR) XML files, *Scalable Vector Graphics* (SVG) files for fast layout debugging, and finite state transducers for neighbour key mistyping error models. The Windows installer includes a tool to register unknown languages, so that even languages never seen on a Windows computer will be properly registered, and thus making it ready to support proofing tools and other language processing tools.

3.3. Morphological analysers

The foundation for all linguistic processing in the *GiellaLT* infrastructure is the morphological analyser, built using formalisms from Xerox: `lexc`, `xfst` and optionally `twolc`. From these source files, the infrastructure creates *finite state transducers* (FST's) using one of three supported FST compilers: Xerox tools (Beesley and Karttunen, 2003), *HFST* (Lindén et al., 2013), or Foma (Hulden, 2009).

All language models are written as rule-based, full form lexicons with explicit morphological descriptions and morphophonological alternations. This makes it possible to create language models for any language, including minority and indigenous languages with no digital presence, as long as there is cooperation with the language community and native speakers.

Support for multiple usages and reuse of the same lexical data is added through analyser tags, so that the compiled FST can be filtered and manipulated to fit a specific usage scenario.

3.4. Morphological and syntactic disambiguation and tagging

All higher-order linguistic processing is done using the VISLCG3 (*visl.sdu.dk*) implementation (Didriksen, 2010) of Constraint Grammar (Karlsson, 1990).

It has over the years proven both robust, fast and flexible, allowing rule-based morphological disambiguation, as well as syntactic and semantic tagging, cf. Table 1. It has also successfully been applied in high-quality grammar checker applications and machine translation systems, and is often used to create the reference tagged corpus used to train machine learning models. Ambiguity in the Sámi languages is mostly based on homonymy between morphological forms of the same part of speech, and less on PoS homonymy between uninflected lemmata.

	North Sámi		**Lule Sámi**	
	Prec	**Recall**	**Prec**	**Recall**
PoS	0.99	0.99	0.94	0.97
morph. dis.	0.93	0.95	0.83	0.94
dependencies	1	1	1	1

Table 1: North and Lule Sámi analysers.

3.5. Tokenisers

Tokenisation is based on an FST model initially presented by Karttunen (2011) in the Xerox tool `pmatch`. The resulting FST is applied using `hfst-tokenise`. The basic idea is to use an FST as a pattern matching tool for natural languages, and add markup to the input text for matching patterns. The FST is used as a tokeniser in a left-to-right, longest-match manner. The tokeniser performs tokenisation and analysis in one go and can handle both known and out-of-vocabulary items. Secondly, the formalism has been extended with explicit backtracking functionality, to allow for multiple tokenisations. The output of ambiguous tokens is disambiguated using linguistic rules with VISLCG3 to specify the correct tokenisation given the context and the available linguistic analyses. This makes it possible to achieve near-perfect tokenisation. In our tokenisation, sentence boundary detection is treated as a special case of ambiguous tokenisation, and solved in the same way, approaching near-perfect sentence boundary identification, cf. Wiechetek et al. (2019b).

3.6. Spelling checkers

As mentioned briefly above in section 3.3, in the *GiellaLT* infrastructure a descriptive language model can be turned into a normative one by way of FST filtering: removing all strings tagged as non-normative. This makes it easy to create acceptors for spelling checkers. Modelling typical spelling errors is also done using FST's, with a default setup that should give reasonable suggestion quality out of the box, but with great flexibility and possibilities for fine-tuning and alternative ways of building error models.

The *GiellaLT* infrastructure also includes a fast implementation of a speller engine using FST's (the acceptor and the error model), and integration with the most popular operating systems and office packages. Combined with a distribution and update tool called *pahkat* as well as continuous integration and delivery, it is possible to develop, test and deliver spellers to the language community with very short cycles. This allows for a good feedback cycle between the language community and the developers of the spelling checker, where the community members can see that their feedback is acted upon, and new updates available in short time.

The rule-based framework allows building high quality spellers even without digital data resources. It should also be pointed out that reusing a language model as a spelling checker would not have been possible using a non-rule-based framework.

3.7. Grammar checkers

Since 2019 the *GiellaLT* infrastructure supports building grammar checkers (Wiechetek et al., 2019a). The grammar checker setup is chaining together several of the modules described above into a pipeline, roughly as follows: `tokenisation & analysis` ⇒ *`Multiword expression`* `(MWE) disambiguation` ⇒ `spellchecking of unknowns` ⇒ `disambiguation` ⇒ `error detection` ⇒ `error correction`. The main technologies used are `hfst-tokenise`, VISLCG3 and our spelling checkers. Evaluations of different error types show good results of the North Sámi grammar checker *GramDivun*. Compound error correction reaches a precision of 81.0% (Wiechetek et al., 2021a). Regression tests give up to 88.8% precision for all a number of error types (real word errors, compound errors, morpho-syntactic errors, formatting errors. (Wiechetek et al., 2021b) The framework for the grammar checker is Constraint Grammar (Karlsson, 1990; Didriksen, 2010). Constraint Grammar as a rule-based approach is a very good fit as it allows partial parses, unfinished disambiguation, and is robust against remaining ambiguities. In addition, the rule-based approach makes it possible to build grammar checkers for languages with no or very little digital resources.

3.8. Machine translation

Another high-level tool available within the *GiellaLT* infrastructure is machine translation. It works in cooperation with the *Apertium* infrastructure (Khanna et al., 2021b), which is also largely rule-based and FST-based. The monolingual resources are developed within the *GiellaLT* infrastructure, using the same morphological analysers as for other tools, but slightly tweaked to match the requirements in Apertium. The output is a set of FST's made available to Apertium, which contain the bilingual resources for a given language pair. This has resulted in systems with Word Error Rates as good as 0.11 (cf. Antonsen et al. (2017), for North to Inari Sámi). The North Sámi to Norwegian *Machine Translation* (MT) system delivers close to 1,000 translations a day[1] (cf. Antonsen and Trosterud (2020, p. 60)).

4. Corpus — Three test cases

This section deals with building and handling corpus in the *GiellaLT* infrastructure. The question of big data is usually not addressed in articles that use the data for their machine learning approaches without creating the data. However, when calculating time-efficiency of the approach, this should be part of the equation. With our three test cases, we illuminate the actual work behind an adequate corpus for a certain tool and the challenges behind the corpus work. Building a corpus with good quality requires selecting native language texts from different domains, marking up a corpus to make it multi-use, and also building a special-purpose corpus (i.e. for speech technology) from scratch.

4.1. Collecting corpus texts

With our infrastructure, it is possible to build tools without corpora but with the help of native speakers. With curated corpora, we can verify that our tools work on a wide range of real-world linguistic phenomena.

We have collected corpora for five of the Sámi languages: North, Lule, South, Inari and Skolt Sámi as well as for 13 other circumpolar languages. The Sámi corpus is owned by the Norwegian Sámi parliament, and all corpora are administered and made accessible to the public by the Divvun and Giellatekno groups. The corpora are split in two based on restrictions set by the copyright owners. Researchers and anyone else can freely download the free part. The whole corpus, also the restricted part, is accessible via a public search interface[2]. We have written a tool named CorpusTools to

[1]jorgal.uit.no/. The system is documented at giellalt.github.io/mt/MachineTranslation.html.

[2]gtweb.uit.no/korp (Sámi), gtweb.uit.no/f_korp (Baltic Finnic and Faroese), gtweb.uit.no/u_korp (other Uralic languages). Cf. also More info about the corpora.

administer, convert and analyse the corpus texts. Original texts and their metadata are saved in svn repositories, then converted to a common XML format, to ease further use of the texts.

Texts in Sámi languages are published daily in both media and by public bodies required to communicate in writing in Sámi. We have written crawlers to ease the collection of and to maintain consistent metadata about these texts. The crawlers gather parallel language versions from those sites that have unambiguous links to such data. Since most of the publishers (typically online) have to provide their site in both Sámi and the majority languages, and they provide interlinks between these page, we are able to build up a rather comprehensive parallel corpus, as well.

Having gathered text since 2005, the largest Sámi corpus is the one for North Sámi, with 38.94 million tokens. The four other Sámi corpora all contain less than 3 million words. The North Sámi corpus is a quite big corpus for an indigenous language, but small compared to majority languages. The respective majority language corpora contain 18.4 billion words (Norwegian), 13.9 billion tokens (Swedish) and 14.1 billion words (Finnish)[3]. It goes without saying that corpora containing billions of words offer possibilities not available to corpora containing millions of words or even less.

This does not mean that smaller corpora are not useful. To the contrary, Antonsen and Trosterud (2020) show that the South and Skolt Sámi language communities (constituting less than 500 speakers each) over a 3-month period via a dictionary interface accessed their respective corpora (containing 1.5 and 0.2 million words, respectively) 47,000 times each. The language community sees the corpora as useful, despite their small size.

We would also like to have a balanced corpus with regard to regional dialects of the same language. The Sámi languages have a stronger legal protection in Norway than in Sweden, our corpus therefore consist of more text written in Norway than in Sweden. This has consequences for some of the tools we are developing, quite clearly for translation memory, but also to some extent for other types of tools such as TTS.

For North Sámi and Norwegian, we have managed to build a parallel corpus containing 3.9 million words. For the other Sámi languages, the corresponding pairs contain 230,000 words or fewer. These corpora mainly contain administrative texts. They may be used as a foundation for vocabulary development, using word alignment methods developed for phrase-based machine translation, (Gerstenberger et al., 2013) or as a source for dictionary examples (Antonsen and Trosterud, 2020).

[3]For Norwegian, see Kummervold et al. (2021), which presents the corpus underlying a language model for Norwegian. The Swedish and Finnish corpora are searchable at spraakbanken.gu.se/korp and korp.csc.fi/korp/, respectively.

4.2. Corpus mark-up — upgrading a corpus

The next step towards corpus-assisted NLP is its refinement. When we want to use a corpus for specific tasks, underlying structures need to be made accessible in addition to raw text. This can be part-of-speech, morphological, syntactic, semantic, pragmatic information on the one hand, and error mark-up together with error categorisation and correction, on the other hand.

The approaches for these two are contrastive. Part-of-speech (PoS), morphological, syntactic and semantic mark-up is done automatically by our rule-based FST and Constraint Grammar analysers and made available to the language community in the online application *Korp*. The error mark-up is done manually serving as a database for testing and evaluating the quality of our spellcheckers and handwritten grammar checker rules.

The earliest manual error mark-up started in 2006 and served as unseen test data for Lule and North Sámi spellers. It was further used to automatise spellchecker testing for Greenlandic, Icelandic, and South Sámi. Moshagen (2008) describes an initial testbench for spellcheckers, and Antonsen (2012) takes a first step towards a grammar checker testbench by adding mark-up of real word errors in addition to non-word spelling errors (i.e. errors that require syntactic context). The L2 corpus of North Sámi has 4,633 words, 800 sentences, 739 misspellings. Today's version of the marked-up North Sámi corpus (*ErrMark-SIKOR*) has 182,450 words and contains mostly administrative and news texts, but also a bit of fiction and the previously mentioned L2 corpus. However, as the rest of the corpus, it is enhanced by error mark-up for grammatical and formatting errors in addition to spelling errors.

Alternatively, synthetic errors can be created if larger corpora are needed, e.g. to train neural networks for e.g. grammar checking modules. (Wiechetek et al., 2021a) For a synthetically created corpus with congruence errors for a neural-network based grammar checker, we generated 20,846,804 sentences and 2,551,236,949 words. The problematic part with a synthetic corpus is that we cannot rely on it. For the previously mentioned corpus, even though synthetic errors were inserted carefully (correct forms replacing incorrect), unexpected homonymies or unclear contexts can lead to the inserted forms still being correct. This can only be discovered by manual checks or a rule-based grammar checker. Another problematic issue is that the corpus will not reflect the actual distribution of errors made in the real world.

The Lule Sámi corpus mark-up started in 2013 with 29,527 words of unpublished native speakers texts, with 1,505 non-word errors. With a multi-use in mind, this corpus was proofread and marked up with other error types including 1,322 morpho-syntactic, syntactic and lexical errors. The texts had neither been spellchecked nor proofread. The large part of errors (altogether 2,827) is probably due to the young written tradition. A standard had first been established in 1983 (Magga,

1994). This also illustrates the urgent need for spelling and grammar aiding tools developed within *GiellaLT*.
A systematic error mark-up has originally been developed for spellchecking, but then been extended to grammatical error mark-up, formatting, punctuation and more. The goal behind it is the development of a machine-readable multi-purpose corpus without changing its originality. Important principles are, therefore, consistent mark-up (in terms of range and error type) and compatibility with our tools. That means, for example, that if we have listed a lexical item as a multi-word, we also need to mark it up as a multi-word.
The error mark-up syntax follows a number of guidelines[4] (Moshagen, 2014) and applies eight different general error types: orthographic, real word, morpho-syntactic, syntactic, lexical, formatting, foreign language and unclassified errors. The error is enclosed in curly brackets, followed by its correction in another set of curly brackets. The second curly bracket may or may not include a part-of-speech, morpho-syntactic criteria and a subclassification of the error type.
Orthographic errors include non-words only. They are misspellings confined to single (error) strings, and the traditional speller should detect them. *Real word errors* are misspellings that cannot be detected by a traditional speller, they are seen as errors due to the context in which they occur. *Morpho-syntactic errors* are case, agreement, tense, mode errors. They require an analysis of (parts of) the sentence to be detected. *Syntactic errors* (marked by ¥) require a partial or full analysis of (parts of) the sentence. They include word order errors, compound errors, missing words, and redundant words. *Lexical errors* include wrong derivations. *Foreign language* includes words in other languages that do not require a correction. *Formatting errors* include spacing errors in combination with punctuation.
In ex. (1), the syntactic error is a missing word and the correction is adding the subjunctive *ahte* 'that'.

(1)	Illá	{jáhkken}¥{missing|jáhkken ahte}
	hardly	think.past.1sg
	lei	duohta.
	be.past.3Sg	true
	'I hardly thought that it was true.'	

Regarding the span of an error, we typically mark as little as possible, even if larger parts of the sentence are responsible for the identification of the error. This is done to facilitate matching error mark-up with grammar checker marking of the error, and it has a direct effect on automatic evaluation. Most of the frameworks we use to process language material in context, e.g. Constraint Grammar, take a token-based approach to language processing, and therefore marking several words can get cumbersome and should be avoided if possible.
The marking of errors has had consequences beyond of what we had originally envisioned. Not only has it resulted in a corpus that can be used in automatic evaluation of our tools, it has also forced us to categorise errors according to the underlying principles of the spelling and grammar checker, which had not necessarily been the same ones a linguist would see in the first place. It became apparent that grammatical errors marked-up before we started working on a grammar checker needed to be recategorised, and their span needed to be shortened. The biggest challenge in marking up a corpus has been consistency, i.e. the same type of error should always be marked in the same way. In addition, marking errors should follow the same pattern in all languages in the *GiellaLT* infrastructure. The mark-up process resulted in an overview of challenges native speakers have with the written languages, which can help to improve literacy education in schools (Antonsen, 2012). It also revealed where the written language lacks writing rules and norms, which could then be passed on to *Giellagálldo*, the normative organ for the Sámi languages.

4.3. Speech corpora and Text-To-Speech

A TTS tool is made to be able to synthesise intelligible speech output from any unseen text input in a particular language. The main objective for developing speech technology tools for indigenous languages is to meet the needs of modern language users in all language communities equally. For the Sámi languages, this would mean equal possibilities to use Sámi in the same contexts as the majority languages. In this way, developing speech and language technology tools for the Sámi languages also contribute to the revitalisation of these languages. Additionally, speech technology tools are useful for many users, also those with special needs. These include language learners (Yaneva, 2021), people with dyslexia, vision impaired individuals, and speakers of the language that are not used to read it.
Developing TTS for an indigenous language with few resources available can be challenging. Any linguistic description, grammar or language learning material is useful, but for speech technology purposes, it is important to have at least some amount of speech material and corresponding text, provided by a native speaker of the language. In this way, it is possible to study the relationship between text and speech in a particular language and to produce a phonetic description in a form of a grapheme-to-phoneme mapping. This mapping (or *text-to-IPA* rule set) can already be used to build a very simple and "old-fashioned" but still usable TTS application, such as the espeak formant synthesis (Kastrati et al., 2014; Pronk et al., 2013). As this framework does not require a speech corpus but only a set of phonetic and phonological rules, any language can be added to the list of the languages covered by `espeak`, only utilising the knowledge of native speakers.

4.3.1. Building a speech corpus

The modern approaches to TTS involve machine learning and complex modelling of speech, which brings

[4] giellalt.uit.no/proof/spelling/testdoc/error-markup.html

	Orth.	IPA	Transl.
Q3	*oarre*	[ʔo͡ɑrːrɪɛ]	’a squirrel’ Nom.Sg
Q2	*oarre*	[ʔoɑrːɪɛ]	’a squirrel’s’ Gen.Sg
			’a reason’ Nom.Sg
Q1	*oare*	[ʔoɑrɪɛ]	’a reason’s’ Gen.Sg

Table 2: Ternary length contrast of consonants in Lule Sámi, underspecified in the orthography. Abbreviations: Q3 – overlong, Q2 – long, Q1 – short. Originally presented in Fangel-Gustavson et al. (2014).

	Orth.	IPA	Transl.
Q3	*beassi*	[pe͡æsːsɪ]	’birchbark’ Nom.Sg
Q2	*beassi*	[peæsːɪ]	’birchbark’ Acc.Sg
			’(bird’s) nest’ Nom.Sg
Q1	*beasi*	[peæsɪ]	’(bird’s) nest’ Acc.Sg

Table 3: Ternary length contrast of consonants in North Sámi, underspecified in the orthography. Abbreviations as in Table 2.

in the requirement for big amounts of speech data to build the models from, ideally covering all phonological contrasts and sound combinations (diphones, triphones) in a given language. This is because in a data-driven or *corpus-based* speech synthesis developed that utilize deep neural networks, the association between textual features and acoustic parameters is learned directly from paired data – the sentence-long sound files and the corresponding texts. The sum of the learned knowledge from the paired data construct the acoustic model (see, e.g., Watts et al. (2016)).

The building of the speech corpus starts from collecting a suitable text corpus which corresponds to at least 10 hours of recorded read speech, that has been shown to be enough to achieve an end-user suitable TTS system for North Sámi (Makashova, 2021). In Divvun, we focus on open-source methodologies, in which case it is important to build a collection of open source texts, with a CC-BY (Creative Commons) licence.

For our on-going Lule Sámi TTS project we reused a part of a Lule Sámi gold corpus from 2013, and collected additional texts we knew to be well written and already proofread, before proofreading these texts once more to avoid confusion when reading the text aloud. We collected and constructed a Lule Sámi text corpus consisting of various text styles (news, educational, parliament etc.) with altogether over 74,000 words. This will approximately correspond to 12 hours of speech recordings when read aloud by professional voice talents.

4.3.2. Text processing

Most orthographies are underspecified with respect to the pronunciation. This creates interesting questions when converting a standard orthographic text to audio waves. In the cases of Lule and North Sámi there is a class of nouns where consonant gradation (i.e. length alternation) is not expressed in the orthography, while still being grammatically crucial, as it is the sole marker of the difference between different syntactic functions, especially *singular nominative* vs *singular genitive*, and for North Sámi also *singular accusative*. That is, for this class of nouns the only difference between the subject and the possessor or between the subject and the object, is expressed through a length distinction that is *not* present in the standard orthography, as seen in Tables 2 and 3. That distinction has to be recreated when converting the orthographic text to a phonemic representation. There are other underspecifications in the orthography, but these are the most crucial.

The length contrast is encoded in the FST model at an intermediate level, but during compilation, this information is lost. We have enhanced the `hfst-pmatch` code to allow the analyser/tokeniser FST to be an on-the-fly composition of two separate FST's, and outputting that intermediate string representation, in effect creating a fake three-tape FST.

With the morphological analysis of all tokens available, we can proceed by disambiguating the sentence, and leaving only the analyses that fit the morphosyntactic context. The end result is that we will be left with the proper analysis (subject or object) *and* with information of the proper length of the word form, to be fed to the module for conversion to IPA. As always, this is done using rule-based components, so we have full control of every step and are able to correct errors in the IPA transcription. There is still a fallback module for cases of unknown words and names.

The IPA transcription provided by the FST technology described above can further improve the accuracy of the TTS, especially the alignment between sounds and characters. When training a speech model with the IPA transcriptions as text input (instead of standard orthography) in a deep neural network structure, the letter-to-sound correspondence will likely be more transparent, also with the ternary quantity cases described above.

The rule-based approach, reusing many components from other parts of the *GiellaLT* infrastructure, also means that high quality speech synthesis is within reach for most language communities.

4.3.3. Approaches to Text-to-Speech

We have experimented with two different open source TTS methodologies: Ossian (Suni et al., 2014) and a Tacotron implementation (largely based on Shen et al. (2018)), specially adapted for low-resource languages, like the Sámi languages (Makashova, 2021).

In our experiments, we used a data set consisting of approximately one hour of speech from a native speaker of Lule Sámi, producing nearly intelligible speech. It is clear that for getting better results, at least 10 hours of training data would be needed, but piloting the methods using small experimental data gives us better insight on the requirements for the speech corpus, i.e. the size and audio quality of the data.

As the expectations for the quality of TTS are very high due to the examples from well-resourced languages such as English, using a neural vocoder (such as *WaveNet*, Oord et al. (2016)) that produces realistic, human-like speech is necessary for good usability and user experience. One should not forget that the environmental cost for complex modelling of speech is high (Makashova, 2021), but it is possible to adapt existing speech models by training the models further with additional data and pre-trained models from a "neighbouring" language. This so-called *transfer learning* (Tu et al., 2019) allows for utilising smaller data sets for training, making it possible to use e.g. the North Sámi TTS model as the starting point for the Lule Sámi TTS.

4.3.4. Future work: approaches to automatic speech recognition

In addition to TTS, we are working towards developing a tool for *automatic speech recognition* (ASR) for Sámi. In Makashova (2021), TTS and ASR models were trained simultaneously in a dual transformation loop, using the same *read speech* data set, corresponding to only six hours of speech from two speakers, three hours each. The model was trained for 30 000 steps and for the evaluation of the model, it reached a WER (Word-Error-Rate) of 41% and 0.5 loss. The most common error types in the ASR predictions seem to be in word boundaries (**earáláhkai – eará láhkái*). These kinds of errors would, however, be easy to correct using the Divvun's spell checking software.

One of the most important differences between training the TTS and ASR models would be that for TTS, the training material needs to be very clean in terms of sound quality and there needs to be as many recordings from a single speaker as possible. For ASR, on the other hand, the recorded materials can be of poorer sound quality and preferably from multiple speakers and from different areal varieties of a language as long as there are good transcriptions of the speech.

State-of-the-art ASR frameworks normally require up to 10,000 hours of multi-speaker data for training reliable and universal models that are able to generalise to any unseen speaker (Hannun et al., 2014). As collecting these amounts of data from small minority languages is not a realistic goal, alternatives such as utilising existing archive materials can be considered for developing speech technology for Sámi. These are provided by, e.g., *The language bank of Finland* and *The language bank of Norway*. As these archive materials contain spontaneous, transcribed spoken materials from various dialects of North Sámi, we are able to significantly improve the WER of our North Sámi ASR.

In summary, the procedures and pipelines described above could be applied to any (minority) language with a low-resource setting, in the task of developing speech technology applications. Most of the applications discussed here can be piloted with or further developed with relatively small data sets (even with $<$ 10 hrs of paired data), compared to the amounts of data used for respective tools for majority languages. This is largely possible thanks to the available open source materials and technologies, especially those relying on, e.g., *transfer learning*.

5. Discussion: Big Data

As the previous test cases have shown, big data in terms of large amounts of data of good quality cannot be assumed in a minority language context. One can probably go as far as saying that it cannot be expected in any context except the few big (written) languages. However, big data is usually assumed to just be available when doing a scientific study or developing language technology tools, and the judgement "too little data" can mercilessly decide over the construction of an MT program, inclusion in predominant writing programs (MS Word etc) as well as whole platforms (Android, iOS).

The often heard question of "why not just train a neural model" can usually readily be answered by the lack of data and also the quality of the data for a given task, for example grammatically perfect language data when training a neural model for grammar correction. The lower bounds of required data have been the centre of machine learnt NLP research in recent years. For example, for morphology, the annual SIGMORPHON task has found that a machine learnt model can learn to fill in dictionary inflection tables from just 200,000 gold annotated examples at 20 % accuracy (Kann et al., 2020). A similar test was conducted in the 2018 version of the shared task concerning a larger number of languages and the accuracy of filling just randomly sampled single word-forms giving better breakdown against artificially restricted training set sizes (i.e. not realistic lower-resource scenario): for 100,000 gold-annotated examples one can get to 60 % accuracy, falling to 40's when restricting the resources down to 1,000 examples (Cotterell et al., 2018), compare to Table 1 on page 3. In machine translation, similar results have been shown in WMT shared task on *very low resource MT* (Fraser, 2020), where it is shown that 60,000 aligned sentences is sufficient for MT between high-resource and low-resource language, in the example German-Sorbian. Similar studies exist for many of the fields of NLP, but the general point is that one still needs tens to hundreds of thousands of annotated, aligned, and representative samples to even begin.

6. Conclusion

In this article we have presented our rule-based tools in the *GiellaLT* multilingual infrastructure built during the last 20 years. The *GiellaLT* infrastructure contains building blocks and support for most of the language technology needs of indigenous and minority languages, from the very basic input technologies like keyboards to high-level advanced tools like world-class grammar checking and machine translation. It does this by using rule-based technologies that makes it possible for any language community to get the language technology tools they want and need. All that is needed is a linguist.

Secondly, we discussed the question of costless and efficient corpus-based machine learning models for

building NLP tools needed by a language community (keyboards, spell and grammar checkers, machine translation and Text-to-Speech tools) and also presented an alternative to these models.
We have illustrated the challenges and efforts in collecting good quality native speaker texts and making them digitally available, as well as further marking up the corpus texts in a consistent way in order to use them for NLP tasks such as spelling and grammar checking.
Multi-billion word corpora are the result of decades of work by countless authors, proof-readers and publishers. For most languages, these resources do not exist, and relying upon them for making language models will in practice exclude the vast majority of languages from getting high-quality tools. Secondly, when corpora exist, they are too dispersed to constitute a foundation for normative language models. For certain tasks like TTS, if a speech corpus must be built from scratch, it has to be designed to prioritise quality over quantity of the corpus. We ensure a good quality and multi-purpose speech corpus by working with professional voice talents and language experts that are native speakers of the language.
In conclusion, building corpora is based on big efforts, requires expertise and is time-costly. We have illuminated the work behind three important steps within building corpora - firstly, collecting and digitalising, secondly upgrading, i.e. adding annotation for special purposes, and proofreading, and thirdly converting from one medium/language to another as in recording speech, translating, or other.
So, for machine learning approaches that simply make use of existing corpora, this work does not come for free, it simply has been done by others.
With our multilingual infrastructure and our language resources we show that while there is a need for corpus data for certain tasks, high quality tools needed by a language community can be built time-efficiently without big data in a rule-based manner.

Bibliographical References

Antonsen, L. and Trosterud, T., (2020). *Med et tastetrykk. Bruk av digitale ressurser for samiske språk*, volume 12 of *Samiske tall forteller. Kommentert samisk statistikk*, pages 43–68. Sámi allaskuvla, Kautokeino.

Antonsen, L., Trosterud, T., and Wiechetek, L. (2010). Reusing grammatical resources for new languages. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May. European Language Resources Association (ELRA).

Antonsen, L., Gerstenberger, C., Kappfjell, M., Rahka, S. N., Olthuis, M.-L., Trosterud, T., and Tyers, F. M. (2017). Machine translation with North Saami as a pivot language. In *Proceedings of the 21st Nordic Conference on Computational Linguistics, NoDaLiDa, 22–24 May 2017, Gothenburg, Sweden*, volume 29 of *NEALT Proceedings Series*, pages 123–131. Linköping University Electronic Press.

Antonsen, L. (2012). Improving feedback on l2 misspellings – an fst approach. In *Proceedings of the SLTC 2012 workshop on NLP for CALL*, pages 11–10, Lund. Linköping Electronic Conference Proceedings 80.

Ardila, R., Branson, M., Davis, K., Henretty, M., Kohler, M., Meyer, J., Morais, R., Saunders, L., Tyers, F. M., and Weber, G. (2019). Common voice: A massively-multilingual speech corpus. *arXiv preprint arXiv:1912.06670*.

Beesley, K. R. and Karttunen, L. (2003). Finite-state morphology: Xerox tools and techniques. *CSLI, Stanford*.

Cotterell, R., Kirov, C., Sylak-Glassman, J., Walther, G., Vylomova, E., McCarthy, A. D., Kann, K., Mielke, S. J., Nicolai, G., Silfverberg, M., et al. (2018). The conll–sigmorphon 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27.

Didriksen, T., (2010). *Constraint Grammar Manual: 3rd version of the CG formalism variant*. GrammarSoft ApS, Denmark.

Fangel-Gustavson, N., Ridouane, R., and Morén-Duolljá, B. (2014). Quantity contrast in Lule Saami: A three-way system. In *Proceedings of the 10th International Seminar on Speech production*, pages 106–109.

Fraser, A. (2020). Findings of the WMT 2020 shared tasks in unsupervised MT and very low resource supervised MT. In *Proceedings of the Fifth Conference on Machine Translation*, pages 765–771, Online, November. Association for Computational Linguistics.

Gerstenberger, C.-V., Eskonsipo, B. M. N., and Eira, M. (2013). Digging for domain-specific terms in North Saami. Conference presentation at Oovtast conference, Inari.

Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., et al. (2014). Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*.

Hulden, M. (2009). Foma: a finite-state compiler and library. In *Proceedings of the Demonstrations Session at EACL 2009*, pages 29–32.

Kann, K., McCarthy, A. D., Nicolai, G., and Hulden, M. (2020). The sigmorphon 2020 shared task on unsupervised morphological paradigm completion. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 51–62.

Karlsson, F. (1990). Constraint grammar as a framework for parsing unrestricted text. In H. Karlgren, editor, *Proceedings of the 13th International Confer-*

ence of Computational Linguistics, volume 3, pages 168–173, Helsinki.

Karttunen, L. (2011). Beyond morphology: Pattern matching with fst. In *International Workshop on Systems and Frameworks for Computational Morphology*, pages 1–13. Springer.

Kastrati, R., Hamiti, M., and Abazi, L. (2014). The opportunity of using espeak as text-to-speech synthesizer for Albanian language. In *Proceedings of the 15th International Conference on Computer Systems and Technologies*, pages 179–186.

Khanna, T., Washington, J. N., Tyers, F. M., Bayatlı, S., Swanson, D. G., Pirinen, T. A., Tang, I., and Alòs i Font, H. (2021a). Recent advances in apertium, a free/open-source rule-based machine translation platform for low-resource languages. *Machine Translation*, pages 1–28.

Khanna, T., Washington, J. N., Tyers, F. M., Bayatlı, S., Swanson, D., Pirinen, F., Tang, I., and Alos i Font, H. (2021b). Recent advances in Apertium, a free/open-source rule-based machine translation platform for low-resource languages. *Machine Translation*, 10.

Kummervold, P. E., De la Rosa, J., Wetjen, F., and Brygfjeld, S. A. (2021). Operationalizing a national digital library: The case for a Norwegian transformer model. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 20–29, Reykjavik, Iceland (Online). Linköping University Electronic Press, Sweden.

Lindén, K., Axelson, E., Drobac, S., Hardwick, S., Kuokkala, J., Niemi, J., Pirinen, T. A., and Silfverberg, M. (2013). Hfst—a system for creating nlp tools. In *International workshop on systems and frameworks for computational morphology*, pages 53–71. Springer.

Magga, O. H. (1994). Hvordan den nyeste nordsamiske rettskrivingen ble til. *Festskrift til Ørnulv Vorren*, 25:269–282.

Makashova, L. (2021). Speech synthesis and recognition for a low-resource language: Connecting TTS and ASR for mutual benefit. Master's thesis, University of Gothenburg.

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

Moshagen, S. N. (2008). A language technology test bench – automatized testing in the Divvun project. In R. Domeij, et al., editors, *Proceedings of the Workshop on NLP for Reading and Writing – Resources, Algorithms and Tools*, NEALT Proceeding Series, pages 19—21, Stockholm.

Moshagen, S. (2014). Test data and testing of spelling checkers, 11. presentation at the NorWEST2014 workshop.

Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

Pirinen, T. A. and Tyers, F. M. (2021). Building language technology infrastructures to support a collaborative approach to language resource building. *Multilingual Facilitation*, page 53.

Pronk, R., Intelligentie, B. O. K., and Weenink, D. D. (2013). Adding Japanese language synthesis support to the espeak system. *University of Amsterdam*.

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerrv-Ryan, R., et al. (2018). Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783. IEEE.

Suni, A., Raitio, T., Gowda, D., Karhila, R., Gibson, M., and Watts, O. (2014). The simple4all entry to the blizzard challenge 2014. In *Proc. Blizzard Challenge*. Citeseer.

Tu, T., Chen, Y.-J., Yeh, C.-c., and Lee, H.-Y. (2019). End-to-end text-to-speech for low-resource languages by cross-lingual transfer learning. *arXiv preprint arXiv:1904.06508*.

Watts, O., Henter, G. E., Merritt, T., Wu, Z., and King, S. (2016). From hmms to dnns: where do the improvements come from? In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5505–5509. IEEE.

Wiechetek, L., Moshagen, S. N., Gaup, B., and Omma, T. (2019a). Many shades of grammar checking – launching a constraint grammar tool for north sámi. In *Proceedings of the NoDaLiDa 2019 Workshop on Constraint Grammar - Methods, Tools and Applications*, NEALT Proceedings Series 33:8, pages 35–44.

Wiechetek, L., Moshagen, S. N., and Unhammer, K. B. (2019b). 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)*, pages 46–55, Honolulu, February. Association for Computational Linguistics.

Wiechetek, L., Pirinen, F., Hämäläinen, M., and Argese, C. (2021a). Rules ruling neural networks - neural vs. rule-based grammar checking for a low resource language. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1526–1535, Held Online, September. INCOMA Ltd.

Wiechetek, L., Pirinen, F. A., Gaup, B., and Omma, T. (2021b). No more fumbling in the dark - quality assurance of high-level NLP tools in a multi-lingual infrastructure. In *Proceedings of the Seventh International Workshop on Computational Linguistics of Uralic Languages*, pages 47–56, Syktyvkar, Russia

(Online), September. Association for Computational Linguistics.

Wolf, T., Chaumond, J., Debut, L., Sanh, V., Delangue, C., Moi, A., Cistac, P., Funtowicz, M., Davison, J., Shleifer, S., et al. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Yaneva, A. (2021). Speech technologies applied to second language learning. A use case on Bulgarian. Bachelor's thesis.