UNIVERSITY OF HELSINKI
FACULTY OF ARTS
DEPARTMENT OF DIGITAL HUMANITIES

Master's Thesis

Text Normalisation of Dialectal Finnish

Tiina Koho

Master's Programme in Linguistic Diversity and Digital Humanities

Supervisor: Yves Scherrer 8.11.2022

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Tiedekunta – Fakultet – Faculty	Koulutusohjelma – Utbildningsprogram – Degree Programme	
Faculty of Arts	Master's Programme in Linguistic Diversity and Digital Humanities	
Opintosuunta – Studieinriktning – Study Track		
Language Technology		
Tekijä – Författare – Author		
Tiina Koho		
Työn nimi – Arbetets titel – Title		
Text Normalisation of Dialectal Finnish		
Työn laji – Arbetets art – Level	Aika – Datum – Month and year	Sivumäärä– Sidoantal – Number of pages
Master's Thesis	11 / 2022	53

Tiivistelmä – Referat – Abstract

Text normalisation is the process of converting non-canonical written language to a standardised form. Dialects are an example of nonstandard language that is frequently used and may differ considerably from the standard variety. Furthermore, the Finnish orthography is largely phonemic, making it possible to express many features of the spoken language varieties through writing. This is especially prevalent in an informal setting, such as on social media, where people often tend to write words as they would pronounce them in real life. In natural language processing, tools and applications that have been trained on traditional text material may not achieve desirable results when applied to nonstandard language. Therefore, texts containing nonstandard language often go through a normalisation step, making the data more understandable for natural language processing systems.

This work is based on earlier research done on the topic of Finnish dialect text normalisation. Earlier research has concluded that character-based BRNN (Bidirectional Recurrent Neural Network) models generally achieve higher accuracy in Finnish dialect text normalisation tasks when compared to other neural network architectures. Moreover, it was shown that chunks of words as training data work better than individual words or whole sentences. In order to have comparable results, this work uses the same dataset and methods as the previous research. The data used in training and testing the models is the Samples of Spoken Finnish corpus maintained by the Institute for the Languages of Finland *(Kotimaisten kielten keskus)*, and the OpenNMT toolkit is used for the text normalisation task. The results obtained from the experiments conducted in this work seem to confirm the findings of the previous research. However, the results also seem to indicate that neural network models may benefit from longer input segments in the training data. Other neural network architectures in addition to the BRNN model are also used, but the BRNN models consistently yields better results. The results are evaluated with the WER (Word Error Rate) metric.

Avainsanat – Nyckelord – Keywords

language technology, text normalisation, dialect normalisation, Finnish dialects, neural networks

Säilytyspaikka – Förvaringställe – Where deposited

Helsinki University Library

Muita tietoja – Övriga uppgifter – Additional information

Contents

1 Introduction

A language is never immutable and uniform. The language we speak today is different from the language spoken a hundred years ago; a person from Liverpool most likely uses a variety of English that is distinctive from that of a Minnesotan speaker. Moreover, the language one uses when writing a professional email is probably vastly different from how one would write a casual, informal message to a close friend. These can all be seen as examples of *language variation*. Language change is variation over time and can be observed by comparing historical texts with contemporary ones, should such written records be available for the language in question. Variation in modern languages is something we all can easily observe in our everyday lives, although we may not always be aware of it. Most of us will instinctively use a slightly different form of language when speaking with an elder or a child, and we might deduce that a speaker is from a certain region by listening to how they speak. No one speaks—or writes—exactly the same in every possible situation.

Text normalisation is the process of converting non-canonical written language to a standardised form. This can happen in the form of transforming historical texts to contemporary writing, or mapping user-generated content to a standard, "normal" text. Natural language processing tools that have been trained on traditional text material often suffer from performance drops when applied to non-standard language, such as social media data, and one way to overcome this problem is text normalisation (Matos Veliz et al., 2019). A normalisation step can then be applied to text like any other upstream component, such as tokenisation or named entity recognition (NER), before downstream processes are introduced in the pipeline. Dialectal language, both written and spoken, is a topic of interest in natural language processing, and such data also typically requires a normalisation step before other tools or applications are implemented.

This thesis focuses on text normalisation in the context of Finnish dialects. Finnish is an agglutinative and morphologically rich language, meaning that a considerable amount of grammatical information is expressed already at the word level through a morphological process known as agglutination. This typically results in a large inventory of possible word-forms, since the stem of a word may be combined with one or more inflectional morphemes to indicate a variety of grammatical or syntactic features, such as case, mood, person, or tense (Chahuneau et al., 2013; Tsarfaty et al., 2013). Consequently, the probability of out-of-vocabulary (OOV) words is a non-trivial challenge, especially when working with morphologically rich languages, and the introduction of dialectal words and other non-standard language complicates the situation even further. In the age of social media, dialectal language also finds its way onto communication platforms more frequently, and the data available on dialectal language is steadily becoming increasingly abundant.

1.1 Motivation

Languages differ from each other in how their users tend to use non-standard language in written form. In the case of Finnish, the spoken regional varieties differ greatly from the written standard, and because of the phonemic nature of the Finnish orthography, these dialectal differences can also be expressed through writing (Hämäläinen et al., 2020b). Therefore, in informal writing, people often tend to write words as they would pronounce them in real life. This differs from the typical usage of English in that even though different varieties of the language can have vastly different pronunciations of the same word, the spelling generally stays the same. This makes Finnish a curious case for text normalisation studies, and papers have recently been written on the topic: both on dialect text normalisation (Partanen et al., 2019), where dialectal text is translated into Standard Finnish, and on dialect adaptation (Hämäläinen et al., 2020b), where text written in Standard Finnish is instead adapted to different dialects.

Personally, there were mainly two separate but related motivations for choosing this topic: First, the idea of working with Finnish, my native language, was very appealing to me. Second, I have always been fascinated by different dialects and language variants, and I also speak a regional variant that is slightly different from the standard colloquial dialect of Finnish. Looking into the future, it would be rewarding to see, for example, an automatic speech recognition system developed for Finnish that would understand not only the standard language, but also the different spoken dialects. While I do not use speech data in my own experiments, automatic speech recognition (ASR) and spoken dialects are discussed in Section 2.1.2, and it is a research topic that I find definitely interesting. Should the opportunity arise, I would one day like to work with dialectal speech and automatic speech recognition. However, for the scope of this thesis, I will only focus on text data.

The paper written by Partanen et al. (2019) has also greatly influenced my thesis, and my own experiments rely heavily on their work. In addition to research on the topic, Partanen et al. have also released a text normalisation tool for Finnish as part of an open-source library called Murre[1], maintained by Mika Hämäläinen. The Murre tool can additionally be used to normalise dialectal Swedish and historical Finnish, and there is also an option to generate different Finnish dialects. I started my thesis by following in the authors' footsteps, using the same dataset and normalisation methods. One of the research questions I had from the start was whether I would be able to replicate the results reported in their paper, and that was largely the driving force at the beginning of my journey. I then continued with further experiments, mainly changing the model architecture and adjusting the type of data that is fed to the models. The results obtained from these experiments were then compared to the ones reported by Partanen et al. (2019).

[1] `https://github.com/mikahama/murre`

1.2 Research Questions

As was already mentioned, my first task was to see whether I could replicate the results obtained by Partanen et al. (2019). Afterwards, I used various different model architectures and input data types to conduct further experiments, finally comparing the end results. My aim was to see if these changes would have an effect on the prediction outputs generated by the models, and if so, what kind of an effect: whether the different methods or input strategies would worsen the results or improve them.

Thus, the research questions of this thesis are the following:

1. How does the model architecture or the input data influence dialect normalisation?
2. What can be done to improve Finnish dialect normalisation?

1.3 Structure of the Thesis

After the introductory chapter, the next section will focus on some of the related work that has been done on the subject of text normalisation. I will first go through the different types of text normalisation, or rather the various domains on which normalisation methods are most often applied: historical text, spoken language, and user-generated content. After this, I will also cover the different methods commonly used for text normalisation. In the following two sections, I will describe the dataset and the methods I used to perform the experiments. Here, I will also discuss the different metrics that I used in the evaluation process. After that, I will discuss the results and findings. Finally, the last section will contain the conclusions drawn from this thesis.

2 Background & Related Work

2.1 Text Normalisation Applications

2.1.1 Historical Text

Normalisation of historical texts, or *modernisation*, is the task of converting historical word-forms to their contemporary equivalents. Digitisation projects are happening all over the world, allowing the utilisation of natural language processing tools and techniques on large collections of historical documents. However, natural language processing of historical texts faces several challenges, a major one of which is spelling variation (Piotrowski, 2012). Texts that predate standardised orthography often differ significantly from modern ones in their spelling conventions; moreover, even texts written in the same time period and by the same author can contain a wide variety of different spelling choices. While there is a certain degree of variation in modern languages as well—for example, the so-called American and British spellings (*color* vs. *colour*, *gray* vs. *grey*)—the variation found in spelling conventions is much rarer than it used to be (Piotrowski, 2012). For example, in historical texts, the English word *their* can be found written in several different forms, such as *thir*, *thayr*, *theaire*, and *þere*, to mention only a few possible variations (Bollmann, 2019).

Several methods have been used in automatic text normalisation of historical texts. In a large-scale comparison overview, Bollmann (2019) separates the different contemporary approaches into five categories: substitution lists, rule-based methods, distance-based methods, statistical models, and neural models. The different normalisation methods will be discussed in more detail in Section 2.2, excluding the simplest technique, which will be covered here: substitution lists, also known as lexical substitution or wordlist mapping. This is a conceptually very simple approach that uses a precompiled dictionary to map historical word-forms to their normalised forms. While this method might not be enough by itself, it can however be useful as a component in a more complex normalisation system (Bollmann, 2019). Such approach is utilised in the implementation of the Variant Detector (VARD) tool, trained on Early Modern English data from the 16th–19th century (Rayson et al., 2005; Baron and Rayson, 2008), and the Norma tool, originally developed and evaluated on Early New High German (Bollmann, 2012). In addition to substitution lists, Norma uses both rule-based and distance-based algorithms, and VARD 2 includes a rule-based component. Rule-based and distance-based approaches will be explained in Section 2.2.1.

2.1.2 Spoken Language

Language variation was briefly mentioned in Section 1, and one form of language variation that is probably familiar to most people is the use of dialects. While clear definitions can sometimes be challenging—what constitutes a language, where is the line between a language and a dialect—dialects are usually seen as mutually intelligible varieties of one particular language. Distinguishing features may occur on any level of language: phonology, morphology, syntax, or lexicon. When the observable distinctions between dialects are mainly realised as differences in pronunciation, the term *accent* may also be used instead. In any case, a dialect is a language variety that is used by the members of a particular group of speakers: this can be a regional variety, such as Liverpool English, or a variety that is otherwise characteristic of a specific group of people, such as African-American Vernacular English. A dialect that is predominantly used by a specific ethnic group is also known as an ethnolect, while a language variety associated with a particular social group—for example, an age group or a social class—can also be called a sociolect.

While sometimes the differences between dialects are relatively small or even barely noticeable, sometimes these differences are substantial enough to affect intelligibility. However, humans are remarkably skilled when it comes to languages. Language production is a natural and effortless task for most of us, but we also excel at understanding the language produced by others. Even if a person is speaking in a way that is very different from our own—for example, using unfamiliar dialectal words or unusual pronunciation—we may very well be able to understand their speech without problems. The speech situation in itself already gives the interlocutors context that helps in language comprehension: not only the topic, but also the physical location in which the conversation takes place, the social roles of the participants, and so on. For example, the sentence *The thing is under the thing* would make no sense without context, but if the topic is a missing item and the location is a sparsely furnished room—perhaps the speaker is even gesturing towards a table while they utter the sentence—the listener is quick to deduce that the item they are looking for is probably found under the table. Speech errors are also something that we all make from time to time, and these errors are generally not a problem for the conversation partners. In addition to various contextual clues, all humans have a wide array of general information available to us, both linguistic and non-linguistic; telling us whether an utterance is grammatical according to the rules of the language that is being spoken, or whether its contents are sensible based on our general understanding of the world.

Humans might be proficient in natural language understanding, but what about machines? What about natural language processing? Let us imagine an application that accepts spoken human commands as input. The application in question has been trained on large amounts of speech data, but all or most of this data is gathered from speakers of a particular dialect. What happens

when someone who speaks the same language, but a decidedly different dialect, tries to use this application? If they speak with a distinct accent that the system has never before seen in its training data, their input might be unintelligible for the system. The dialect speaker might then choose to alter their own pronunciation to make it more understandable for the application, but this can lead to frustration and a poor user experience. Furthermore, if the purpose of the tool is to convert spoken language into written text, the system should be able to produce text that is intelligible even to people who speak the language differently: in most cases, this means that the output text is in standard language, which is often deemed the most "neutral" option.

Text-to-speech (TTS) and speech-to-text (STT) applications typically require the text to be in a standardised form, regardless of the direction of the conversion. A text-to-speech system that converts written language into synthesised human speech requires the input to be in a form that the system understands; while a speech-to-text system—also called automatic speech recognition (ASR)—typically produces text in standardised writing as its output. As the natural spoken language can be very different from the written standard, text normalisation is needed when speech is converted to text. While normalising dialectal variants is a major part of ASR, there are also other aspects to consider: for example, dates and numbers. The spoken English utterance *thirty-nine dollars and ninety-nine cents* would typically not be written as such, but rather the segment would be converted to a more conventional string of symbols, *$39.99*. Similarly, the Finnish phrase *kolmas toukokuuta* 'May the third' would probably be transformed into *3.5.* in the written form.

Swiss German has been of special interest in the context of text normalisation: regional variation is strong, and while the different dialects are widely used in spoken everyday communication, there is no standard orthography (Samardžić et al., 2015; Nigmatulina et al., 2020). For this reason, there is a growing interest for Swiss German in ASR technology as well. For example, a choice has to be made as to how non-standardised input language is represented in the system's output. Nigmatulina et al. (2020) explore two different approaches: dialectal writing and normalised writing. In their experiments, the dialectal writing composes of approximate phonemic transcriptions that are meant to roughly correspond to the acoustic signal, whereas the normalised writing resembles standard German. The latter provides some consistency, but the correspondence between graphemes and the acoustic signal is less apparent. The situation with two different transcription strategies is to some extent similar to the Finnish language dataset used in this thesis: the normalised annotation is close to Standard Finnish, while the semi-narrow transcription resembles the original spoken samples. The dataset will be discussed in more detail in Section 3.1.

2.1.3 User-Generated Content

The ability to process and understand non-standard language is required in a variety of natural language processing applications and tools. Such an application does not even need to be advanced enough to use speech data: the same problems described in Section 2.1.2 persist with written language. Even if a language does have a standardised orthography, its speakers might not be aware of all the writing conventions or choose to ignore those altogether. Again, context matters: one would not be overly concerned about correct spelling and punctuation when scribbling a quick note for a family member, but they might spend a considerable amount of time writing a formal letter. These days, social media is also a major platform on which people interact and communicate with each other. This particular platform has its own characteristics, one of which is the informal and casual nature of the language people tend to use. Thus, social media text can be drastically different when compared to texts from other domains.

User-generated content (UGC) is any type of content that has been created by users on online platforms. This includes not only text but also images, videos, and audio. One typical form to produce text is via written computer-mediated communication (CMC), including formats such as emails, instant messaging (IM), internet relay chat (IRC), and online message boards (Thurlow et al., 2004). Recently, social media has grown to be a prominent part of how people communicate online, and as such it has also become an important application domain for natural language processing. Text typically seen on social media services, such as Twitter, contains a plethora of linguistic challenges that are characteristic to the domain: this includes non-standard and unconventional punctuation, spelling, and capitalisation, and even its own distinct vocabulary and syntax (Eisenstein, 2013b). For example, *I love you* becomes *i luv u*, and *I know right?* could be written as *ikr???*. Data containing large amounts of text like this can be problematic for natural language processing tools that are trained solely on standard language, so often the text goes through a normalisation step first.

The language that people use online does not need to be as extreme as these examples just given. As was mentioned in Section 1.1, Finnish is an example of a language that can express much of the spoken regional variation through writing. For example, the word *näen* /ˈnæe̯n/ ‘I see’ is mostly pronounced in colloquial Finnish as /ˈnæːn/, and thus it could in informal context be written as *nään*. Similarly, *menetkö (sinä)* ‘will you go’ could very well be shortened to *meetkö sä* or even *meeksä*. Finnish is not an unique language in this sense: even English, despite not having as straightforward a relationship between its phonology and orthography as Finnish does, can express features of spoken language through writing. For example, Eisenstein (2013a) finds that consonant cluster reduction, a phonological variable common in several English dialects, can also be observed on Twitter data, and that its usage is sensitive to the surrounding linguistic context, much like in spoken language.

2.2 Text Normalisation Methods

2.2.1 Rule-based & Distance-based

Lexical substitution was already mentioned in Section 2.1.1 as the conceptually simplest form of text normalisation. After substitution lists, rule-based methods are already a step further into the realm of more complex systems: instead of simply consulting a wordlist, rule-based approaches aim to find patterns and regularities between the differently spelled variants by following specifically tailored replacement rules (Bollmann, 2019). Distance-based methods, on the other hand, utilise edit distance measures like the Levenshtein distance (Levenshtein, 1966) to determine the relative differences between spelling variants. Both rule-based and distance-based methods are also commonly found in an information retrieval (IR) context (Bollmann, 2019).

In the context of historical text normalisation, finite-state transducers have often been successfully utilised when approaching the problem with a rule-based solution. For example, Porta et al. (2013) use weighted finite-state transducers to normalise old Spanish from the Middle Ages. Their system uses an edit transducer and a set of rules expressing the phonetic and phonological sound changes in Spanish to map the historical word-forms to their modern variants, and the approach achieves higher accuracy when compared to the baseline Levenshtein distance model. Similarly, Etxeberria et al. (2016) use weighted finite-state transducers and language models to normalise Spanish, Basque, and Slovene.

Pettersson et al. (2013a) use a weighted Levenshtein-based algorithm to normalise historical Swedish. Their approach includes edit operations for multiple characters in addition to single edit operations, and they also use compound splitting to allow the system to normalise parts of a compound word individually. While the traditional Levenshtein distance computation gives all edit operations a cost of one, Pettersson et al. (2013a) assign lower weights for frequently occurring edits in the training corpus, improving precision. Their system also outperforms a previously implemented approach that uses a set of hand-written normalisation rules.

Rule-based and distance-based methods can also be combined, utilising the strengths of both approaches. Adesam et al. (2012) use iterated Levenshtein distance alignment to align spelling variants for old Swedish, and these character-aligned spelling variants are then used to derive substitution rule sets. They experiment with three different rule sets: one with manually crafted replacement rules, and two with automatically extracted substitutions. The hand-crafted rules have higher precision, though the automatically extracted n-gram rules find more matches overall.

2.2.2 Statistical Machine Translation

More recently, statistical machine translation (SMT) methods, and especially character-based statistical machine translation (CSMT), have often been applied successfully to text normalisation tasks. Indeed, text normalisation can be seen as a translation task, in that the word-forms have to be transformed, or "translated", into their standardised forms. In a character-based approach, instead of simply treating a sentence as a sequence of tokens, every token is treated as a sequence of characters. For this reason, the method is particularly effective at capturing intra-word transformations, making it a suitable tool for text normalisation (Lusetti et al., 2018). It can also be highly useful when working with low-resource languages: the vocabulary for a CSMT system is small, just the set of characters used, so the model can learn sequence patterns effectively even when available training data is scarce. Another advantage of character-based models is their greater adaptability to out-of-vocabulary (OOV) words, since the transformation patterns that the model has recognised and observed for certain strings of characters can also be easily applied to unknown words (Lusetti et al., 2018; Hämäläinen et al., 2020b). The underlying idea behind the SMT approach can be seen as a *noisy channel model*, which has also been widely used in spelling correction, machine translation, and speech recognition (Lusetti et al., 2018; Bollmann, 2018; Bollmann, 2019). Here, the idea is that the source word is in some way scrambled or jumbled, and the system has to fix it. The desired end results is that the processed source token is identical, or as close as possible, to the corresponding target word.

In historical text normalisation, character-based statistical machine translation methods have been shown to achieve good results for a variety of languages: for example, Spanish (Sánchez-Martínez et al., 2013), Icelandic and Swedish (Pettersson et al., 2013b), and Slovene (Scherrer and Erjavec, 2013; Ljubešić et al., 2016). Sánchez-Martínez et al. (2013) also note that the statistical machine translation approach works well for modernisation purposes, because historical text normalisation is essentially an asynchronous, non-deterministic, and monotonous process. *Asynchronous* means that normalisation cannot be done on a simple grapheme-by-grapheme basis: a digraph may translate to a monograph, and the other way around. *Non-deterministic* means that the replacement rules are not set in stone and can in fact differ even in the same context: in some cases, the old spelling of a word should be preserved in the normalised output, while in others it should be translated into a different word-form. *Monotonous* means that the character replacement rules do not generally tend to show a long range dependence on the context (Sánchez-Martínez et al., 2013).

Pettersson et al. (2013b) apply the CSMT approach successfully to historical texts in both Icelandic and Swedish. They note that the training process of character-based SMT models used for text normalisation tasks usually involves four basic steps: word alignment, character alignment, language modelling, and parameter tuning. Word alignment essentially builds the parallel

training data for the translation models, creating word pairs from all corresponding words in the data. The next step, character alignment, aligns all characters for the entire corpus after the word pairs have been created. Once alignment is done both on word level and character level, the next step is character-based language modelling. This is done by training language models on monolingual data in the target language. The final task, parameter tuning, is done after the language models have been trained, and it involves tuning these models on some development data to ensure the best possible performance and results. Pettersson et al. (2013b) further note that the text normalisation step is easily worthwhile when the aim is to apply modern NLP tools such as taggers and parsers to analyse historical text, as the results are consistently better with normalised text. They also find that even a small amount of word-aligned training data is enough to achieve good results, and if no parallel data is available for training, automatic sentence alignment methods may be used to create the necessary training data (Pettersson et al., 2013b).

Scherrer and Erjavec (2013) describe two different experiments in their paper about normalising historical Slovene: one with a supervised setup, and one with unsupervised. The supervised model is trained on bilingual data, as is usually the case with SMT systems. In this case, the training data consists of word pairs composed of a historical word-form and its modern variant. The unsupervised approach, on the other hand, uses monolingual data, imitating a case where a bilingual training lexicon containing historical words and their associated contemporary word-forms is not available for a given language. In this setup, a bootstrapping step is implemented: each historical word is paired with a modern word-form that is deemed the most similar to it. The similarity between the word pairs is computed using the BI-SIM measure (Kondrak and Dorr, 2004), which captures the graphemic similarity between words by using character bi-grams (Scherrer and Erjavec, 2013). The supervised approach yields consistently better results when compared to the unsupervised setup, as can be expected, but the differences are less notable with more contemporary data. The lexicons of historical Slovene used in the experiments contain material from various time periods: the earliest data is from the second half of the 18th century, while the newest material is from the latter half of the 19th century. The more recent the time period is, the closer the language in the corpora is to the contemporary language variant: consequently, there are fewer character replacements for the modernisation model to learn, and the results are usually better.

Ljubešić et al. (2016) apply the CSMT approach successfully to both historical and contemporary Slovene. In addition to historical datasets, they use a social media dataset consisting of Twitter tweets to normalise contemporary non-standard user-generated content. They note that while tweets typically contain a fair amount of non-standard and dialectal language, there is also a notable portion of tweets that consist of completely or mostly standard language. For this reason, Ljubešić et al. (2016) classify the texts automatically into three different levels according

to technical and linguistic standardness (Ljubešić et al., 2015). Technical standardness contains features such as the use of punctuation, capitalisation, and the presence of spelling mistakes, while linguistic standardness is more related to the linguistic knowledge of the writer. Authors always make decisions—both conscious and unconscious—regarding the linguistic standardness of their texts: this involves features such as spelling, morphology, and word order (Ljubešić et al., 2015). Ljubešić et al. (2016) prepare two datasets according to the automatically calculated standardness scores of the tweets: the so-called easy dataset, consisting of tweets that are both technically and linguistically standard, and the hard dataset, consisting of technically standard but linguistically non-standard samples. In their experiments, they compare both token-level and sentence-level approaches and find that a sentence-level system may work better for datasets with high token-level ambiguity, of which their hard tweet dataset is an example. The authors also note that their best performing model achieves good results for both historical and contemporary datasets, so there is no need for two different systems for modernisation of historical texts and text normalisation of user-generated content (Ljubešić et al., 2016).

Pettersson (2016) compares four different normalisation methods for historical texts: a rule-based approach, a Levenshtein-based approach, a memory-based approach, and an SMT-based approach. The rule-based method was developed for Swedish and, consequently, also evaluated on Swedish only. The other approaches were all evaluated on five languages: English, German, Hungarian, Icelandic, and Swedish. The multilingual evaluation also includes one more approach: a combination of the Levenshtein-based and the memory-based methods. Pettersson (2016) notes that the character-based SMT approach generally performed the best, though the overall results of the different normalisation methods varied between languages. In the case of Icelandic, for example, the Levenshtein-based normalisation method combined with a memory achieves the best results. All the methods used in the multilingual setting are language-independent and thus generally applicable to any language, even in cases where the amount of available training data is scarce (Pettersson, 2016).

Character-based SMT has also been used for dialectal Swiss German (Samardžić et al., 2015). Swiss German dialects are commonly spoken in the Northeastern parts of Switzerland, where the sociolinguistic setting is a typical example of diglossia: Swiss German is the most common and widely used vernacular language variety of the speech community at large, but the language of choice in written contexts has traditionally been and still usually is Standard German. These days, however, Swiss German is used increasingly in written contexts as well, especially in computer-mediated communication (Lusetti et al., 2018). Consequently, this makes Swiss German a language of interest, not only in dialectological studies, but also in the text normalisation context: written Swiss German has no standardised orthography, so applying normalisation on dialectal texts can be an especially challenging task. Moreover, the natural language process-

ing tools developed for Standard German do not generally work particularly well with Swiss German, as the differences between the dialectal variety and the standard language are quite notable (Samardžić et al., 2015). The differences are not limited to lexicon and pronunciation either, as there is considerable variation in both morphology and syntax as well. Occasional intra-speaker variation can also be found, which causes further inconsistency in written Swiss German. Samardžić et al. (2015) identify two separate challenges: lexical mismatches and word boundaries. An example of a lexical mismatch is the Swiss German word *öpper* 'someone': its semantic counterpart in Standard German would be *jemand*, though there is no etymological relation between the two words. Such words can be normalised by forming pairs between the semantically motivated variants, even though the similarity between the word-forms is lacking on the surface level. Alternatively, the dialectal variant can be mapped to a historically common form that does not exist in the contemporary standard language: in this case, *etwer*. Word boundaries also often behave differently in written Swiss German, and the general conventions used in Standard German might not apply. For example, the phrase *hettemers*, written as one word, corresponds to *hätten wir es* in Standard German. In this case, a choice has to be made as to whether to keep the standard word boundaries or to follow the intuition of native speakers (Samardžić et al., 2015).

De Clercq et al. (2013) use statistical machine translation methods to normalise Dutch user-generated content. Their training data contains text messages, tweets, and message board posts from a social networking site. Such text material is typically particularly noisy, including misspellings, unconventional capitalisation, and emoticons. Other characteristics commonly observed in the mentioned domains include the conscious omission of words or characters, unconventional spelling at the lexical level—for example, spelling *gelijk* 'like, similar' as *lyk*—and the frequent and highly productive use of abbreviations and acronyms, such as *LOL* or *smh* in English. In the case of Dutch, one feature of written informal speech specific to the language is the concatenation of tokens, which causes clitics and pronouns to disappear on the surface: for example, writing *khou* instead of *ik hou* 'I love', or *edde* instead of *heb je* 'do you have'. De Clercq et al. (2013) focus on text messages in their experiments, and they process the data through a cascaded SMT system consisting of two different modules: first, a token-based SMT model, and a character-based SMT module following after that. Both unigram and bigram models were tested for the character-based approach. The authors find that the cascaded approach yields the best results when compared to simple token-based, unigram-based, and bigram-based systems.

2.2.3 Neural Machine Translation

Following SMT, neural machine translation (NMT) has reached state-of-the-art results in many natural language processing tasks. Similarly to CSMT, character-based neural machine translation (CNMT) can be used to effectively capture the variation in word-forms. Partanen et al. (2019) use character-based NMT in a text normalisation task to normalise Finnish dialects, finding that the bidirectional recurrent neural network (BRNN) architecture achieves good results. Moreover, they experiment with three different normalisation levels—word-level, chunk-level, and sentence-level—and find that the models trained on chunked data yield the best results. I aim to replicate their approach in my own experiments, so the dataset and the methods they use are discussed in greater detail in Section 3.

Similarly, Hämäläinen et al. (2020a) use character-based neural machine translation methods to normalise Finland Swedish dialects. The Swedish dialects spoken in Finland differ considerably from the Swedish language variants spoken in Sweden, and the variation between dialects is great even within the borders of Finland, which can be attributed to the wide geographical span of the Swedish speaking communities and Finland's low popular density. Hämäläinen et al. (2020a) use input data consisting of chunks of varying lengths to train the models, similar to the approach used in Partanen et al. (2019). However, they find that the word-level model achieves better results than the models trained on longer segments, which is different from the earlier conclusions presented by Partanen et al. (2019). One possible cause is the difference in training data sizes: with smaller training set sizes, the model does not benefit from a larger context, and too much context can in fact have a negative effect on model performance (Hämäläinen et al., 2020a).

Character-based NMT methods have also been applied to historical text normalisation. Bollmann and Søgaard (2016) use a bi-directional long short-term memory (BiLSTM) network to normalise historical German, and they also experiment with a multi-task learning (MTL) setup. In multi-task learning, a model is trained on one or more auxiliary tasks in addition to its primary task to improve its overall performance: in essence, the model is trained on two datasets in parallel. Bollmann and Søgaard (2016) define spelling normalisation within a given historical text as their main task, while normalisation within related domains—for example, texts from the same time period—are considered the model's auxiliary tasks. The results are compared to those obtained by two baseline methods, the Norma tool (Bollmann, 2012) and a tagger based on conditional random fields (CRFs). Since the MTL approach cannot be applied to the other methods, Norma and the CRF-based tagger are also evaluated with augmented data, so a random sample of 10k examples from all texts is added to the training set. However, the results show that the additional normalisation data generally does not help the Norma tool and the CRF tagger: on the contrary, it may even lower the prediction accuracy. The BiLSTM model con-

sistently outperforms the two baseline methods, and while the accuracy of the model improves with multi-task learning for some texts, it decreases for others.

Bollmann et al. (2017) also apply a multi-task learning approach to a BiLSTM model to normalise historical German, but they define their auxiliary task as a sequence prediction task where characters are mapped to phonemes. This is done by using a simple grapheme-to-phoneme dictionary as auxiliary data, and the model has to learn to predict the correct sequence of phonemes based on an input sequence of graphemes. The prediction results show that the base BiLSTM model benefits from beam search, lexical filtering, and an attention mechanism, but adding attention to the model trained with multi-task learning actually lowers its accuracy. Bollmann et al. (2017) observe similarities between multi-task learning and attention mechanisms, and they further hypothesise that the multi-task learning already takes care of focusing attention during the training process, which is why combining the approach with a separate attention mechanism hurts the final score. Bollmann et al. (2018) further observe that multi-task learning is especially beneficial to text normalisation when the target data is very scarce.

Domingo and Casacuberta (2018) compare SMT and NMT methods for modernisation purposes for historical Spanish and Slovene, finding that the SMT approach yields better results in their experiments. They also compare token-based and character-based approaches, and they conclude that the character-based methods outperform the the word-based models, especially with smaller corpora. This is to be expected, considering the fact that most changes occur at a character level in text normalisation tasks. Interestingly, even a simple statistical dictionary approach can yield satisfactory results in the absence of more advanced text normalisation methods: here, the dictionary is built by looking at the frequency of the spelling changes inspected in the training corpora. Going through a document, each word is checked for a translated counterpart in the dictionary. Then, if a match is found, the word is substituted with its corresponding translation; if there are no matches, the word is left as is. The statistical dictionary approach manages to achieve considerable improvements when compared to the baseline, even if the results are not quite as good as those obtained by the more advanced SMT and NMT models (Domingo and Casacuberta, 2018).

Çolakoğlu et al. (2019) also compare NMT and SMT methods to normalise non-canonical written Turkish and find that their SMT model performs better than the NMT model. Their machine translation approach also outperforms the cascaded approach used by Torunoğlu and Eryiğit (2014), formerly considered the state-of-the-art approach for Turkish text normalisation. Like Finnish, Turkish is a morphologically rich and highly agglutinative language, and the challenges in text normalisation are therefore closely related. Another example of a language with morphological inflection is Japanese. In the case of Japanese, there is the additional challenge of the characteristics of the writing system: written Japanese lacks explicit word boundaries, so

word segmentation is a non-trivial task in Japanese language processing. Ikeda et al. (2016) use the encoder-decoder neural network architecture to normalise Japanese social media texts, comparing their model to two baseline methods: a rule-based system, and an approach based on character-level conditional random fields (CRFs). The encoder-decoder model performs poorly on small corpora, but the results improve when the amount of training data increases. Saito et al. (2017) aim to tackle the problem of insufficient training data sizes with data augmentation: they extract morphological conversion patterns from a small dataset, and these patterns are then used to generate augmented data.

Whereas Domingo and Casacuberta (2018) and Çolakoğlu et al. (2019) find that the SMT approach yields better results in their text normalisation experiments, Lusetti et al. (2018) state that the encoder-decoder methods they use to normalise Swiss German text messages outperform their SMT model. They also report that the normalisation errors encountered in the predicted outputs produced by the models are rather similar in both approaches. For example, both the SMT and NMT models struggle with ambiguity, such as when a source token has more than one possible normalisation form. Irregularities and non-standard vowel reduplication often encountered in casual, informal text—for example, writing *bitteeeee* instead of *bitte*—are also challenging for the models. One of the corpora that Lusetti et al. (2018) use also contains a fair amount of emojis, but both models process and normalise these without problems. This is a positive finding, as it means that there is no need for an extra preprocessing step consisting of removing emojis from the source data before feeding it to the models.

Similarly, Hämäläinen et al. (2018) compare different normalisation methods for historical English and report that the NMT approach outperforms the other methods in their experiments. They evaluate three different approaches: a rule-based finite-state transducer, a distance-based approach, SMT, and NMT. In the rule-based approach, the replacement rules for character sequences are based on VARD 2 (Baron and Rayson, 2008), and a finite-state transducer is built by using HFST (Lindén et al., 2013). The edit-distance approach is based on the Levenshtein distance, and the list of normalisation candidates is filtered down by context and estimated pronunciation. Both the SMT and NMT methods are character-based. Comparing the results of the individual approaches, the NMT method is the one that yields the best results, but Hämäläinen et al. (2018) also experiment with combining the different approaches to harness the individual strengths of the various methods. The evaluation of this combined approach is done by picking the best normalisation from the list of candidates suggested by all the different models. First, possible non-words produced by the machine translation approaches are filtered out by attempting to map the outputs to the Oxford English Dictionary entries. Next, the best candidate is chosen by using one of three alternative methods: voting, weighted voting, and a Markov chain. The simple voting mechanism picks the normalised output that is most often suggested by the different methods: if three approaches produce the exact same predicted normalisation and one

approach results in a different output, the normalisation candidate with three votes behind it wins. The weighted approach is otherwise the same, but it is weighted according to the reported accuracies of the individual methods to ensure that the normalisation methods with the highest accuracies have more sway over the voting results. The third method, the Markov chain approach, picks the most likely normalisation based on context, meaning that it simply looks at the probability at which a given token follows another in a sequence. Comparing the accuracies of the combined approach resulting from the different decision-making methods, it seems that the weighted voting performs the best. Hämäläinen et al. (2018) state that there is still room for improvement, but since all of the methods do have their own strengths when it comes to text normalisation, it seems like a worthwhile task to try to combine them.

Tang et al. (2018) compare a baseline SMT system by Pettersson et al. (2014) and several different NMT architectures for historial text normalisation. Similar to Pettersson et al. (2014), they conduct the experiments on five different languages: English, German, Hungarian, Icelandic, and Swedish. The NMT architectures are the following: vanilla recurrent neural network (RNN), long short-term memory (LSTM), gated recurrent unit (GRU), and Transformer. The authors also compare different attention mechanisms: no attention and soft attention for the RNN-based models, and the multi-headed self-attention mechanism of the Transformer model. In the end, they conclude that the NMT models consistently outperform the SMT approach in terms of character error rate (CER). However, they also note that NMT models are more likely to generate incorrect normalisations of unchanged spellings: that is, the models predict a change in a character sequence, when the historical spelling should actually be identical to its modern counterpart. For this reason, Tang et al. (2018) propose a hybrid method using both NMT-based methods and a dictionary-based method. They also find that vanilla RNNs are competitive with the more advanced GRU and LSTM architectures in some cases. Transformer models, having very fine-grained self-attention and multi-headed attention mechanisms, generally perform better than RNN-based models with soft attention, but only when the amount of training data is large enough. The authors also compare character-level models to subword-level models with LSTMs. The subword units are generated with a byte pair encoding (BPE) algorithm with different vocabulary sizes, and the results show that subword-level models can outperform the character-level models when the BPE vocabulary is small enough. Large training sets cause data sparsity, so setting the BPE vocabulary size too high yields worse accuracy results when compared to the character-based models.

Bollmann (2018) compares the encoder-decoder architecture to two different systems for historical text normalisation: Norma (Bollmann, 2012) and cSMTiser (Ljubešić et al., 2016). The Norma tool is a combination of wordlist mapping, rule-based approaches, and distance-based approaches, while cSMTiser is a tool that uses established statistical machine translation software, such as Moses (Koehn et al., 2007). The comparison is done on eight different languages:

English, German, Hungarian, Icelandic, Portuguese, Slovene, Spanish, and Swedish. Both word accuracy and character error rate (CER) are measured in evaluation. The results show that the character-based SMT approach outperforms the NMT methods on all datasets except the German corpus, even when the NMT models are extensively tuned for the normalisation task. Closer inspection reveals that the higher word accuracy of the CSMT approach is mostly due to its better performance on OOV words: for this reason, Bollmann (2018) suggests that any historical text normalisation system could benefit from a wordlist mapping component.

The Transformer (Vaswani et al., 2017) neural network architecture has achieved state-of-the-art results in various natural language processing tasks. However, while the model architecture has been shown to outperform RNN-based encoder-decoder and sequence-to-sequence (Seq2Seq) models in many word-level tasks, it still often falls behind when the task requires a character-based approach, such as text normalisation. Wu et al. (2021) show that batch size is a crucial hyperparameter in the performance of a character-based Transformer, and their Transformer model manages to outperform the RNN-based baseline methods in their experiments on four different character-level tasks: grapheme-to-phoneme conversion, morphological inflection, transliteration, and historical text normalisation. Their approach is also utilised in my own experiments, and I will go through the parameters in more detail in Section 3.2.1.

3 Methods

3.1 Data

The data used in this study is the Samples of Spoken Finnish corpus[2] (Institute for the Languages of Finland, 2014). It is based on a series of booklets that were published between 1978 and 2000, each booklet covering about two hours' worth of dialectal speech in transcribed form. The modern digital corpus covers 50 Finnish rural municipalities, each of which includes two dialect samples. The whole corpus contains 696,376 transcribed words, of which 684,977 include a normalised word-form in Standard Finnish. The transcriptions are done in a semi-narrow manner, containing non-standard annotations that aim to capture the phonetic and prosodic features of dialectal speech (Hämäläinen et al., 2020b). The data is already tokenised, and the transcribed texts are aligned with their normalised counterparts on word level, forming a parallel dataset: this makes the corpus well-suited for text normalisation tasks.

The transcribed data has been normalised according to a set of guidelines made for this specific dataset and task (Vilkuna, 2014). While the normalisation process is fairly simple and straightforward for many tokens—for example, changing the dialectal form *kohvi* 'coffee' to the Standard Finnish variant *kahvi*—sometimes the standardisation task is more complex. For instance, the correspondence is not always one-to-one: the dialectal word-form *ennenkö* 'before; prior to' translates into two separate tokens, *ennen kuin*, in written Standard Finnish. Sometimes the exact meaning behind word-forms can be ambiguous, and linguistic features such as grammatical case or tense are not immediately obvious. For example, Standard Finnish makes the distinction between present tense and past tense in *sanovat* / *sanoivat* 'they say / they said', but a dialectal variant may simply use one word-form, *sanovat*, to include both. In this case, grammatical tense is inferred from other parts of the utterance. Additionally, the inventory of inflectional and derivational suffixes in Finnish is fairly large, but spoken language and some dialects typically cut the suffix short. This is especially the case with common and frequently used words, and the feature is characteristic of certain dialects, most notably the Southwest Finnish dialects. This habit of dropping the final sounds from the end of the word blurs the distinction between word-forms and causes further ambiguity: *meil* can mean either *meillä* 'on/at us' or *meille* 'to/for us', and context is needed to interpret the intended meaning (Vilkuna, 2014).

The Finnish language also makes use of clitics, which are syntactically independent but phonologically dependent morphemes. Their usage, however, may differ between dialects and the standard language. For example, according to the guidelines, the common dialectal variant type *onks* is systematically normalised as *onkos* in the data (Vilkuna, 2014). The simple neutral

[2] `http://urn.fi/urn:nbn:fi:lb-201407141`

version of this word-form would be *onko*, which is a combination of the third-person singular present form of the verb *olla* 'to be' and the interrogative clitic *-ko*. The dialectal variant and its chosen standardised version, however, also ostensibly contain the clitic *-s*. One of the functions of this clitic is to soften an order or a question, and it can also serve as a way to make the tone of the whole utterance more familiar. In dialectal usage, however, the final *-s* does not necessarily carry any meaning typical of the clitic: rather, the utterance can be a completely neutral question, despite the fact that the additional clitic is present on the surface. Here, a decision has been made to always systematically include the final clitic in the normalised word-form, because trying to infer the intended tone and meaning from the utterances can be a challenging and also rather sensitive task (Vilkuna, 2014).

To train the models, the sentences in the data are randomly sorted and then split into training, test, and validation sets. The training set contains 70% of the original sentences, while the test and validation sets are 15% each: this follows the same split that was used by Partanen et al. (2019). The models are trained on character level, so words are always split into characters by white spaces, both on the source side and the target side. For example, the word *koira* 'dog' becomes *k o i r a*. If the input contains more than one token, word boundaries are indicated with an underscore character (_): for example, *k o i r a _ j u o k s e e _ .* 'dog runs'. As the data is already tokenised, punctuation marks are also treated as individual tokens. Sentence boundaries are marked with an artificially empty line, the special tag <*BLANK*>: the main purpose of this is to help with constructing the sentences back at the original places before evaluation. A special tag is used, mainly because in my experiments a simple blank line proved to be somewhat unreliable as a sentence boundary marker, resulting in poor model performance. Similarly, converting all text to lowercase seemed to affect the results negatively. Therefore, letter case is left as is: in the data, proper nouns are capitalised according to the writing conventions of Standard Finnish, so named entities such as the names of people and places start with an uppercase character. The first word of a sentence is not capitalised, except in the case of interviewer utterances: this is an interesting convention used in the corpus, highlighting the two different roles of the original recorded speech situation. One could argue that only the sentences with a role labelled as an interviewee should be considered in the training data, since these utterances are the point of interest in this study. However, I have opted to include the interviewer utterances as well, as these are also examples of spoken Finnish. The script I used to preprocess the data is written in the Python programming language.

I also experiment with different input strategies, meaning that the models are trained by feeding them either one whole sentence or a subset of a sentence at a time. The sentence-level models have an input of one sentence at a time, and the tokens are separated by underscores as previously described. The word-level models receive single tokens as input, so no word boundary markers are needed, but the characters are still separated by white spaces. On chunk level, I

experiment with different setups: I replicate the 3-chunk approach that Partanen et al. (2019) used in their experiments, but I additionally train a set of models with chunks of 2, 4, and 5, to see if changing the chunk size yields different results. An example of the source and target input tokens in chunks of 3 can be seen in Table 1. I also experiment with a sliding/rolling window approach: instead of splitting the sentences into separate chunks, I use a sliding window to determine the subsets. Therefore, the earlier example of the short sentence consisting of three tokens, *k o i r a _ j u o k s e e _ .*, with a chunk size of 2 would yield the following overlapping subsets: *k o i r a _ j u o k s e e* ; *j u o k s e e _ .*. A more extensive example can be seen in Table 2. The downside of this approach is that it causes the training set sizes to grow considerably, since every token is repeated in the input data. The original training data set contains 35k parallel rows on sentence level, and the amounts grow incrementally for the various subsets: the approximate training set sizes for different setups can be seen in Table 3.

3.2 Models

3.2.1 Neural Network Models

I use the OpenNMT toolkit (Klein et al., 2017) to train the neural network models. Two of the architectures are similar to what was used in the experiments conducted by Partanen et al. (2019); namely, the bidirectional recurrent neural network (BRNN) model and the Transformer model. In addition to these models, I perform experiments with a simple unidirectional RNN model and a convolutional neural network model. Each model will be trained on six different types of sequences: words, sentences, and differently sized chunks (2, 3, 4, 5). Moreover, the RNN and BRNN models will be trained on two different sets of data: the transcribed version that includes detailed linguistic transcription markings, and the corresponding simplified data that lacks these transcriptions. An example of the transcribed source input tokens and the aligned simplified tokens can be seen in Table 4. My initial hypothesis is that the models trained with the more detailed, transcribed data will consistently achieve better results, but I am interested to see how notable the differences are. The RNN, BRNN, and Transformer models will further be trained on data that consists of chunks of varying lengths with a rolling/sliding window, and these results will be compared to the results obtained from the models trained on separate chunks without the overlapping window. It is possible that the models with a rolling window approach outperform the regular chunk-level models, since they have more information on the context around the target tokens. In the preliminary experiments, the convolutional models generally seemed to perform rather poorly, which is why no further experiments with the sliding window approach are conducted with this particular model architecture. All models are trained for the default 100,000 training steps.

Source	Target
s i t ä _ e i _ m a k s a n	s i t ä _ e i _ m a k s a
' n y k k ä ä ŋ _ k ö _ ,	n y t k ä ä n _ k u i n _ ,
' s a t à - _ , _ - k a k s k y m m e n t ä ' v i i s	- _ , _ s a t a k a k s i k y m m e n t ä v i i s i
m a r k k a a _ ' l i t r a _ ' m a a l i (ö l i j y h ä n	m a r k k a a _ l i t r a _ m a a l i ö l j y h ä n
o ŋ _ " k o l o m a n n e l l a _ s a j a L L A	o n _ k o l m a n n e l l a _ s a d a l l a
.	.
<BLANK>	<BLANK>
e i _ .	e i _ .
<BLANK>	<BLANK>
k y l l ä _ s e _ ' ò Í	k y l l ä _ s e _ o l i
' k o v o ˆ a _ e n - _ ,	k o v a a _ - _ ,
' ' e n n e ' _ ' ' a i ˇ k a a n ˇ _ s e	e n n e n _ a i k a a n _ s e
' e l ä m ä _ , _ i h m i s i l L A ¨	e l ä m ä _ , _ i h m i s i l l ä
.	.
<BLANK>	<BLANK>
' m i n a _ h a n o _ e i h ä m	m i n ä _ s a n o i _ e i h ä n
' m i n ä _ " s i t ä _ h o k s a n n u	m i n ä _ s i t ä _ h o k s a n n u t
e t t ä _ ' s e _ n y t	e t t ä _ s e _ n y t
' s i n n e k _ k u _ s e	s i n n e _ k u n _ s e
' j ä i _ ' s i t t e _ h ä n	j ä i _ s i t t e n _ h ä n
' l a s k i _ " m a a t a k _ k u	l a s k i _ m a a t a _ k u n
, _ ' j a l a t _ t u l i	, _ j a l a t _ t u l i
" k i p e (ä k s _ .	k i p e ä k s i _ .
<BLANK>	<BLANK>
j a _ m u s t i i _ m y s s y l ö i	j a _ m u s t i a _ m y s s y j ä
j a _ o Í _ .	j a _ o l i _ .
<BLANK>	<BLANK>
' s a - _ , _ ' s a u ´ -	- _ , _ -
, .	, .

Table 1: An example of the preprocessed input training data, when the input format is chunks of 3. Sentences are separated by a special tag *<BLANK>*, and word boundaries are indicated by an underscore character. Characters are separated by white spaces.

Source	Target
' è i _ a n n a k _ k i r v e s t ä	e i _ a n n a _ k i r v e s t ä
a n n a k _ k i r v e s t ä _ p i ʲ e l l ä	a n n a _ k i r v e s t ä _ p i d e l l ä
k i r v e s t ä _ p i ʲ e l l ä _ .	k i r v e s t ä _ p i d e l l ä _ .
<BLANK>	<BLANK>
' e i _ ' s e _ o l p	e i _ s e _ o l e
' s e _ o l p _ ' p i t k ä	s e _ o l e _ p i t k ä
o l p _ ' p i t k ä _ ' m a t k a	o l e _ p i t k ä _ m a t k a
' p i t k ä _ ' m a t k a _ .	p i t k ä _ m a t k a _ .
<BLANK>	<BLANK>
' j u u _ , _ n i i n	j u u _ , _ n i i n
, _ n i i n _ ◡ o l ì §	, _ n i i n _ o l i
n i i n _ ◡ o l ì § _ j a	n i i n _ o l i _ j a
◡ o l ì § _ j a _ ' n y t t	o l i _ j a _ n y t
j a _ ' n y t t _ ◡ o n	j a _ n y t _ o n
' n y t t _ ◡ o n _ n i i n	n y t _ o n _ n i i n
◡ o n _ n i i n _ ,	o n _ n i i n _ ,
n i i n _ , _ " t a v à t t o m a s t i	n i i n _ , _ t a v a t t o m a s t i
, _ " t a v à t t o m a s t i _ , a s ù n ˇ § n o i t a	, _ t a v a t t o m a s t i _ a s u n t o j a
" t a v à t t o m a s t i _ a s ù n ˇ § n o i t a _ ' j o	t a v a t t o m a s t i _ a s u n t o j a _ j o
a s ù n ˇ § n o i t a _ ' j o _ e t	a s u n t o j a _ j o _ e t t ä
' j o _ e t _ .	j o _ e t t ä _ .
<BLANK>	<BLANK>

Table 2: An example of the preprocessed input training data, when the input format is chunks of 3 with a rolling window.

Input data	Training size	
Words	590k	
Chunks of 2	303k	553k
Chunks of 3	208k	520k
Chunks of 4	161k	487k
Chunks of 5	132k	456k
Sentences	35k	

Table 3: Training set sizes. For chunk-level input data, the different training set sizes of the rolling window approach can be seen on the right-most column.

Source	hän	‿oŋ	ko-	,	semmoϑek	kursik	käynny	,	sitt	‿etÄ
Simplified	hän	on	ko-	,	semmosek	kursik	käynny	,	sitt	etÄ
Target	hän	on	-	,	semmoiset	kurssit	käynyt	,	sitten	että

Table 4: An example from the training data showing the differences between the transcribed dialectal source tokens, the simplified source tokens without linguistic transcription markers, and the corresponding normalised target tokens.

LSTM (Long Short-Term Memory) is a recurrent neural network architecture with feedback connections (Hochreiter and Schmidhuber, 1997). This is the vanilla RNN model used in my experiments, and it is trained using the OpenNMT (Klein et al., 2017) default settings. Both the encoder and the decoder have two layers, and the attention mechanism is the general global attention model by Luong et al. (2015). The bidirectional model (BiLSTM) uses mostly the exact same settings as the unidirectional one, the only difference being that it is bidirectional: the input data is processed in two directions, both forwards and and backwards. From now on, the bidirectional recurrent neural network model used in this work will be mostly referred to as BiLSTM; however, when comparing the results in Section 4, I will refer to the similar architecture used by Partanen et al. (2019) as BRNN to stay consistent with the authors' terminology.

The third model architecture is a Transformer. Here, the models are trained using two different setups. In the first approach, I train the models using parameters that mimic the setup presented by Vaswani et al. (2017), which should also be fairly close to what Partanen et al. (2019) used in their experiments: six layers both in the encoder and the decoder, a word embedding size of 512, and a scaled dot product attention mechanism with self-attention. Each of the six layers in the encoder and decoder stacks has two sublayers, which consist of a multi-head self-attention mechanism layer and a position-wise connected feed-forward network layer with the dimensionality of 2048 (Vaswani et al., 2017). The decoder also has a third sublayer for attention, which is performed over the output of the encoder stack. The optimiser is Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.98$, and the dropout rate is $P_{drop} = 0.1$. The learning rate is 2 with 4000 warmup steps and the Noam decay method. Usually, the learning rate is set quite low with Adam: the typical value is 0.001. However, in my preliminary experiments, I noticed that the Transformer models performed exceptionally poorly, and increasing the learning rate seemed to improve the results. Similarly, batch size is increased to 1000, because smaller batch sizes seemed to affect the prediction results negatively.

The second Transformer setup is inspired by Wu et al. (2021). In their experiments, the authors find that batch size has a significant impact on the performance of character-based Transformer models. They set the batch size to 400, when the size of the training set is rather small, only 10k. They also use an overall smaller Transformer: four layers in the encoder and the decoder, four self-attention heads instead of eight, and an embedding size of 256. The hidden size of the

feed-forward layer is 1024. I try to replicate their setup and parameter settings in my own work. However, in the preliminary experiments, I found that a batch size of 400 produced rather poor results, so I increased the batch size to 4000: this is significantly higher than what Wu et al. (2021) used in their experiments, but it seemed to work better for my models. One reason for this may be the amount of available training data: the training set size for my models is larger, which is probably why a larger batch size yields better results. I also use a slightly larger batch size of 1000 in the other Transformer setup, because a smaller batch size value seemed to lead to poor results, as was described earlier in the previous paragraph. The optimiser settings are the same as for the other Transformer setup: Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.98$, and a higher learning rate of 2.

The fourth architecture used in these experiments is a convolutional model (LeCun and Bengio, 1995) following the setup presented by Gehring et al. (2017). Some parameter values are taken from Yolchuyeva et al. (2018). The model has 6 convolutional layers and a kernel width of 3, and the layers use 512 hidden units and 256 embedding dimensions. The optimiser settings are Adam with the values of $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and a learning rate of 0.001. The sentence-level model proved particularly challenging in my experiments, and despite some extensive hyperparameter tuning, I was not able to get any reasonable results. Therefore, in the case of the sentence-level model, I will report and evaluate the results for the model that is trained using only the default parameter values: for example, stochastic gradient descent (SGD) is used as the optimisation method instead of Adam. Other convolutional models—that is, the word-level model and the different chunk-level models—are trained using the parameters described earlier.

3.2.2 Statistical Machine Translation Model

In addition to the neural network models, I use a character-based statistical machine translation approach to normalise Finnish dialects. Several authors have evaluated and compared both SMT and NMT methods in text normalisation tasks, and some have reported better results with the SMT approach: for example, Domingo and Casacuberta (2018) and Çolakoğlu et al. (2019). In my experiments, I use Moses (Koehn et al., 2007), a standard open-source toolkit for statistical machine translation. In the SMT pipeline, Moses is used for phrase extraction and decoding. MGIZA++ (Gao and Vogel, 2008), a multi-threaded implementation of GIZA++ (Och and Ney, 2003), is used for character alignment. For language modelling, I use KenLM (Heafield et al., 2013), and minimum error rate training (MERT) is used as a tuning algorithm for the language model. In the case of SMT, the input data is only processed one sentence at a time, and smaller subsets—words and chunks—are not used as input in my experiments. However, the method is still character-based, so characters are separated by white spaces, and token boundaries are indicated by underscore characters, similar to the input data used by the NMT models.

3.3 Evaluation

Partanen et al. (2019) use the word error rate (WER) to evaluate their results. WER is a metric commonly used to measure the accuracy of text normalisation or automatic speech recognition systems, based on the Levenshtein distance metric (Levenshtein, 1966). The calculation can be formulated as follows:

$$WER = \frac{S+D+I}{S+D+C} \tag{1}$$

where S is the number of substitutions, D is the number of deletions, I is the number of insertions, and C is the number of correct words. Thus, the smaller the value is, the better the score is. I use WER to measure the performance of my models, but I will additionally use two other evaluation metrics for the sake of comparison: BLEU and CHRF. BLEU (Papineni et al., 2002) is a widely popular evaluation metric in machine translation, and CHRF (character n-gram F-score) (Popović, 2015) has been shown to work particularly well with morphologically rich languages. The BLEU metric ranges from 0 to 1, where the score of 1 would mean that the texts that are being compared would essentially be identical. For both BLEU and CHRF, the higher the score is, the better it is, while the opposite is true for WER.

The input data has been preprocessed as described in Section 3.1: sentences are tokenised, and tokens are split into characters separated by white spaces. In the case of longer segments—that is, chunks and whole sentences—tokens are also separated by underscore characters. Therefore, the models generate similarly constructed outputs. Before evaluation, the predictions produced by the models go through a corresponding but opposite process: that is, the white spaces between characters are removed, so that *k o i r a* becomes *koira* again. Similarly, word boundary markers and the special sentence boundary tokens are removed. The desired end result is a collection of standardly formed sentences, one sentence per line.

In the next section, I will report and evaluate the results. Unless otherwise indicated, the Transformer models mentioned in the following section are of the smaller architecture, with fewer layers and a larger batch size. This is because the small model architecture seemed to yield overall better results than the large model. However, I will also briefly compare the results obtained by the two different Transformers. Therefore, if needed, these two model types will be referred to as the 'Small Transformer' and the 'Large Transformer', according to the overall architecture used in the models. Still, a simple 'Transformer' will also refer to the Small Transformer.

4 Results & Discussion

The initial results generally seem to follow the same pattern that was reported by Partanen et al. (2019): the BiLSTM model trained on chunks of 3 performs the best, as can be seen in Table 5. It achieves the lowest WER score, and the highest BLEU and CHRF scores. The results of the BiLSTM and Transformer models can be compared to the results by Partanen et al. shown in Table 6. As a reminder, it should be noted that the Transformer model mentioned here and used in my experiments is the Small Transformer: the architecture is slightly different from the one described in Partanen et al. (2019), but it seemed to yield better results, which is why I chose to use it here for comparison. However, the recurrent neural networks should be fairly similar, as they are trained using mostly only the default settings. Nevertheless, I was not able to get as good WER scores as Partanen et al. did for their best performing models: this may be due to some slight differences in the model settings, such as the values used for hyperparameters or optimisation. It is, however, worth mentioning that both sentence-level models seem to achieve rather good results, when compared to the corresponding WER scores reported by Partanen et al. For example, the sentence-level BiLSTM (BRNN) model has a WER socre of 9.16, while Partanen et al. report a WER score of 46.52 for their sentence-level model. I am not quite sure why this happens, as the training and testing sets should be rather similar in our cases, and encoder-decoder models typically struggle with long sequences.

	BiLSTM			Transformer		
	WER	BLEU	chrF	WER	BLEU	chrF
Words	8.76	83.1	94.5	8.68	83.3	94.6
Chunks of 3	**7.89**	**84.8**	**95.2**	8.04	84.5	95.1
Sentences	9.16	84.6	94.2	10.74	83.3	93.1

Table 5: Results of the initial experiments, following the same setup that was used by Partanen et al. (2019). Both the BiLSTM and Transformer architecture chunk-level models outperform the word-level and sentence-level models in terms of accuracy.

	No normalization	Words		Chunks of 3		Sentences	
		BRNN	Transformer	BRNN	Transformer	BRNN	Transformer
WER	52.89	6.44	6.34	**5.73**	6.1	46.52	53.23

Table 6: Results reported by Partanen et al. (2019, p. 144).

One notable feature of this particular dataset is that the sentences can vary highly in terms of length: some consist of only one token, while others are well over 1,000 characters long, though the latter occurs much more infrequently. The data is transcribed from spoken speech, so some utterances result in sentences that seem overly long in written form, possibly affecting model

performance. In machine translation, a fixed maximum length limit is sometimes set to discard these cases of extra long sentences in order to mitigate the effect: this can be done either in the training phase or in testing. Alternatively, long sentences can be truncated according to a set limit, or they can be simply ignored. In the OpenNMT settings, the default maximum prediction length is 250, which was not changed in my experiments. There is also a minimum prediction length: by default, the value of this is 0, but I increased the value to 2 for the sentence-level Transformer, because the model generated an alarming amount of empty lines on default settings. Arguably, this may skew the results, as all other models use the default minimum length value for prediction. Moreover, changing the value may have caused some erratic behaviour: while going through the prediction outputs produced by the models, I noticed that my sentence-level Transformer had a habit of cutting some sentences abruptly very short. In some cases, these predictions end up being only a few tokens long, sometimes even only a few characters, although the corresponding target sentences are much longer. These instances may be the cases where the model would have generated empty lines on the default prediction settings. However, when the model does manage to predict the whole sentence, the output generally looks fine, and the heavily truncated outputs seem to be a rare enough occurrence that the overall score is not affected too much. The accuracy is still inferior to the word-level and chunk-level models, and the sentence-level BiLSTM model as well, but the differences are rather small.

As predicted, models trained on transcribed data consistently outperform those trained on simplified data, as can be seen in Table 7. This means that the models seem to benefit from a more detailed source input, which makes sense. Consequently, different training data, such as texts solely from social media, where no such linguistic transcriptions can be found, could produce very different results. Possible future work could entail combining these two approaches: using both phonetic transcriptions from spoken speech and user-generated content in the training set.

	LSTM			LSTM-simple-data		
	WER	BLEU	chrF	WER	BLEU	chrF
Words	7.92	84.1	**94.4**	8.34	83.3	94.1
Chunks of 3	**7.83**	**84.2**	94.3	8.16	83.7	94.1
Sentences	9.09	83.9	93.4	9.31	83.5	93.3
	BiLSTM			BiLSTM-simple-data		
	WER	BLEU	chrF	WER	BLEU	chrF
Words	7.71	84.5	94.5	8.02	83.9	94.3
Chunks of 3	**6.82**	**86.2**	**95.2**	7.07	85.8	95.0
Sentences	8.11	86.1	94.2	8.27	85.9	94.1

Table 7: Results of the LSTM and BiLSTM models, comparing the models trained on simplified data to the models that are trained on transcribed data.

The experiments with the rolling/sliding window approach yield rather interesting results, and the different evaluation scores can be seen in Table 8. The unidirectional LSTM method seems to benefit from the sliding context window on smaller chunks, with the 2-chunk model even achieving a better accuracy than the regular 5-chunk model. The scores of the two different 5-chunk models are in fact almost identical for the LSTM architecture. On the other hand, while the bidirectional BiLSTM method also benefits slightly from the rolling window approach on chunks of 2, the regular variants outperform the corresponding sliding window models on longer chunks. Finally, the Transformer models trained on chunks with a rolling window consistently yield better results than the corresponding regular models. However, the difference is more noticeable with smaller chunks, and the scores of the 5-chunk Transformer models are almost identical: this behaviour is similar to the LSTM models. Overall, the regular 5-chunk BiLSTM model without the sliding window approach still achieves the best score, when compared to the Transformer and LSTM models trained on overlapping chunks.

	LSTM			LSTM-overlapping		
	WER	BLEU	chrF	WER	BLEU	chrF
Chunks of 2	7.89	84.2	94.3	**7.69**	**84.5**	**94.5**
Chunks of 3	7.83	84.2	94.3	7.70	**84.5**	**94.5**
Chunks of 4	7.86	84.2	94.3	7.89	84.1	94.3
Chunks of 5	7.85	84.3	94.3	7.85	84.2	94.3
	BiLSTM			BiLSTM-overlapping		
	WER	BLEU	chrF	WER	BLEU	chrF
Chunks of 2	7.06	85.8	95.0	6.97	85.9	95.1
Chunks of 3	6.82	86.2	95.2	6.95	86.0	95.1
Chunks of 4	6.65	86.6	95.3	6.97	86.9	95.1
Chunks of 5	**6.52**	**86.8**	**95.4**	6.90	86.1	95.1
	Transformer			Transformer-overlapping		
	WER	BLEU	chrF	WER	BLEU	chrF
Chunks of 2	8.40	83.8	94.9	8.04	84.4	95.1
Chunks of 3	8.04	84.5	95.1	7.86	84.8	95.2
Chunks of 4	7.90	84.8	95.2	7.72	85.1	95.3
Chunks of 5	7.64	**85.3**	**95.4**	**7.63**	85.2	**95.4**

Table 8: Results of three different model architectures: LSTM, BiLSTM, and the Transformer. Here, the results of the regular chunk-level models are compared to the models that are trained on chunks with a rolling/sliding window.

Regarding the two different Transformer setups, the Small Transformer seems to yield overall better results. The best score is obtained by the chunk-5 models of both architectures, with some slight variation between the other chunk-level models. However, there are notable differences

between the Small Transformer and the Large Transformer when it comes to the word-level and sentence-level models. Interestingly, the Small Transformer outperforms the Large Transformer on both word level and sentence level by a rather large margin, while the scores obtained by the chunk-level models are very similar to each other. The word-level Large Transformer model performs exceptionally poorly when compared to the corresponding Small Transformer model: the WER score of 14.97 obtained by the word-level model is even worse than the sentence-level score, 14.32. The difference in scores is similar in the case of the sentence-level models. However, both sentence-level models also generated a rather large amount of empty lines on the default settings during testing, which is why the minimum prediction length was increased for prediction for both models. Nevertheless, the sentence-level Small Transformer achieves better results.

	Small-Transformer			Large-Transformer		
	WER	BLEU	chrF	WER	BLEU	chrF
Words	8.68	83.3	94.6	14.97	70.9	87.6
Chunks of 2	8.40	83.8	94.9	8.02	83.9	94.2
Chunks of 3	8.04	84.5	95.1	7.99	83.9	94.3
Chunks of 4	7.90	84.8	95.2	8.03	83.9	94.2
Chunks of 5	**7.64**	**85.3**	**95.4**	7.90	84.2	94.3
Sentences	10.74	83.3	93.1	14.32	80.0	88.8

Table 9: Results of the Transformer models.

Comparing all the model architectures that were used in my experiments, the BiLSTM method seems to achieve the overall best results. The different architectures and their evaluation scores can be seen in Table 10. The performance of the convolutional models is especially weak when compared to the other methods: this could possibly be fixed or improved with some intensive fine-tuning, but I was not able to get better scores than the ones reported here, even after trying several different model settings. It should be noted that the sentence-level convolutional model is trained using slightly different parameters than the word-level and chunk-level models, for the sole reason that the default settings were the only way for me to obtain any kind of results on the sentence level. I tried the default settings also on the word-level and chunk-level convolutional models, but the results were slightly worse than the ones reported here. The Transformer model architecture also proved particularly challenging in terms of hyperparameters: however, in the end, I managed to get better results with a larger batch size and a higher learning rate. The results obtained by the statistical machine translation method can be seen in Table 11: when compared to the best-performing NMT model, the BiLSTM model, the SMT approach does not yield desirable results. However, the SMT method is only evaluated on sentence level: this is because it did not seem like smaller segments of data inputs would yield reasonable results, at least with the script I used for training and testing.

Regarding the different input types, chunk-level normalisation methods seem to consistently outperform both word-level and sentence-level systems on all model architectures, although the differences are rather small with the unidirectional LSTM model. However, it is interesting to note that the BiLSTM model seems to benefit from longer sequences, as the model trained on chunks of 5 scores better when compared to the models that are trained on smaller chunks. Still, sentence-long sequences hurt the score. This would suggest that a certain amount of context is necessary for the models to generate good predictions, but too much context can in fact affect the results negatively. The ideal length of the context window may be linked to the amount of training data, as suggested by Hämäläinen et al. (2020a).

	LSTM			BiLSTM		
	WER	BLEU	chrF	WER	BLEU	chrF
Words	7.92	84.1	94.4	7.71	84.5	94.5
Chunks of 2	7.89	84.2	94.3	7.06	85.8	95.0
Chunks of 3	7.83	84.2	94.3	6.82	86.2	95.2
Chunks of 4	7.86	84.2	94.3	6.65	86.6	95.3
Chunks of 5	7.85	84.3	94.3	**6.52**	**86.8**	**95.4**
Sentences	9.09	83.9	93.4	8.11	86.1	94.2
	Transformer			Convolutional		
	WER	BLEU	chrF	WER	BLEU	chrF
Words	8.68	83.3	94.6	19.17	64.2	88.2
Chunks of 2	8.40	83.8	94.9	16.03	70.1	89.1
Chunks of 3	8.04	84.5	95.1	18.07	67.3	87.7
Chunks of 4	7.90	84.8	95.2	19.25	65.7	86.9
Chunks of 5	7.64	85.3	**95.4**	20.59	64.1	86.0
Sentences	10.74	83.3	93.1	46.29	39.7	68.4

Table 10: Results of the different NMT model architectures. The overall best scores measured in three different evaluation metrics have been bolded.

	BRNN			SMT		
	WER	BLEU	chrF	WER	BLEU	chrF
Words	7.71	84.5	94.5			
Chunks of 3	**6.82**	**86.2**	**95.2**			
Sentences	8.11	86.1	94.2	10.53	80.0	93.5

Table 11: SMT results.

In the following section, I will go through the prediction outputs generated by the models in greater detail and describe some normalisation errors found in the predictions. Unless otherwise indicated, the models mentioned in this manual error analysis are of the BiLSTM architecture.

4.1 Manual Error Analysis

Some common error types can be found when taking a closer look at the predicted outputs produced by the models, and these errors generally seem to fall in line with the findings reported by Partanen et al. (2019). Many dialectal variants of single words match their Standard Finnish counterparts, so in many cases the regional word-forms end up requiring no changes at all. Most of the changes required between the source and target are individual insertions, replacements or deletions, especially at the end of the word.

Partanen et al. (2019) specifically mention the following error types:

Sporadic vowel lengthening Finnish distinguishes between short and long vowels, and this distinction is realised in writing as doubled characters. Thus, *tuli* 'fire' and *tuuli* 'wind' are both written and pronounced differently. In the training data used here, however, there are instances of vowel lengthening that does not mark phonological contrast. Since the data is transcribed from spoken speech, such sporadic long vowels may simply be a sign of the speaker emphasising this particular part of the utterance in an unconventional way. This kind of behaviour is highly irregular and, consequently, difficult to predict.

Compound words Finnish utilises compounding, and this is a highly productive word formation process in the language. Long character sequences can possibly be challenging for a normalisation model to learn, as there is more room for irregular transformation patterns.

Long vowel sequences Standard Finnish vowel sequences correspond to diphthongs in many dialects, or they may be broken up by an internal *-h-*. This forms multiple transformation patterns, possibly proving difficult for the models to learn.

Ambiguous word-forms As was mentioned in Section 3.1, the fairly common habit of dropping or shortening morphological suffixes causes ambiguity between word-forms. For example, the dialectal word-form *meil* can translate either to the adessive form *meillä* or the allative variant *meille*.

Sandhi Various sandhi phenomena are very common and frequent in Finnish. Sandhi means sound changes between morphemes and individual words: in the Finnish language, these phonological processes can be split into assimilation and gemination. An example of assimilation would be the labialisation of the consonant sequence /np/ in *olenpa*, resulting in *olempa*. As an example of gemination, the word-initial /p/ in *tule pian* is usually pronounced as a lengthened consonant at the word boundary, resulting in *tulep pian*. These sound changes are not represented in the standard orthography, but especially intra-word sandhi may be written down in informal contexts—for example, the aforementioned

olempa instead of *olenpa*, or *tännekkin* instead of *tännekin*. The training data consists of transcribed spoken speech, so sandhi sound changes are also marked down on the source side, and there are numerous instances of such phonological phenomena.

The sporadic vowel lengthening that Partanen et al. (2019) mention observing can also be found in the test set I used for my models. There is an instance of an inflected dialectal pronoun variant *hee*, which has the target translation of *heidän* 'their'. The 5-chunk BiLSTM model leaves this word-form as is, while the word-level and 3-chunk models transform it into *he*, which would be the uninflected word-form with the meaning of 'they' in Standard Finnish. This could be related to the dialectal pronoun variant *heet* Partanen et al. (2019) mention in their findings. There are also a few instances of non-standard extra long vowels: *eeei* instead of *ei*, and *eeehä* instead of *eihän*, both cases uttered by the same speaker. Both words have the basic meaning of 'no', but the latter is additionally followed by the emphatic clitic *-han*, surfacing as *-hän* because of vowel harmony. Both of these examples seem like cases of sporadic, emphatic vowel lengthening, but the models have no problems normalising them. There is even an instance of a particularly emphatic 'no' in the test data: *'eee(ei!*. This is, again, correctly normalised as *ei!* by all models.

A related phenomenon is the tendency for long vowel sequences to surface either as diphthongs or a word-internal /h/ in dialectal speech. For example, there is an instance of the Standard Finnish word *varmaan* 'probably' being represented as *varmahal* in the test data, and another instance of the word *maailma* 'world' surfacing as *moalima*. Both words receive correct predictions by the BiLSTM models. However, the dialectal word-form *"soahaaŋkoon*, which should be transformed into *saadaankohan*, is incorrectly translated as *saavaankoon* by the 5-chunk model. Here, the first diphthong is correctly interpreted as a long /a/, but the word-final clitic *-han* surfacing as a long vowel proves difficult for the model. Other models struggle with this particular word as well: alternative predictions produced by the different models can be seen in Table 12. While the output generated by the 5-chunk model is not a valid Finnish word-form, the transformation patterns it seems to follow do make certain sense: the base verb *saada* 'get; receive' can go through the sound change from [d] to [v] when forming the active present participle of the verb, *saava*. For example, *Lapsi haluaa saada lahjoja* 'The child wants to get gifts' vs. *Lahjoja saava lapsi* 'The child who gets gifts'. This word-form can also go through case declension: *Hän tutustui lahjoja saavaan lapseen* 'S/he got to know the child who gets gifts'. Additionally, the word-final *-koon* can be used to form the passive imperative: *Saakoon lahjoja* 'Let her/him get gifts'. However, these two sequences do not fit together, and the resulting output *saavaankoon*, suggested by the model, is decidedly ungrammatical. Another example of an invalid prediction is the one generated by the 3-chunk model, *saahankohan*: here, the long vowel sequence in the middle of the source token is transformed into a short /a/, and the resulting *-han-* can possibly be interpreted as an extra clitic, rendering the prediction ungram-

matical. On the other hand, the word-level model shortens the token radically into *saankaan*. This is a perfectly valid inflected form of the base verb *saada* ‘get; receive’, but the meaning is very different from the intended target and, in the context of this particular sentence, rather nonsensical. Both word-forms are a bit difficult to translate on their own, but the target token contains two clitics—first, the clitic *-ko* to form a question, and then *-han* to soften the question, expressing politeness—while the predicted output contains a different clitic, *-kaan*, expressing wonder and surprise.

Source	minä,	'kyššyy	multa	että,	"soàhaaŋ̀koon	'sitä.
Target	minä,	kysyy	minulta	että,	saadaankohan	sitä.
BiLSTM-W	minä,	kysyy	minulta	että,	**saankaan**	sitä.
BiLSTM-3	minä,	kysyy	minulta	että,	**saahankohan**	sitä.
BiLSTM-5	minä,	kysyy	minulta	että,	**saavaankoon**	sitä.
BiLSTM-Sent	minä,	kysyy	minulta	että,	**saadaanko**	sitä.

Table 12: An example sentence from the test data, showing the detokenised source and target tokens, and the predictions generated by different models. The source token *"soàhaaŋ̀koon* contains a diphthong and two long vowels: all models interpret the dialectal diphthong correctly, transforming it into the corresponding long vowel found in Standard Finnish, but the other two vowel sequences prove challenging for the models.

The data contains multiple instances of very long words and compound words. Many of these are actually numbers spelled out: for example, the year 1878 is written as *tuhatkahdeksansataaseitsemänkymmentäkahdeksan*. Consequently, the models have never encountered an actual number sequence in the training data. Similarly, dates are always written out as words rather than numbers: *kahdeskymmenesneljäs päivä heinäkuuta* instead of 24.7. The Finnish numerals are, however, generally highly systematic, even when inflected, so the models do not seem to have problems with them. There are a few instances of incorrect predictions: for example, the 5-chunk model translates *sataanelijääkymmentä* incorrectly as *sataneljäkymmentä*, in its basic non-inflected form, even though the target is in the partitive case, *sataaneljääkymmentä*. Some examples of long numerals, both correctly and incorrectly translated, can be seen in Table 13 and Table 14.

Long nouns, especially rare ones, receive more consistently incorrect predictions. For instance, the test set has the Standard Finnish word *kreikkalaiskatolilaiseksihan*—a dated term for a member of the Orthodox Church, inflected and also combined with a clitic—represented as *reikalaiskatońilaiseksha*. The character *l* surfacing as *ń* in the middle of the word seems like a rather irregular transformation pattern, possibly a speech error. The 5-chunk model translates this sequence of characters into *reikälaiskatonilaiseksihan*, interpreting the start of the word, *reika*, incorrectly as *reikä* ‘hole’, and violating the rules of Finnish vowel harmony in the process. The word-level model strips the final clitic *-han* off the word-form but manages to inflect

Source	no	'kyl	nes	'siin	‿olìk
Target	no	kyllä	ne	siinä	olivat
BiLSTM-W	no	kyllä	ne	siinä	**oli**
BiLSTM-3	no	kyllä	ne	siinä	olivat
BiLSTM-5	no	kyllä	ne	siinä	**oli**
BiLSTM-Sent	no	kyllä	ne	siinä	**oli**
Source	'kahðeŋgymne,	'jokùssi	olì	likì	"kolmeŋgymneŋgi
Target	kahdenkymmenen,	jokusia	oli	liki	kolmenkymmenenkin
BiLSTM-W	kahdenkymmenen,	jokusia	oli	liki	kolmenkymmenenkin
BiLSTM-3	kahdenkymmenen,	jokusia	oli	liki	kolmenkymmenenkin
BiLSTM-5	kahdenkymmenen,	jokusia	oli	liki	kolmenkymmenenkin
BiLSTM-Sent	kahdenkymmenen,	jokusia	oli	liki	kolmenkymmenenkin
Source	mut,	ol	semmoϑϑì	'hyvì	ja,
Target	mutta,	oli	semmoisia	hyviä	ja,
BiLSTM-W	mutta,	oli	semmoisia	hyvin	ja,
BiLSTM-3	mutta,	oli	semmoisia	hyvin	ja,
BiLSTM-5	mutta,	oli	semmoisia	hyviä	ja,
BiLSTM-Sent	mutta,	oli	semmoisia	hyviä	ja,

Table 13: An example of a part of a sentence from the test data, showing the detokenised source and target tokens, and the predictions generated by different models. In the middle section of the table, there are two examples of a numeral being translated correctly by all compared models: *'kahðeŋgymne* and *"kolmeŋgymneŋgi*. In the top part of the table, however, there is an example of a conjugated verb affected by word-boundary sandhi: the target token is *olivat* 'be-PST-3PL', but three models translate the source token into the corresponding singular word-form *oli* 'be-PST-3SG'.

the word according to the rules of Standard Finnish: *reikalaiskatonilaiseksi*. Uncommon words like these are often challenging for the models to learn, especially if they contain multiple different character transformation patterns.

There are a few examples of rare word-forms that only occur in the test set and never in the training data: for example, an old dialectal word *myöläkkä*, the meaning of which I was regrettably unable to find. This word is the same on the source and target sides, but the models are unable to normalise it correctly. There are valiant efforts, though, and the models do give predictions that seem to make sense and follow the phonotactic rules of the Finnish language, even if the outputs are not actual recognisable words. Examples of these generated outputs can be seen in Table 15. Sometimes the words themselves are not necessarily uncommon, but the form they take on the source side may be unconventional: such is the case with *-kevväimestä*, normalised as *keväästä* 'spring-PTV'. Here, the models seem to focus on different parts of the character sequence, possibly confused by the initial hyphen used as a linguistic transcription marker. The 5-chunk model offers *keväistä* 'springlike-PTV' as a prediction, which is visually quite close to the intended normalisation, the only difference being a matter of one character—

Source	men	kaks	sit	soδàs	jà,
Target	meni	kaksi	sitten	sodassa	ja,
BiLSTM-W	meni	kaksi	sitten	sodassa	ja,
BiLSTM-3	meni	kaksi	sitten	sodassa	ja,
BiLSTM-5	meni	kaksi	sitten	sodassa	ja,
BiLSTM-Sent	meni	kaksi	sitten	sodassa	ja,
Source	kolmàs	kual	ny	viimmeis	talvel...
Target	kolmas	kuoli	nyt	viimeis	talvella...
BiLSTM-W	kolmas	kuoli	nyt	**viimeissä**	talvella...
BiLSTM-3	kolmas	kuoli	nyt	**viimeistä**	talvella...
BiLSTM-5	kolmas	kuoli	nyt	**viimeissä**	talvella...
BiLSTM-Sent	kolmas	kuoli	nyt	viimeis	talvella...
Source	semne	ko	ol	kuutkymmentkolmì	käy(vä).
Target	semmoinen	kun	oli	kuuttakymmentäkolmea	käyvä.
BiLSTM-W	semmoinen	kun	oli	**kuutkymmentäkolme**	**käydä.**
BiLSTM-3	semmoinen	kun	oli	**kuutkymmentäkolmia**	**käydä.**
BiLSTM-5	semmoinen	kun	oli	**kuutkymmentäkolmea**	**käydä.**
BiLSTM-Sent	semmoinen	kun	oli	**kuutkymmentäkolmea**	**käydä.**

Table 14: An example of a part of a sentence from the test data, showing the detokenised source and target tokens, and the predictions generated by different models. In the bottom section of the table, there is an example of a numeral receiving incorrect predictions by all models: *kuutkymmentkolmì*. In the same section, the final active present participle *käy(vä)* is incorrectly interpreted as an uninflected verb.

still, the output itself is a valid Finnish word-form, albeit a different word class than the intended target token. The word-level model, on the other hand, normalises the word-form as *seitsemänväimestä*: a combination of *seitsemän* 'seven' and a nonsensical but phonologically possible ending *-väimestä*. The 3-chunk level model suggests *viimeisestä* 'last-PTV': this is, again, a perfectly valid Finnish word-form, but the meaning is completely different. Other predictions can be seen Table 15.

Ambiguous word-forms also prove challenging for the models. The aforementioned example of *meil* can be found in the training data: in Table 17, there is an example of the chunk-5 model predicting the noun case of the word correctly, while the other models suggest a wrongly inflected word-form. The sandhi phenomena further complicate ambiguities between word-forms. For example, *kymmentäkään vuotta* is realised as *kymmentäkkääv vuotta* in the dialectal transcriptions, exhibiting gemination both on a morpheme boundary and the word boundary. This particular case receives the correct prediction, and the model knows to substitute the word-final character *v* with the genitive suffix *-n*. As an example of an incorrect prediction, *kakskymmentäkakskil lypsävätä* is translated as *kaksikymmentäkaksikin lypsävää*: here, the model interprets the word-final characters as the clitic *-kin*, even though the word should be in its basic form, *kaksikymmentäkaksi*. The sandhi example that Partanen et al. (2019) give can also be found in

Source	alakuja	ai-,	alakutala-,	tuota,	-kevväimestä
Target	alkujaan	-,	alku,	tuota,	keväästä
BiLSTM-W	alkuja	-,	**-,**	tuota,	**seitsemänväimestä**
BiLSTM-3	alkuja	-,	**-,**	tuota,	**viimeisestä**
BiLSTM-5	alkuja	-,	**-,**	tuota,	**keväistä**
BiLSTM-Sent	alkuja	-,	**-,**	tuota	**keväimestä**
Source	sillä	eij	◡oom	'myöläkkä	niin
Target	sillä	ei	ole	myöläkkä	niin
BiLSTM-W	sillä	ei	ole	**myödenkään**	niin
BiLSTM-3	sillä	ei	ole	**myöden**	niin
BiLSTM-5	sillä	ei	ole	**myödenkään**	niin
BiLSTM-Sent	sillä	ei	ole	**myödenkään**	niin
Source	se	'reijällaita	niin	s	◡eij
Target	se	reiänlaita	niin	se	ei
BiLSTM-W	se	reiänlaita	niin	se	ei
BiLSTM-3	se	reiänlaita	niin	se	ei
BiLSTM-5	se	reiänlaita	niin	se	ei
BiLSTM-Sent	se	reiänlaita	niin	se	ei

Table 15: An example of a part of a sentence from the test data, showing the detokenised source and target tokens, and the predictions generated by different models. There are two cases of rare word-forms, *-kevväimestä* and *'myöläkkä*, that get incorrectly translated by all models.

my test data: *'vuoristol 'laitaa*. In this case, the target token for the first word-form is *vuoriston* 'mountain-GEN', but the genitive suffix *-n* disappears from the surface form because of the sound change caused by the following word. In my experiments, the word-level model incorrectly interprets this word-form as *vuoristolla* 'mountain-ADESS', but all the other models predict the output correctly. A similar example of incorrect predictions can be seen in Table 18: here, the target token *tallin* 'stable-GEN' is realised as *'tallil* because of the sandhi influence caused by the following token, *"laatti◡alle*. The word-level model offers the word-form *tallilla* 'stable-ADESS', while the sentence-level and chunk-5 models suggest a different prediction, *tallille* 'stable-ALL'. However, the 3-chunk level predicts the output correctly.

An interesting example is the dialectal word *ku* (realised as *ko* in some dialects): in Standard Finnish, *kuin* 'as' and *kun* 'when' are spelled and pronounced differently, but in spoken colloquial Finnish the distinction is not usually made, resulting in an identical word-form. The word has several slightly differing meanings and grammatical functions, and as such it is considerably common in speech and can have various surface forms. This case is also specifically mentioned in the normalisation guidelines that were used for the dataset: in ambiguous cases, a choice has been made in favour of *kun*, because it covers a wider array of functions and use cases (Vilkuna, 2014). In my experiments, the models seem to have difficulties detecting whether the shortened dialectal *ku* should be normalised as *kuin* or *kun*, resulting in some ungrammatical sentences.

Source	näkyy	"lehess	◡olevan	että	"siell
Target	näkyy	lehdessä	olevan	että	siellä
BiLSTM-W	näkyy	lehdessä	olevan	että	siellä
BiLSTM-3	näkyy	lehdessä	olevan	että	siellä
BiLSTM-5	näkyy	lehdessä	olevan	että	siellä
BiLSTM-Sent	näkyy	lehdessä	olevan	että	siellä
Source	◡olih	vielä	'ollu,	"rulli	'matkassa
Target	oli	vielä	ollut,	trulli	matkassa
BiLSTM-W	**olisi**	vielä	ollut,	trulli	matkassa
BiLSTM-3	**olisi**	vielä	ollut,	**rulli**	matkassa
BiLSTM-5	**olisi**	vielä	ollut,	**rulli**	matkassa
BiLSTM-Sent	**olisi**	vielä	ollut,	**rulli**	matkassa
Source	ku	lamphaaŋ	karittaŋ	kyl^{i}jek	kerìnyh
Target	kun	lampaan	karitsan	kyljet	kerinnyt
BiLSTM-W	kun	lampaan	karitsan	**kylje**	kerinnyt
BiLSTM-3	kun	lampaan	karitsan	**kylje**	kerinnyt
BiLSTM-5	kun	lampaan	karitsan	**kylje**	kerinnyt
BiLSTM-Sent	kun	lampaan	karitsan	**kylje**	kerinnyt
Source	semmosem	"pläsin	että	ne	on
Target	semmoisen	pläsin	että	ne	on
BiLSTM-W	semmoisen	**läsin**	että	ne	on
BiLSTM-3	semmoisen	**läsin**	että	ne	on
BiLSTM-5	semmoisen	**läsin**	että	ne	on
BiLSTM-Sent	semmoisen	pläsin	että	ne	on
Source	o-,	vähhääkhä	jääny	vilLAA,	"villaa
Target	-,	vähääkään	jäänyt	villaa,	villaa
BiLSTM-W	-,	**vähänhän**	jäänyt	villaa,	villaa
BiLSTM-3	-,	vähääkään	jäänyt	villaa,	villaa
BiLSTM-5	-,	vähääkään	jäänyt	villaa,	villaa
BiLSTM-Sent	-,	vähääkään	jäänyt	villaa,	villaa
Source	siihem,	"pilkav	vaim	'mitȁ	hään
Target	siihen,	pilkan	vain	mitä	hän
BiLSTM-W	siihen,	pilkan	vain	mitä	hän
BiLSTM-3	siihen,	pilkan	vain	mitä	hän
BiLSTM-5	siihen,	pilkan	**vai**	mitä	hän
BiLSTM-Sent	siihen,	pilkan	**vai**	mitä	hän

Table 16: An example of a part of a sentence from the test data, showing the detokenised source and target tokens, and the predictions generated by different models. There are cases of sandhi at several word boundaries: for instance, *lamphaaŋ karittaŋ kyl^{i}jek kerìnyh*. The word-final consonants of the first two tokens are velarised, but the models have no problems recognising the transformation pattern from *ŋ* to *n*. However, each model fails to normalise the third word-form of the sequence correctly: *kyl^{i}jek* is consistently translated as *kylje*, which is not a valid Finnish word.

Source	no,	a	kyl	"meil	'riitti
Target	no,	a	kyllä	meille	riitti
BiLSTM-W	no,	a	kyllä	**meillä**	riitti
BiLSTM-3	no,	a	kyllä	**meillä**	riitti
BiLSTM-5	no,	a	kyllä	meille	riitti
BiLSTM-Sent	no,	a	kyllä	**meillä**	riitti
Source	ko	myö,	'myö	vietii	'teält
Target	kun	me,	me	vietiin	täältä
BiLSTM-W	kun	me,	me	vietiin	täältä
BiLSTM-3	kun	me,	me	vietiin	täältä
BiLSTM-5	kun	me,	me	vietiin	täältä
BiLSTM-Sent	kun	me,	me	vietiin	täältä
Source	"kolkyönt	säkkii	"perunŏà,	ja	"possut.
Target	kolmekymmentä	säkkiä	perunaa,	ja	possut.
BiLSTM-W	kolmekymmentä	säkkiä	perunaa,	ja	possut.
BiLSTM-3	kolmekymmentä	säkkiä	perunaa,	ja	possut.
BiLSTM-5	kolmekymmentä	säkkiä	perunaa,	ja	possut.
BiLSTM-Sent	kolmekymmentä	säkkiä	perunaa,	ja	possut.

Table 17: An example of a part of a sentence from the test data, showing the detokenised source and target tokens, and the predictions generated by different models. In the top section of the table, there is an example of the dialectal word-form *meil* receiving incorrect predictions by all but the chunk-5 model.

Source	tonnet	tonne,	sinnet	'tallil	"laatti(alle	ja
Target	tuonne	tuonne,	sinne	tallin	lattialle	ja
BiLSTM-W	tuonne	tuonne,	sinne	**tallilla**	lattialle	ja
BiLSTM-3	tuonne	tuonne,	sinne	tallin	lattialle	ja
BiLSTM-5	tuonne	tuonne,	sinne	**tallille**	lattialle	ja
BiLSTM-Sent	tuonne	tuonne,	sinne	**tallille**	lattialle	ja
Source	siit	ne	'se(attii	'hyvin	"sekonee	ja
Target	sitten	ne	seattiin	hyvin	sekoineen	ja
BiLSTM-W	sitten	ne	seattiin	hyvin	sekoineen	ja.
BiLSTM-3	ja sitten	ne	seattiin	hyvin	sekoineen	ja.
BiLSTM-5	sitten	ne	seattiin	hyvin	sekoineen	ja.
BiLSTM-Sent	sitten	ne	sekattiin	hyvin	sekoineen	ja.

Table 18: An example of a part of a sentence from the test data, showing the detokenised source and target tokens, and the predictions generated by different models. There is an example of the source token *'tallil* receiving incorrect predictions due to the sandhi phenomenon happening at the word boundary.

Some examples of this can be seen in Table 19. Here, the source and target tokens are compared to the prediction outputs produced by the 3-chunk model. At lines 2 and 5, the dialectal *ku* receives different predictions, but these predictions do not match the target normalisation. At line 15, there is a different dialectal variant of the same word, *köm*, most likely influenced by

the surrounding sounds in this particular utterance: here, the model does recognise the word as a variation of *ku/ko*, but again the prediction ends up being an incorrect one.

The first two sentences of the previous set of examples can also be seen in Table 20: this time, illustrating the predictions generated by the BiLSTM model trained on longer segments of data (*N*=5). The model makes the same mistakes on the first few rows as the other chunk-level model, but *köm* at line 9 is correctly normalised as *kuin*. This may be a case of lucky chunking process: when the sentences have been tokenised and separated into chunks, *köm* has been paired with the word *ennen* in the same context window, whereas on the chunks of 3 approach these two words have been severed from each other. The phrase *ennen kuin* 'before' is most likely a fairly common occurrence on the target side of the training data—on the other hand, *ennen kun* would be highly atypical and potentially ungrammatical—so context clues may guide the model to correctly choose *kuin* instead of *kun*. The opposite may be true for line 5 in Table 19 and line 3 in Table 20: both chunk windows happen to include the word *niin* preceding the key word *ku*. This is again a rather frequent combination of words, which is typically realised as *niin kuin* 'as if' in Standard Finnish, and the models have probably learnt this pattern from the training data. In this particular case, however, *niin* is tied to the verb before it rather than to the following conjunction, describing the intensity of the action, and *kun* has the standard meaning of 'when', separate from the preceding word. This would be a case of an unlucky chunking process: both models are making a very sensible educated guess as to what the normalised form of *ku* should be, based on the training data, but in this case the prediction turns out to be incorrect.

Regarding this so-called unlucky chunking process, input data that utilises the sliding/rolling window approach could possibly solve problems like the one described earlier. In theory, models that have been trained with this type of data have more information on the context around the target token, and they have more examples of ambiguous word-forms in different positions inside a sentence. However, looking at the predictions, the 5-chunk model with an overlapping window also struggles with the aforementioned case of *kun* vs. *kuin*: in fact, the predictions generated by the model are exactly identical to the outputs shown in Table 20.

In Table 19, there is also an example of the model translating the token *työ* 'work' as *te* 'you-PL'. This is an interesting case of overlearning: in Eastern Finnish dialects, the short vowels of the plural personal pronouns are replaced with the diphthong *yö*. Thus, *me* becomes *myö*, *te* becomes *työ*, and so on. The word *työ* also means 'work' and, in Standard Finnish and many other dialects, this is the sole meaning of the word, without the additional meaning of the plural personal pronoun 'you'. However, the model has encountered the pattern enough times that it applies the transformation rule to the characters even when this is not needed.

	Source	Prediction	Target
1	eikä se tài̯la	eikä se taida	eikä se taida
2	niin niukkoakkaa ku	niin **niukkaakaan kun**	niin niukkoakaan kuin
3	toi... se	**toi**... se	tuo... se
4	niukkuu, ja	**niukkua**, ja	niukkoo, ja
5	irvistelie niiŋ ku	irvistelee niin **kuin**	irvistelee niin kun
6	sillej juomaakii viep	sille juomaakin vie	sille juomaakin vie
7	nin, se	niin, se	niin, se
8	irvistie nii että	**irvistiä** niin että	irvistää niin että
9	luulie että se	**luulia** että se	luulee että se
10	purie mut ei	puree mutta ei	puree mutta ei
11	os sentä purru	ole sentään purrut	ole sentään purrut
12	.	.	.
13	sittes seki ostettihij	sitten sekin ostettiin	sitten sekin ostettiin
14	jo talooksi enneŋ	jo taloksi ennen	jo taloksi ennen
15	köm m‿oon	**kun** minä olen	kuin minä olen
16	tähän tullu.	tähän tullut.	tähän tullut.
17	mutta, mutta	mutta, mutta	mutta, mutta
18	, kunn‿ei	, kun ei	, kun ei
19	sitä rahaa tahtonus	sitä rahaa tahtonut	sitä rahaa tahtonut
20	saaran niin sitten	saada niin sitten	saada niin sitten
21	taharottihij jotta päästääs	tahdottiin jotta **päästää**	tahdottiin jotta päästäisiin
22	piiʲaksi.	piiaksi.	piiaksi.
23	kài̯ työ oĺpa	kai **te** olipa	kai työ olipa
24	hiäv vài̯kka miŋkälàista	hän vaikka minkälaista	hän vaikka minkälaista
25	ni, se	niin, se	niin, se
26	piti käsin tehä.	piti käsin tehdä.	piti käsin tehdä.
27	.	.	.

Table 19: An example of the detokenised chunk-level (N=3) source and target tokens, and the predictions generated by the model. Errors in the prediction output have been bolded.

	Source	Prediction	Target
1	eikä se tài̯la niin niukkoakkaa	eikä se taida niin **niukkaakaan**	eikä se taida niin niukkoakaan
2	ku toi... se niukkuu	**kun toi**... se **niukkuu**	kuin tuo... se niukkoo
3	, ja irvistelie niiŋ ku	, ja irvistelee niin **kuin**	, ja irvistelee niin kun
4	sillej juomaakii viep nin,	sille juomaakin vie niin,	sille juomaakin vie niin,
5	se irvistie nii että luulie	se **irvisti** niin että luulee	se irvistää niin että luulee
6	että se purie mut ei	että se puree mutta ei	että se puree mutta ei
7	os sentä purru.	ole sentään purrut.	ole sentään purrut.
8	sittes seki ostettihij jo talooksi	sitten sekin ostettiin jo taloksi	sitten sekin ostettiin jo taloksi
9	enneŋ köm m‿oon tähän	ennen kuin minä olen tähän	ennen kuin minä olen tähän
10	tullu.	tullut.	tullut.

Table 20: An example of the detokenised chunk-level (N=5) source and target tokens, and the predictions generated by the model. Errors in the prediction output have been bolded.

5 Conclusion

In this thesis, I have attempted to replicate the results that have been reported in previous work on Finnish dialect text normalisation by Partanen et al. (2019). The authors use neural machine translation methods to normalise Finnish dialects, obtaining very good results in the process. The chosen model architectures are the bidirectional recurrent neural network (BRNN) and the Transformer. The results show certain similarities when compared, all experiments indicating that the BRNN model outperforms other character-based machine translation architectures in dialectal text normalisation. However, the evaluation scores obtained from my experiments are considerably lower than those reported by Partanen et al. It is difficult to say why this is the case, as there are various possible reasons as to why my models performed poorly in comparison. The dataset is the same, so the training data that is fed to the models should be fairly similar. There may, however, be some differences regarding the hyperparameter values or other model settings: these may affect the results, despite the models sharing the same general architecture. The evaluation processes may also be slightly different: for example, I have chosen to include punctuation marks in the prediction outputs that are used for evaluation, and punctuation is also detokenised according to standard writing conventions. The sentences are also not lowercased, so the training set retains instances of capitalised words, such as names.

I have also conducted further experiments with different neural network models and a statistical machine translation approach. The neural network architectures that are used in the experiments are the recurrent neural network (RNN) model and the convolutional model, in addition to the BRNN and Transformer models that were also used by Partanen et al. (2019). The vanilla RNN model is outperformed by the more advanced BRNN model, although the differences in the evaluation scores are rather small. In the end, I was not able to get good results with the convolutional models, despite trying out several different parameter settings. I also had no luck with the SMT approach: when the final results of the NMT and SMT methods are compared, the NMT models generally seem to outperform the SMT approach.

I have also used two different Transformer setups, and the results obtained by these models were then compared and evaluated. In my experiments, the Transformer with smaller dimensions and fewer layers outperforms the larger Transformer model. The smaller Transformer also uses a larger batch size, which may have a positive effect on the model performance, as noted by Wu et al. (2021). Even though the final differences in the evaluation scores are very small, especially with the various chunk-level models, this is a positive finding: smaller Transformers result in models that train faster and require less disk space. With some fine-tuning, Transformers may even be competitive with the RNN-based methods. In my experiments, however, the BRNN method outperforms both Transformer models.

In addition to different model architectures, I have also experimented with different input strategies. In this work, I use the same input strategies as Partanen et al. (2019): word-level, chunk-level, and sentence-level training data. Additionally, I use chunks of 2, 4, and 5. I also experimented with a rolling/sliding window approach. One of the key findings is that chunk-level training data seems to work consistently better than word-level and sentence-level methods in text normalisation tasks. In this work, I have also demonstrated that the neural network models might even benefit from longer input chunks, with the BRNN model trained on chunks of 5 outperforming the similar model that was trained only on chunks of 3. However, the size of the training set may be crucial regarding the optimal input type and the length of the character segments, and smaller chunks may be the preferred option when the amount of available training data is scarce.

Active research has been done on the topic of Finnish dialect normalisation, and there are also existing tools for the task, such as the Murre normaliser tool by Partanen et al. (2019). Future work could entail focusing on more contemporary data, such as social media texts, or spoken language. For example, the dataset used in this thesis, the Samples of Spoken Finnish Corpus (Institute for the Languages of Finland, 2014), contains dialectal speech that is already quite different from the spoken Finnish variants of today. Dialect normalisation continues to be a topic of interest; not only in the context of Finnish, but other languages as well. For example, there has been recent work done on Estonian dialect normalisation, where the Finnish dialect generation models developed by Hämäläinen et al. (2020b) are used to convert Standard Estonian into a pseudo-dialect to combat data sparsity (Hämäläinen et al., 2022). Dialects are indeed often under-resourced in natural language processing, and generating synthetic data can potentially help in developing tools and applications for dialectal data. Low-resource languages such as Veps or Votic, both relatives of Finnish and Estonian, could benefit from a similar approach.

References

Adesam, Yvonne, Malin Ahlberg, and Gerlof Bouma (2012). "bokstaffua, bokstaffwa, bokstafwa, bokstaua, bokstawa. . . Towards lexical link-up for a corpus of Old Swedish". In: *Proceedings of the 11th Con-ference on Natural Language Processing (KON-VENS 2012), LThist 2012 workshop*. Vienna, Austria, pp. 365–369.

Baron, Alistair and Paul Rayson (2008). "VARD 2: A tool for dealing with spelling variation in historical corpora". In: *Proceedings of the Postgraduate Conference in Corpus Linguistics*.

Bollmann, Marcel (2012). "(Semi-)Automatic Normalization of Historical Texts using Distance Measures and the Norma tool". In: *Proceedings of the Second Workshop on Annotation of Corpora for Research in the Humanities (ACRH-2)*. Lisbon, Portugal, pp. 3–12.

Bollmann, Marcel (2018). "Normalization of Historical Texts with Neural Network Models". Bochumer Linguistische Arbeitsberichte, 22. PhD thesis.

Bollmann, Marcel (2019). "A Large-Scale Comparison of Historical Text Normalization Systems". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 3885–3898. DOI: 10.18653/v1/N19-1389. URL: https://aclanthology.org/N19-1389.

Bollmann, Marcel, Joachim Bingel, and Anders Søgaard (2017). "Learning attention for historical text normalization by learning to pronounce". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, pp. 332–344. DOI: 10.18653/v1/P17-1031. URL: https://aclanthology.org/P17-1031.

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*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 131–139. URL: https://aclanthology.org/C16-1013.

Bollmann, Marcel, Anders Søgaard, and Joachim Bingel (2018). "Multi-task learning for historical text normalization: Size matters". In: *Proceedings of the Workshop on Deep Learning*

Approaches for Low-Resource NLP. Melbourne: Association for Computational Linguistics, pp. 19–24. DOI: 10.18653/v1/W18-3403. URL: https://aclanthology.org/W18-3403.

Chahuneau, Victor, Eva Schlinger, Noah A. Smith, and Chris Dyer (2013). "Translating into Morphologically Rich Languages with Synthetic Phrases". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, pp. 1677–1687. URL: https://aclanthology.org/D13-1174.

Çolakoğlu, Talha, Umut Sulubacak, and Ahmet Cüneyd Tantuğ (2019). "Normalizing Non-canonical Turkish Texts Using Machine Translation Approaches". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*. Florence, Italy: Association for Computational Linguistics, pp. 267–272. DOI: 10.18653/v1/P19-2037. URL: https://www.aclweb.org/anthology/P19-2037.

De Clercq, Orphée, Sarah Schulz, Bart Desmet, Els Lefever, and Véronique Hoste (2013). "Normalization of Dutch User-Generated Content". In: *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*. Hissar, Bulgaria: INCOMA Ltd. Shoumen, BULGARIA, pp. 179–188. URL: https://aclanthology.org/R13-1024.

Domingo, Miguel and Francisco Casacuberta (2018). "Spelling Normalization of Historical Documents by Using a Machine Translation Approach". In: *Proceedings of the 21st Annual Conference of the European Association for Machine Translation*. Universitat d'Alacant, Alacant, Spain, pp. 129–137.

Eisenstein, Jacob (2013a). "Phonological Factors in Social Media Writing". In: *Proceedings of the Workshop on Language Analysis in Social Media*. Atlanta, Georgia: Association for Computational Linguistics, pp. 11–19. URL: https://aclanthology.org/W13-1102.

Eisenstein, Jacob (2013b). "What to do about bad language on the internet". In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational Linguistics, pp. 359–369. URL: https://aclanthology.org/N13-1037.

Etxeberria, Izaskun, Iñaki Alegria, Larraitz Uria, and Mans Hulden (2016). "Evaluating the Noisy Channel Model for the Normalization of Historical Texts: Basque, Spanish and Slovene". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), pp. 1064–1069. URL: https://aclanthology.org/L16-1169.

Gao, Qin and Stephan Vogel (2008). "Parallel Implementations of Word Alignment Tool". In: *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*. Columbus, Ohio: Association for Computational Linguistics, pp. 49–57. URL: https://aclanthology.org/W08-0509.

Gehring, Jonas, Michael Auli, David Grangier, and Yann Dauphin (2017). "A Convolutional Encoder Model for Neural Machine Translation". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, pp. 123–135. DOI: 10.18653/v1/P17-1012. URL: https://aclanthology.org/P17-1012.

Hämäläinen, Mika, Khalid Alnajjar, and Tuuli Tuisk (2022). "Help from the Neighbors: Estonian Dialect Normalization Using a Finnish Dialect Generator". In: *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*. Association for Computational Linguistics, pp. 61–66. DOI: 10.18653/v1/2022.deeplo-1.7. URL: https://aclanthology.org/2022.deeplo-1.7.

Hämäläinen, Mika, Niko Partanen, and Khalid Alnajjar (2020a). "Normalization of Different Swedish Dialects Spoken in Finland". In: *GeoHumanities'20: Proceedings of the 4th ACM SIGSPATIAL Workshop on Geospatial Humanities*. United States: ACM, pp. 24–27. DOI: 10.1145/3423337.3429435.

Hämäläinen, Mika, Niko Partanen, Khalid Alnajjar, Jack Rueter, and Thierry Poibeau (2020b). "Automatic Dialect Adaptation in Finnish and its Effect on Perceived Creativity". In: *Proceedings of the 11th International Conference on Computational Creativity (ICCC'20)*. Portugal: Association for Computational Creativity, pp. 204–211.

Hämäläinen, Mika, Tanja Säily, Jack Rueter, Jörg Tiedemann, and Eetu Mäkelä (2018). "Normalizing Early English Letters to Present-day English Spelling". In: *Proceedings of the Second Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social*

Sciences, Humanities and Literature. Santa Fe, New Mexico: Association for Computational Linguistics, pp. 87–96. URL: https://aclanthology.org/W18-4510.

Heafield, Kenneth, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn (2013). "Scalable Modified Kneser-Ney Language Model Estimation". In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Sofia, Bulgaria: Association for Computational Linguistics, pp. 690–696. URL: https://www.aclweb.org/anthology/P13-2121.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Computation* 9.8. MIT Press, pp. 1735–1780. ISSN: 1530-888X. Cambridge, MA.

Ikeda, Taishi, Hiroyuki Shindo, and Yuji Matsumoto (2016). "Japanese Text Normalization with Encoder-Decoder Model". In: *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 129–137. URL: https://aclanthology.org/W16-3918.

Institute for the Languages of Finland (2014). *Suomen kielen näytteitä – Samples of Spoken Finnish [online-corpus], version 1.0*. Helsinki. URL: http://urn.fi/urn:nbn:fi:lb-201407141.

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*. Vancouver, Canada: Association for Computational Linguistics, pp. 67–72. URL: https://www.aclweb.org/anthology/P17-4012.

Koehn, Philipp et al. (2007). "Moses: Open Source Toolkit for Statistical Machine Translation". In: *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*. Prague, Czech Republic: Association for Computational Linguistics, pp. 177–180. URL: https://aclanthology.org/P07-2045.

Kondrak, Grzegorz and Bonnie Dorr (2004). "Identification of Confusable Drug Names: A New Approach and Evaluation Methodology". In: *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*. Geneva, Switzerland: COLING, pp. 952–958. URL: https://aclanthology.org/C04-1137.

LeCun, Yann and Yoshua. Bengio (1995). “Convolutional Networks for Images, Speech, and Time-Series”. In: *The Handbook of Brain Theory and Neural Networks*.

Levenshtein, Vladimir I. (1966). “Binary codes capable of correcting deletions, insertions, and reversals”. In: *Soviet Physics Doklady* 10.8, pp. 707–710.

Lindén, Krister, Erik Axelson, Senka Drobac, Sam Hardwick, Juha Kuokkala, Jyrki Niemi, Tommi Pirinen, and Miikka Silfverberg (2013). “HFST—a System for Creating NLP Tools”. In: *Systems and Frameworks for Computational Morphology*. Ed. by Cerstin Mahlow. Ed. by Michael Piotrowski. Communications in Computer and Information Science. Springer-Verlag, pp. 53–71. ISBN: 978-3-642-40485-6. DOI: 10.1007/978-3-642-40486-3_4.

Ljubešić, Nikola, Darja Fišer, Tomaž Erjavec, Jaka Čibej, Dafne Marko, Senja Pollak, and Iza Škrjanec (2015). “Predicting the Level of Text Standardness in User-generated Content”. In: *Proceedings of the International Conference Recent Advances in Natural Language Processing*. Hissar, Bulgaria: INCOMA Ltd. Shoumen, BULGARIA, pp. 371–378. URL: https://aclanthology.org/R15-1049.

Ljubešić, Nikola, Katja Zupan, Darja Fišer, and Tomaž Erjavec (2016). “Normalising Slovene data: historical texts vs. user-generated content”. In: *Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016), volume 16 of Bochumer Linguistische Arbeitsberichte*. Bochum, Germany, pp. 146–155.

Luong, Thang, Hieu Pham, and Christopher D. Manning (2015). “Effective Approaches to Attention-based Neural Machine Translation”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, pp. 1412–1421. DOI: 10.18653/v1/D15-1166. URL: https://aclanthology.org/D15-1166.

Lusetti, Massimo, Tatyana Ruzsics, Anne Göhring, Tanja Samardžić, and Elisabeth Stark (2018). “Encoder-Decoder Methods for Text Normalization”. In: *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, pp. 18–28.

Matos Veliz, Claudia, Orphee De Clercq, and Veronique Hoste (2019). “Comparing MT Approaches for Text Normalization”. In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. Varna, Bulgaria: INCOMA

Ltd., pp. 740–749. DOI: 10.26615/978-954-452-056-4_086. URL: https://aclanthology.org/R19-1086.

Nigmatulina, Iuliia, Tannon Kew, and Tanja Samardžić (2020). "ASR for Non-standardised Languages with Dialectal Variation: the case of Swiss German". In: *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*. Barcelona, Spain (Online): International Committee on Computational Linguistics (ICCL), pp. 15–24. URL: https://aclanthology.org/2020.vardial-1.2.

Och, Franz Josef and Hermann Ney (2003). "A Systematic Comparison of Various Statistical Alignment Models". In: *Computational Linguistics* 29.1, pp. 19–51. DOI: 10.1162/089120103321337421. URL: https://aclanthology.org/J03-1002.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu (2002). "Bleu: a Method for Automatic Evaluation of Machine Translation". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://aclanthology.org/P02-1040.

Partanen, Niko, Mika Hämäläinen, and Khalid Alnajjar (2019). "Dialect Text Normalization to Normative Standard Finnish". In: *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*. Hong Kong, China: Association for Computational Linguistics, pp. 141–146.

Pettersson, Eva (2016). "Spelling Normalisation and Linguistic Analysis of Historical Text for Information Extraction". PhD thesis. Uppsala: Uppsala University, Department of Linguistics and Philology.

Pettersson, Eva, Beáta Megyesi, and Joakim Nivre (2013a). "Normalisation of Historical Text Using Context-Sensitive Weighted Levenshtein Distance and Compound Splitting". In: *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013)*. Oslo, Norway: Linköping University Electronic Press, Sweden, pp. 163–179. URL: https://aclanthology.org/W13-5617.

Pettersson, Eva, Beáta Megyesi, and Joakim Nivre (2014). "A Multilingual Evaluation of Three Spelling Normalisation Methods for Historical Text". In: *Proceedings of the 8th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaT-*

eCH). Gothenburg, Sweden: Association for Computational Linguistics, pp. 32–41. DOI: 10.3115/v1/W14-0605. URL: https://aclanthology.org/W14-0605.

Pettersson, Eva, Beáta Megyesi, and Jörg Tiedemann (2013b). "An SMT Approach to Automatic Annotation of Historical Text". In: *In Proceedings of the Workshop on Computational Historical Linguistics at NODALIDA 2013, NEALT Proceedings Series 18; Linköping Electronic Conference Proceedings 87*, pp. 54–69.

Piotrowski, Michael (2012). *Natural Language Processing for Historical Texts*. Synthesis Lectures on Human Language Technologies. San Rafael: Morgan Claypool Publishers. ISBN: 1608459462.

Popović, Maja (2015). "chrF: character n-gram F-score for automatic MT evaluation". In: *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Lisbon, Portugal: Association for Computational Linguistics, pp. 392–395. DOI: 10.18653/v1/W15-3049. URL: https://aclanthology.org/W15-3049.

Porta, Jordi, José Luis Sancho Sánchez, and Javier Gómez Gómez (2013). "Edit Transducers for Spelling Variation in Old Spanish". In: *Proceedings of the Workshop on Computational Historical Linguistics at NODAL-IDA2013, NEALT Proceedings Series18/LinköpingElectronic Conference Proceedings 87*. Linköping University Electronic Press, pp. 70–79.

Rayson, Paul, Dawn Archer, and Nicholas Smith (2005). *VARD versus WORD: A comparison of the UCREL variant detector and modern spell checkers on English historical corpora*.

Saito, Itsumi, Jun Suzuki, Kyosuke Nishida, Kugatsu Sadamitsu, Satoshi Kobashikawa, Ryo Masumura, Yuji Matsumoto, and Junji Tomita (2017). "Improving Neural Text Normalization with Data Augmentation at Character- and Morphological Levels". In: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, pp. 257–262. URL: https://aclanthology.org/I17-2044.

Samardžić, Tanja, Yves Scherrer, and Elvira Glaser (2015). "Normalising orthographic and dialectal variants for the automatic processing of Swiss German". In: *Proceedings of the 7th Language and Technology Conference*.

Sánchez-Martínez, Felipe, Isabel Martínez-Sempere, Xavier Ivars-Ribes, and Rafael C. Carrasco (2013). *An open diachronic corpus of historical Spanish: annotation criteria and automatic modernisation of spelling*.

Scherrer, Yves and Tomaž Erjavec (2013). "Modernizing historical Slovene words with character-based SMT". In: *Proceedings of the 4th Biennial International Workshop on Balto-Slavic Natural Language Processing*. Sofia, Bulgaria: Association for Computational Linguistics, pp. 58–62. URL: https://aclanthology.org/W13-2409.

Tang, Gongbo, Fabienne Cap, Eva Pettersson, and Joakim Nivre (2018). "An Evaluation of Neural Machine Translation Models on Historical Spelling Normalization". In: *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, pp. 1320–1331. URL: https://aclanthology.org/C18-1112.

Thurlow, Crispin, Laura Lengel, and Alice Tomic (2004). *Computer Mediated Communication: Social Interaction and the Internet*. Thousand Oaks, California: Sage Publications. ISBN: 0-7619-4953-4.

Torunoğlu, Dilara and Gülşen Eryiğit (2014). "A Cascaded Approach for Social Media Text Normalization of Turkish". In: *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*. Gothenburg, Sweden: Association for Computational Linguistics, pp. 62–70. DOI: 10.3115/v1/W14-1308. URL: https://aclanthology.org/W14-1308.

Tsarfaty, Reut, Djamé Seddah, Sandra Kübler, and Joakim Nivre (2013). "Parsing Morphologically Rich Languages: Introduction to the Special Issue". In: *Computational Linguistics* 39.1, pp. 15–22. DOI: 10.1162/COLI_a_00133. URL: https://aclanthology.org/J13-1003.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is All You Need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates, Inc., pp. 6000–6010. ISBN: 9781510860964.

Vilkuna, Maria (2014). *SKN-aineiston yleiskielistys*.

Wu, Shijie, Ryan Cotterell, and Mans Hulden (2021). "Applying the Transformer to Character-level Transduction". In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics, pp. 1901–1907. DOI: 10.18653/v1/2021.eacl-main.163. URL: https://aclanthology.org/2021.eacl-main.163.

Yolchuyeva, Sevinj, Géza Németh, and Bálint Gyires-Tóth (2018). "Text normalization with convolutional neural networks". In: *International journal of speech technology* 21.3, pp. 589–600. ISSN: 1381-2416.