KU LEUVEN

Natural Language Processing on Limburgish: a non-standardized, low-resource language

Andreas Simons

Thesis submitted for the degree of Master of Science in Artificial Intelligence, option Engineering and Computer Science

Supervisor:
prof. dr. Tim Van de Cruys

Assessors:
prof. dr. Marie-Francine Moens
prof. dr. Hugo Van Hamme

Assistant-supervisors:
prof. dr. Stefano De Pascale
dr. Karlien Franco

Academic year 2022 – 2023

Preface

I would like to thank Tim Van de Cruys, Stefano De Pascale and Karlien Franco for accepting this thesis proposal, their guidance throughout and their feedback. I would also like to thank my friends and family for their support.

Finally, a special shout out to the Artes Erasmushuis library for their wealth of non-digitized books on Limburgish, which they generously let me keep months past the due date.

Andreas Simons

Contents

Abstract

We study the development of NLP applications for non-standardized, low-resource languages with a focus on Limburgish. A large geotagged Limburgish dictionary is processed to be used in further NLP pipelines and datasets are generated for dialect normalization and dialect machine translation tasks. We study different deep learning architectures and evaluate different metrics on extremely granular, short-text dialect identification, dialect machine translation, and dialect normalization.

A novel modification to the transformer architecture is proposed, which embeds continuous linguistic information. We specify the study of this transformer adaption to the embedding of geographic coordinates, which can be leveraged into successful dialect normalization and extremely granular dialect machine translation and outperforms the normal transformer architecture in the task of dialect normalization. We finally compare our results to notions and maps in traditional Limburgish dialectology.

1 Introduction

Natural Language Processing or *NLP* is one of the most active fields in Artificial Intelligence research and its recent advances have caused a global shock wave of publications and media interest. State-of-the-art deep learning architectures such as the transformer (Vaswani et al., 2017), (Liu and et al., 2023) and BERT (Devlin et al., 2019) outperform previous decades of Machine Learning applications and have resulted in very human-like language models. Most research, however, is conducted on high-resource languages such as English, French, or Chinese, leaving most of the world's languages out of cutting-edge NLP applications. As a response, the field of low-resource NLP is gaining increasing traction. More extreme is the category of NLP on non-standardized, low-resource languages. Apart from the typical problems associated with low-resource NLP such as smaller datasets, these languages could have additional unique problems such as a lack of spelling normalization and internally varying phonology, morphology, vocabulary and grammar. Research on non-standardized, low-resource languages is scarce, although examples of related research can be found for Swiss-German (Honnet et al., 2018), Arabic (Shoufan and Alameri, 2015), (Muller et al., 2020) or Alsatian (Millour and Fort, 2019). A recent review of available datasets for most Germanic, non-standardized, low-resource languages can be found in (Blaschke et al., 2023).

Few NLP applications have been developed on Limburgish, to the best of our knowledge we have found two corpora: dumps of the Limburgish Wikipedia and one developed by the Limburgish Academy (Michielsen Tallman et al., 2017). Additionally, two relatively small corpora exist: Ubuntu localization files and Web Crawl data (Blaschke et al., 2023). Within the list provided by (Blaschke et al., 2023), Limburgish is one of the languages with the fewest available data, together with Pennsylvania Dutch, Palatine German, Colognian, Saterland, and North Frisian. Published research constitutes the use of Limburgish on social media (Jongbloed-Faber et al., 2017), the relation between geography and loanwords (Franco et al., 2019b), and lexical diversity (Franco et al., 2019a). Recently, Meta's *No Language Left Behind* included Limburgish through its *FLORES-200* dataset, although only featuring the Maastricht dialect. The latter has led to some limited multilingual applications on Hugging Face (Hugging Face). Recently, OpenAI's ChatGPT (OpenAI, 2021) has also crawled enough Limburgish data to be able to code-switch when explicitly asked to.

In this thesis we will study the development of some basic NLP tools, which can be employed in larger NLP on Limburgish and by extension on other non-standardized, low-resource languages. Due to a lack of large corpora, and the severe lack of normalization in the existing corpora, we will not use a corpus, but rather an extensive geolocated Limburgish dictionary. First, a general introduction to Limburgish will be provided in Chapter 1. In Chapter 2, we provide an introduction to the history, structure and area covered by this dictionary. In Chapter 3, we extensively study the quality of this dictionary and develop the necessary preprocessing steps in order for any future NLP application on this dataset to succeed.

In Chapter 4, we will study and compare various deep learning techniques and evaluation metrics for the purpose of short-text dialect identification on Limburgish. Due to the large quantity of geotagged dialect entries, this is to the best of our knowledge the most granular dialect identification study and the only regression study so far that aims to predict the exact coordinates of a dialect input. We will also perform a detailed linguistic analysis, including notions of traditional dialectology, to explain the performance of the dialect identification model.

In Chapter 5, we introduce a new transformer-based architecture that embeds continuous linguistic data. We will use this architecture to embed the coordinates of dialect words to study the development of a dialect normalization model and a dialect machine translation model. We generate normalization and dialect translation datasets and study different evaluation metrics. A detailed linguistic analysis is again performed on both tasks, with a particular focus on the generation of language variation maps and their comparison to traditional maps in Limburgish dialectology.

Conventions used in this thesis:
Whenever a Dutch word is presented in quotes, e.g. "vlaai", it refers to a concept or keyword in the WLD

(see Chapter 3). Whenever a Limburgish word is presented in a phonetic notation, it is typeset in the Kurinto Mono font to properly display all phonetic characters and diacritics, e.g. vlõi̯ (a Limburgish pie). All Limburgish words are accompanied by an English translation between brackets.

The word locality is used for any unique location (city, town, hamlet, etc.) with a separate kloeke code (see Chapter 3). The term "dialect" refers to the variety of Limburgish spoken in that specific geographic locality (see Chapter 2).

2 Limburgish

Limburgish is a West-Germanic language spoken by approximately $1.2-1.5$M people (Limburgish Academy, a) in Belgium, the Netherlands and Germany. Some discussion exists about the exact outer boundaries of the Limburgish language area as they run across national and official linguistic borders (see the sections below). Limburgish is protected as an official minority language in the Netherlands and as an endogenous language by the French-speaking community of Belgium (Limburgish Academy, a), although the number of its speakers is decreasing (Veldeke, 2021) and it is classified as a vulnerable language (Moseley, 2010).

2.1 Historical development

Limburgish is particularly interesting for the study of dialect identification and dialect translation as the various Limburgish dialects follow highly predictable patterns. Even though Limburgish is a low-resource language, it has been extensively studied in the past by dialectologists. Here we review the linguistic patterns that can be found within Limburgish, along with their historical causes:

The Limburgish area is the westernmost region in Europe that took part in the High German consonant shift, whose outer boundary is the so-called *Uerdinger* line and corresponds with the isophone *ik→ich* (Goossens, 1965). This boundary is therefore often taken to be the northern and western demarcation of the Limburgish language area. Politically this region was isolated from the rest of the low countries as the western half (modern Belgian Limburg) belonged to the Prince-Bishopric of Liège (Pauwels and Morren, 1960), which resisted unification with the medieval Netherlands and did not contribute to the modern Dutch language (Belemans and Keulen, 2004). Incorporated in the Holy Roman Empire, this region was linguistically oriented towards the east and under strong influence of High German, specifically the city of Cologne, in a process that is known as "Cologne Expansion". The city of Cologne radiated linguistic and cultural influence for over a millennium in almost concentric isophones into a region that is known as the "Rhenish fan" (Figure 1):

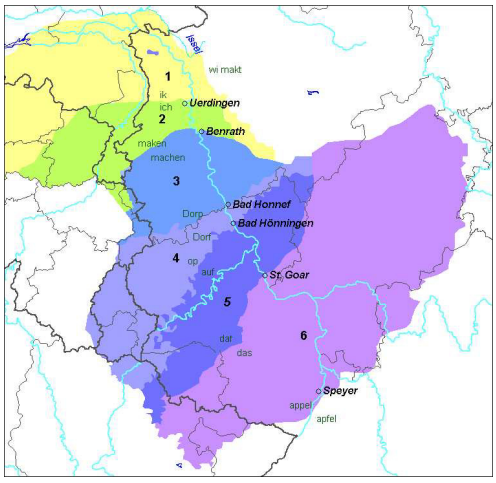


Figure 1: Rhenish fan by (Erren, 2010). 1 is Low Franconian, 2 is Limburgish, 3 is Ripuarian, 4 and 5 are Moselle Franconian and 6 is Rhenish Franconian.

One major isogloss in the Rhenish fan, typically associated with isophones *k→ch*, *p→pf* and *t→ts*, is called the Benrather line and separates the Limburgish area from the Ripuarian area, as it is taken to be the eastern border of Limburgish. This region, between the Uerdinger and Benrather isoglosses, and in the south demarcated by the Germanic-Romance transition is often defined to be the Limburgish language area (Goossens, 1965). It should be noted that, although frequently visualized as such, these isoglosses are not

clear borders, but rather transitions. The determination of isoglosses frequently depends on the combination of many isophones, typically derived from administrative boundaries of municipalities into idealized borders. At most points, the Uerdinger line is approximately 1 km wide, whereas the Gete line (see below) is tens of kilometers wide in some parts. In reality, there is likely a more continuous nature to these isophones, depending on the time and the person whose dialect was recorded. In Figure 2 we therefore purposefully represent the isoglosses as wide lines.

In 1288, the county of Loon (modern Belgian Limburg), the duchy of Brabant, and the burghers of Cologne won a decisive battle against the bishops of Cologne in the Battle of Worringen, which became a turning point in the influence of Cologne on the Limburgish area. The Cologne Expansion did not immediately stop, but due to this political event a large portion of present Dutch Limburg came under the influence of the dukes of Brabant and the Brabantian cities of Leuven and Brussels started a period of influence on Limburgish called the "Brabantian Expansion" (Goossens, 1988). As a result of these two periods of western and eastern influence, many features of Limburgish dialects vary in a horizontal direction, rather than a vertical direction.

Finally, the southern demarcation is typically taken to be the Germanic-Romance language "border", although it should be noted that this does not constitute a clear border or coincide with the Belgian Dutch-French language border, which was politically determined. Particularly in the northeast of Liège we can find Germanic dialects (Weijnen et al., 1983), and in other places across the official language border (see Figure 3 in 4.1) dialects are spoken that are classified as Limburgish, since they fall between the Benrather and Uerdinger lines. It is conjectured (Leenen, 1938) that certain isophones and features of Limburgish extend to Walloon and vice versa, along with mutual influence in vocabulary. Additionally, it should be noted that Belgian Limburg and Liège were historically united in a multilingual state, with intense Germanic-Romance contact (Wintgens, 1994).

2.2 Characteristics

As part of the West-Germanic languages, Limburgish has common characteristics with Dutch, German, Ripuarian, etc. but it also contains a large number of unique features. Limburgish has a very large phonology (see Chapter 5 for a detailed discussion): the Weerter dialect has 28 vowels in a phonetic five-height system and possibly has the largest vowel system of any language in the world, as compared to 12 vowels in standard Dutch, or 15 vowels in Californian English. One of the characterizing features of all Limburgish dialects is the use of tonality as a phoneme (Bakkes et al., 2007), which it has in common with most Ripuarian dialects.

Limburgish strictly differentiates three genders, as opposed to standard Dutch which generally only differentiates neuter from male and female (Bakker, 1997). Plurals and diminutives are mostly formed through umlautization, as in Middle and High German, e.g. *tak→tek* (branch) and *man→menneke* (man). Most varieties of Limburgish still have a gerund and some a grammatical subjunctive (Bakkes et al., 2007), (Klein, 2020). Verb conjugation differs from both German and Dutch and personal pronouns are heavily influenced by High German (see Table 1). Limburgish vocabulary is generally aligned with Rhenish-German vocabulary (Limburgish Academy, a) but also contains many common cognate words with standard Dutch and German, including archaic words that have fallen in disuse in both languages. Additionally, Limburgish vocabulary is strongly influenced by Walloon (Bakkes et al., 2007), (Goossens, 1996).

	Limburgish (Maastricht)	Standard German	Standard Dutch
I eat	iech eet	ich esse	ik eet
you eat	diech its	du isst	jij eet
he/she/it eats	heer/zie/'t it	er/sie/es isst	hij/zij/het eet
we eat	veer ete	wir essen	wij eten
you eat	geer et	ihr esst	jullie eten
they eat	zie ete	sie essen	zij eten

Table 1: Conjugation of "to eat", the Limburgish conjugation is sourced from Maastricht (Weijenberg).

2.3 Classification of dialect groups

Apart from the Uerdinger and Benrather isoglosses, other major isoglosses and isophone bundles are used to classify the Limburgish area in different language groups. Most classifications consider the Panninger line, Panninger sideline and Tonality line. The Panninger line consists of many isophones where s→š if the s appears at the beginning of a word, in front of consonants such as p, t, l, and m, e.g. *sjpele, sjlaope, zjwart* instead of *spele, slaope, zwart* (to play, to sleep, black) (Bakkes et al., 2007). The Panninger sideline is associated with the isophone sx→š in words such as *school→sjool* (school) and *schrieve→sjrieve* (to write). West of the Panninger sideline, the singular second-person pronoun becomes a variant of *gae* or *gij* instead of *du* and *dich*. Finally, west of the Tonality line, tones are not used as phonemes anymore.

An additional isogloss, or more specifically isophone bundle, is the Gete line. (Pauwels and Morren, 1960) found that 26 isophones are bundled compactly in a region closely approximating the old border between the duchy of Brabant and the Prince-Bishopric of Liège and therefore also closely approximates the border between present-day Limburg and Flemish-Brabant. They conjecture that this is an example of a political border turned into a linguistic border. This Gete line is therefore sometimes proposed as an alternative western boundary of the Limburgish language area, although it should be noted that the width of the isoglosses that make up the Gete line are especially broad in the north and south of Belgian Limburg. We have visualized all these isoglosses and bundles in Figure 2. Based on these isoglosses, seven groups can be differentiated within the Limburgish language area, we will briefly review these groups and their most prominent characteristics, as sourced from (Bakkes et al., 2007):

Ripuarian:
Ripuarian constitutes all dialects east of the Benrather line and is therefore not traditionally considered to be Limburgish. This leads to some localities in Dutch Limburg such as Kerkrade not being classified as Limburgish. The Benrather line cuts through the northeast of Liège, leading to localities such as Kelmis and Eupen being classified as Limburgish, while e.g. Raeren is considered Ripuarian. A prominent characteristic east of the Benrather line, apart from the ones we have described above, is the isophone x→j in words such as *good→jood* (good).

East-Limburgish:
Demarcated by the Panninger line in the west, Benrather in the east, and Uerdinger in the north, we find the East-Limburgish dialects. Here we can find "palatalization" ("mouillering") in words such as *hond→hondj* (dog) or *wandele→wanjele* (to walk). Another prominent characteristic is "velarization" of words such as *hand→hank* (hand) or *vinde→vinge* (to find). Northeast of the Uerdinger line, these dialects cross the Dutch-German language border into a region far beyond Belgian and Dutch Limburg, including places such as Krefeld and Mönchengladbach (Goossens, 1965) (see Figure 1). There is no linguistic reason to stop the Limburgish language area at the German border, and therefore sometimes these dialects are also called Limburgish, even though locally they are called "Südniederfränkisch". While in Belgium and the Netherlands the rule of thumb exists that a dialect is considered to be Limburgish if the sentence "ik maak" (I make) is translated as "ich maak" (matching the Benrather and Uerdinger isoglosses), a similar rule exists for "ech kall" (I speak) vs. "ek spräák" within Südniederfränkisch (LVR-Institut für Landeskunde und Regionalgeschichte).

Central-Limburgish:
Demarcated by the Panninger sideline in the west, Panninger line in the east, and Uerdinger line in the north we find the Central-Limburgish dialects. Most of their characteristics follow directly from the isoglosses.

West-Limburgish:
Demarcated by the Panninger sideline in the east, the Tonality line in the southwest, and Uerdinger line in the north and northwest, we find West-Limburgish. As West-Limburgish lies west of the Panninger sideline, it does not use *du/dich* as singular second-person pronouns anymore.

West-Limburgish transitional dialects
West of the Tonality line but below the Uerdinger line, we find the westernmost Limburgish dialects, which transition to Brabantian. The Gete bundle lies mostly in this region and it is therefore one of the strongest transitions in the Limburgish area. This region does not have tonality anymore and uses typical Brabantian characteristics such as r-deletion in words as *zwart→zwet* (black) or *kaars→kaes* (candle). Prominent isophones as part of the Gete bundle are long monophthongs where typically Limburgish has diphthongs, e.g. *hoes→haas* (house) or *doezend→duuzend* (thousand).

Brabantian-Limburgish
North of the Uerdinger line, we find a transition zone of Brabantian into Limburgish, the so-called Brabantian-Limburgish dialects. These dialects do not have tonality nor use pronouns such as *ich, mich*. The Brabantian r-deletion exists here as well. Prominent characteristics include the isophones *w→f* in words such as *snief→sneeuw* (snow) or the use of glottislag `ʔ` in words such as `baʔə` instead of `bakə` (to bake).

Kleverlandic
Northeast of the Uerdinger line we find the Kleverlandic dialects, which also do not have pronouns such as *ich, mich*, except for a region around Venlo, the so-called "mich-kwartier", which does use *ik* and *mich* simultaneously. Typical characteristics include monophthongs where most Limburgish dialects have diphthongs such as *huus* instead of *hoes* (house) or *tied* instead of *tieëd* (time).

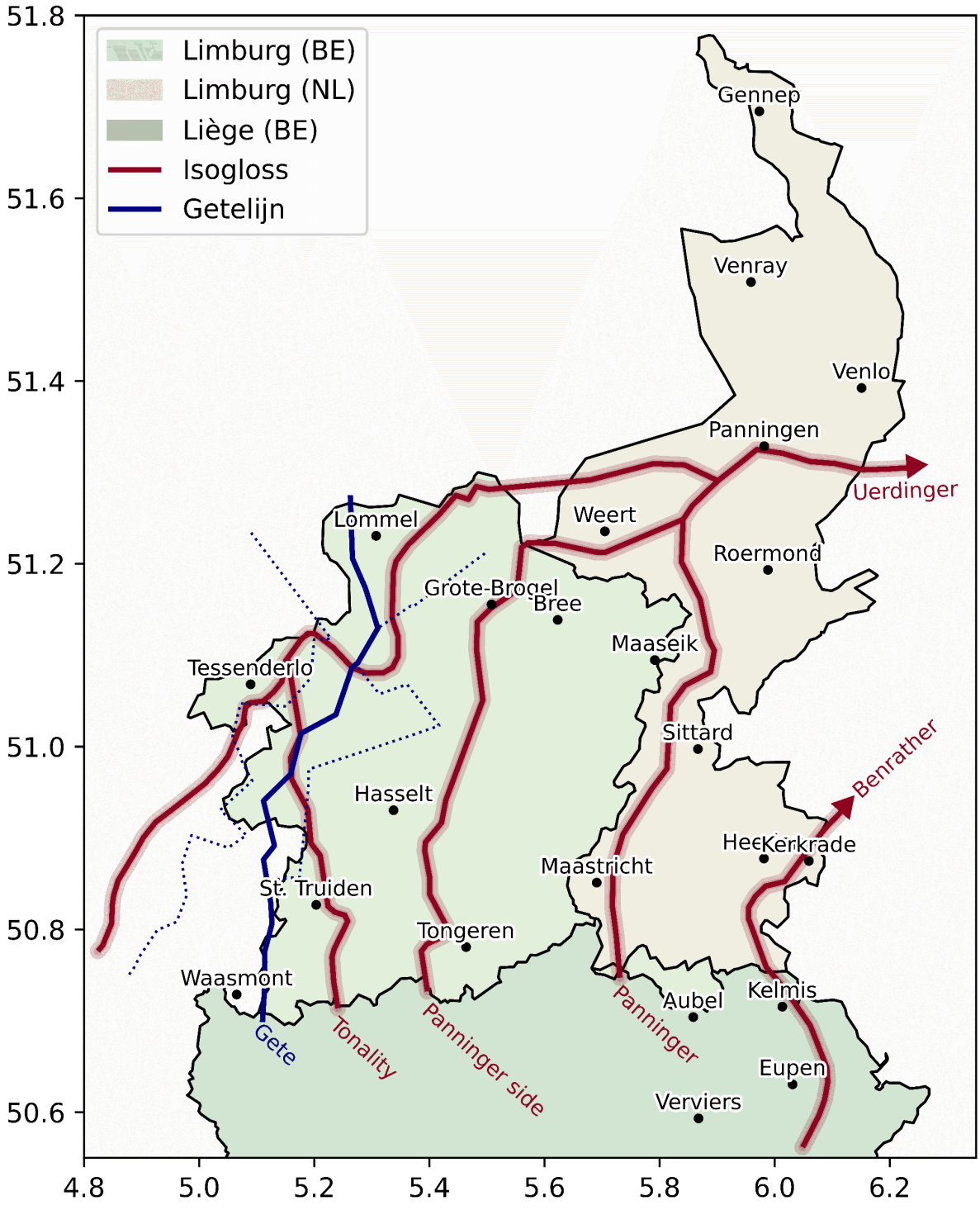


Figure 2: All major isoglosses in the Limburgish language area, the provincial borders, and the Gete bundle. The dotted lines next to the Gete line represent the maximum width of the isophones that constitute the Gete bundle. The isoglosses are sourced from (Bakkes et al., 2007), while the extension of the Uerdinger line beyond Belgian Limburg is sourced from (Goossens, 1965) and the Gete line from (Pauwels and Morren, 1960).

3 Woordenboek van de Limburgse Dialecten

The *Woordenboek van de Limburgse Dialecten* (Dictionary of the Limburgish dialects) (Weijnen et al., 1983) or *WLD* is a series of books that constitute an onomasiological dictionary for the dialects of Limburgish, published between 1983 and 2008. The dictionary is onomasiological in the sense that it is indexed along concepts and keywords, where "Dutchified" words represent the lexical variety of Limburgish cognates. The WLD consists of 39 volumes or books, each covering a semantic domain but grouped into three parts: agrarian terminology (plowing, cattle, …), non-agrarian professional terminology (bakery, mining, smithing, …), and general terminology (health, sexuality, religion, …). The WLD is a large lexicon (van Hout et al.), containing approximately 17k concepts and 137k keywords for a total of 1.8M entries geographically spread over 1000 places in mostly Belgian and Dutch Limburg and the north of Liège.

3.1 Origin of the WLD

The first idea for creating a general dictionary that covers all Limburgish dialects possibly originated in 1914 when Nijmegen professors J. Schrijnen, J. van Ginneken and J.J. Verbeeten systematically collected detailed surveys, the so-called *SGV* survey (Schrijnen et al., 1914)[1], from 95 places in Dutch Limburg, asking translations of vocabulary, specific phenomena in phonetics (e.g. palatalization), grammar (e.g. conjugation or existence of a gerund) and folklore. In total, nearly 200k entries were collected (Kruijsen et al., 2004) with the purpose of determining the continuation of various West-Germanic isoglosses that Jacob Ramisch had described a few years earlier in his *Studien zur niederrheinischen Dialektgeographie*, but ended right at the border between Dutch Limburg and Germany.

It was not until 1933 (Weijnen et al., 1983) when W. Roukens published an article in *Veldeke* (a journal for Limburgish literature and dialectology) suggesting the idea of a (Dutch) Limburgish dictionary. Four years later this idea was formalized in Roukens' dissertation (Roukens, 1937), under the supervision of van Ginneken, with the help of Leuven professors L. Grootaers and J.L. Pauwels of the *Zuidnederlandse Dialectcentrale* or *ZND*. The ZND had been collecting similar surveys as SVG since 1922 and curated the Willems survey of 1885, the first modern language survey covering Belgium, the Netherlands, and the adjacent Rhineland region. Further in 1942, Grootaers called for a Limburgish dictionary featuring both provinces, citing the large number of already collected data and the distant, isolated language of the province (Weijnen et al., 1983). Finally, in 1960 an article titled *Het Limburgs Woordenboek* (Roukens, 1960) (The Limburgish Dictionary) was published in *Veldeke* by Roukens, J.L. Pauwels, and A. Weijnen, which would be the foundation of the WLD. In this article, the authors for the first time strongly advocated for a joint dictionary of both Belgian and Dutch Limburg, based on linguistic and sociocultural ties.

A. Weijnen, who had two years prior initiated the project of the *Woordenboek van de Brabantse Dialecten*, a dictionary of the Brabantian dialects (Weijnen et al., 1983), directed the formation: Roukens' earlier collected data was supplemented over the next years by the collection of the same surveys in Belgian Limburg and the northeast of Liège by the ZND. After a few years, Jan Goossens, who had earlier collected extensive data on agricultural terms in Belgian Limburg, joined the curation of the WLD and included his agricultural dataset. Together with various (unpublished) dialect dictionaries, data from the P.J. Meertensinstituut, the university of Liège, university of Ghent and the *Reeks Nederlandse Dialektatlassen* or *RND*, the contents of the WLD were curated until 1983 when the first book was published.

3.2 Structure of the WLD

The entries in the WLD are not fully synchronic: the Willems data was collected around the end of the 19th century, SGV in 1914, ZND from 1922, and the equivalent of Roukens' data in Belgian Limburg from the 1960s

[1] For a recent review, see Kruijsen (2006) (Kruijsen, 2006).

onwards. This means that parts of the WLD were collected during a major linguistic shift: between 1950 and 1980 a period of hyperstandardization (Van Hoof and Jaspers, 2012) in Belgium sought to promote standard Dutch and stigmatize any deviation. Simultaneously there had been indications that the state border between Belgium and the Netherlands was slowly becoming a linguistic border (Weijnen and Van Coetsem, 1957). A major economic shift was also taking place: traditional professions and artisans were rapidly declining and much of their jargon was disappearing. It was therefore decided to start the WLD volumes in order of urgency: agrarian terms, professional terms, and only then general vocabulary. Each volume is therefore based on a specific semantic field, with a focus on terminology that varies geographically (Weijnen et al., 1983). The WLD therefore does not function as a "regular" or even frequency dictionary, but focuses on capturing the geographic diversity of Limburgish. The volumes are indexed as follows:

WLD I: Agrarian terminology

- I.1 Fertilizing and plowing
- I.2 Harrowing and dragging
- I.3 Pasture cultivation
- I.4 Cultivation of cereal crops
- I.5 Cultivation of tubers and other crops
- I.6 Farm, commercial buildings
- I.7 Farm, yard and vegetable garden
- I.8 Farmlands
- I.9 The horse
- I.10 Horse harness
- I.11 Cattle, milk and butter, animal husbandry in general
- I.12 Small livestock and poultry
- I.13 Agricultural vehicles

WLD II: Non-agrarian professional terminology

- II.1 Butcher and baker
- II.2 Beer brewer and syrup maker
- II.3 Miller
- II.4 Turf extractor and ore miner
- II.5 Miner
- II.6 Beekeeper, straw or bentgrass weaver
- II.7 Tailor, seamstress, hand spinner, hand weaver, rope maker, hat maker
- II.8 Potter, brick maker, tile maker, vitrified clay pipe industry
- II.9 Bricklayer, carpenter, roofer, plumber, plasterer, house painter
- II.10 Shoemaker, saddle maker/harness maker
- II.11 Blacksmith, plumber, coppersmith
- II.12 Lumberjack, carpenter, joiner, wheel and wagon maker, cooper, clog maker, basket maker

WLD III: General terminology

- III.1.1 The human body
- III.1.2 Movement and health
- III.1.3 Clothing and body care
- III.1.4 Personality and feelings
- III.2.1 The home
- III.2.2 Family and sexuality
- III.2.3 Food and drink
- III.3.1 Social behavior, school and education
- III.3.2 Party and entertainment
- III.3.3 Church and faith
- III.4.1 Birds
- III.4.2 Other animals
- III.4.3 Flora
- III.4.4 The material and abstract world

Each entry in a volume is primarily indexed by a concept which represents the semantic concept, e.g. "eekhoorn" (squirrel) and then keywords representing the lexical items corresponding to that concept: "eekkatsje, enkbeugel, enkpetje, hazel, …" These keywords correspond to either the closest cognate in standard Dutch or are newly constructed word from the etymology of the Limburgish word, creating a "Dutchified" version (Weijnen et al., 1983). The keywords do not account for any typical phenomena in Limburgish phonology or grammar such as the use of umlauts, diminutives, and plurals. The WLD does not provide any grammatical info per entry due to the high complexity of the task and the lack of collected data from the surveys. Plural forms, diminutives, gender, and declensions which tend to differ significantly from standard

Dutch are therefore not present in the WLD.

Per keyword, the various Limburgish dialects that correspond to this Dutchified keyword are listed, together with a geographic tag, the so-called *kloeke code*. The Limburgish words are transcribed in a custom "morpho-phonological" spelling: a combination of standard Dutch orthography where possible, supplemented by characters from the International Phonetic Alphabet. An example of a Limburgish word corresponding to the keyword *afvegen* is therefore not āvē̜gə but af$_>$vē̜gə(Weijnen et al., 1983). It should be noted that although the WLD outlines clear phonological rules, the various sources that form the WLD vary in quality: sometimes Dutch orthography is used that conflicts with IPA graphemes or important phonological information such as diacritics or tones are not represented. In other cases, the diacritics are present in the IPA but confusingly only signify that the sound differs from the corresponding Dutch phoneme (Weijnen et al., 1983). In other cases, different spelling systems such as the Veldeke spelling are used (see Chapter 2). For an overview of these conventions we refer to Appendix A.

The *kloeke code* serves as a geographic tag for each WLD entry. In 1926, G.G. Kloeke sought to standardize the manner in which linguistic data in Belgium, the Netherlands, the north of France, and the west of Germany was gathered and represented (Meertens Instituut). Contemporary linguists such as van Ginneken employed "closed maps", linguistic maps with clearly defined borders. He criticized these cartographic practices as they are suggestive and rely on interpretation. He further noted that these borders often rely on municipal or administrative borders and do not accurately reflect the linguistic data. Instead, he proposed that each municipality, village, or hamlet receives its own standardized code so that linguistic data can be represented in "open maps", scatter plots that more accurately reflect the granularity of the underlying data. The contemporary kloeke table was supplemented by the WLD editors with new localities: either missing localities or splitting kloeke codes if there exist significant linguistic differences within the original code. A full list of all kloeke codes in the WLD can be found in Appendix B.

The semantic indexing of the WLD, its geographic tagging through kloeke codes and its wide and extensive geographic covering makes the WLD a unique linguistic dataset in Europe and has allowed for studies on its lexical diversity (Franco et al., 2019a) and the influence of geography on loanwords (Franco et al., 2019b).

3.3 Area covered by the WLD

The editors of the WLD decided to include a geographical area that is larger (for Belgium and the Netherlands) than the traditional demarcation of the Limburgish dialects, based on the traditional isoglosses (Weijnen et al., 1983). The Limburgish dialects that continue after the Dutch-German state border are not included in the WLD. Originally it was decided that the idealized Gete line (Pauwels and Morren, 1960) (see Chapter 1) would form the western border between the Limburgish and Brabantian area for the purposes of distinguishing the WLD and *WBD* (Dictionary of Brabantian Dialects). To avoid arbitrary selection of isoglosses, the editors decided on an area that does not follow traditional isoglosses but includes the entirety of Belgian and Dutch Limburg as of the provincial borders in 1963. Additionally, it includes all localities that are east of the idealized Gete line in the former province Brabant (e.g. Rummen and Grazen) and therefore overlaps for some kloeke codes with the WBD. Later volumes of the WLD also arbitrarily include limited data on localities west of the Gete line in Brabant as well as localities that are between the Gete line and the border of Belgian Limburg, but are presently in Walloon-Brabant (such as Neerheylissem and Racour). Similarly, later volumes of the WLD include limited data of localities outside the border of Dutch Limburg, in North-Brabant and Gelderland.

The southern demarcation approximately follows the Belgian language border as officially adopted in 1963, although sometimes arbitrarily (Weijnen et al., 1983): the northeastern portion of the province Liège (e.g. Aubel, Limbourg, Welkenraedt, Eupen) is included in the WLD based on an assertion by Roukens and Grootaers, who had included this region in their earlier surveys. As previously mentioned, the eastern demarcation stops exactly at the state border between Germany and the Netherlands. This is a result of the onset of the WLD: to continue the isoglosses and surveys that Ramisch had abruptly stopped for the Rhine

region at the state border.

In conclusion, the area of the WLD covers a region larger than the traditional demarcation of Limburgish in the north and west but in the southwest follows the political Belgian language border of 1963 (with a few exceptions). The southern border further includes the northeast of Liège and in the east the political state border between Germany and the Netherlands. A map of the studied region can be found in the section below.

3.4 Digitization of the WLD

The volumes of the WLD were published in printed form, which makes any type of statistical analysis or application in Natural Language Processing unfeasible. In 2015, N. van der Sijs and R. van Hout digitized the first two parts of the WLD as part of the CARE project (*CurAtion and integration of REgional dictionaries*) (van Hout et al., 2018), the third part of the WLD (WLD III on general terminology) was edited digitally and did not need digitization. Using OCR they scanned and semi-automatically converted the WLD to a digital database. The extremely detailed use of diacritics in the WLD hindered the OCR process, which resulted in an original editor of the WLD, Joep Kruijsen, performing semi-automatic correction of the diacritics and phonetic symbols. During the digitization, it became apparent that many errors were present in the kloeke codes, either during the OCR process or already present in printed form.

The results of the digitization are available on `www.e-wld.nl`, together with the original printed editions of the WLD. For the purpose of this thesis, the WLD web portal was scraped using a Python script. Below in Table 2, we list a few examples of entries in the WLD after digitization for the concept "squirrel".

concept	keyword	dialect entry	kloeke code	volume	source
eekhoorn	eekhoorntje	eikheurke	L289p	III 4.2	SGV (1914)
eekhoorn	eekhoorntje	ekørəntšə	Q012p	III 4.2	ZND B2 (1940sq)
eekhoorn	enkbeugel	inkbi-jgel	L358p	III 4.2	/

Table 2: Sample entries of the digitized version of the WLD.

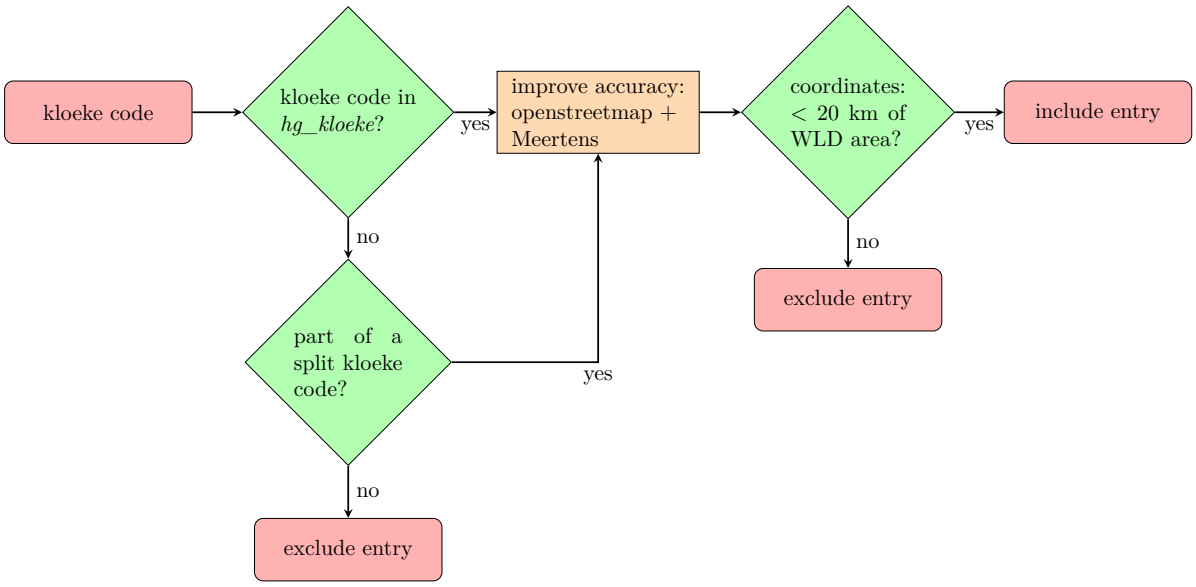
For the first two entries, it is clear how the keyword neglects the diminutive in Limburgish, both `š` and `k` are represented using Dutch *tj* orthography. In the third entry, the *i-j* is converted into *eu* to reflect the reversal of delabialization (`i`→`y`). The first dialect entry is written in Dutch orthography, the second in the morpho-phonological spelling of the WLD. The third entry follows the Veldeke spelling (see Appendix A). The third entry is without source as it does not come from either of the major surveys (SGV, ZND, RND, …) but from other archives or dialect dictionaries.

4 Preprocessing the WLD

The scraped dataset constructed from the digitized WLD contains 1 732 195 entries over 139 413 keywords and 17 121 concepts, which is different than the advertised figures (van Hout et al.). Before this dataset is used in large scale statistical analysis or NLP applications, it has to be further processed: the kloeke codes have to be converted into geographic coordinates, the words or characters have to be embedded in numerical representations and any bias in the data has to be correctly sampled.

4.1 Geographic coordinates

Even though each kloeke code in the WLD represents an exact geographic location, this abstract representation does not encode proximity between two kloeke codes in a numerically comprehensive way. In any NLP application where the geography is used either as part of the input or the output, it would be beneficial to encode the geographic location in a numerical manner. A dataset (*hg_kloeke*) containing kloeke codes and their geographic coordinates (Web Mercator projection) is available online (Spaan, 2015). The coordinates in this dataset do not sufficiently match the exact centers of towns and hamlets, which automatically induces a numerical error which we want to minimize. Additionally, some kloeke codes in the WLD are outdated and have been split in revised versions of the kloeke table. Each kloeke code is therefore converted to an exact geographic coordinate with a degree precision of 10^{-7}, corresponding to a precision in approximately 10^{-2} m latitudinally and longitudinally using the following workflow:



Each unique kloeke code in the WLD is looked up in the *hg_kloeke* dataset. If not present, we look for any kloeke codes that encompass that code (e.g. "L319" is not present, but "L319p" is). If no such code is found, it is assumed that the kloeke code is a typo and the entry is removed from the WLD. Once geographic coordinates for the kloeke code are looked up, a small algorithm is run in Python that attempts to improve the coordinate's precision: the name of the locality is found through the Meertens Instituut's kloeke search tool (Instituut), which is then used to find the coordinates of the exact geographic center of that locality using Nomatim (Nomatim), a search engine for OpenStreetMap. OpenStreetMap is generally highly accurate and if their respective coordinates do not diverge too much from the *hg_koeke* dataset's, these improved coordinates then replace the originals. Using the *geopandas* package together with GIS data from Europe's NUTS regions (Eurostat, 2023) in Python, it is then verified that the kloeke code's coordinates are situated within 20 km of the borders of Belgian Limburg, Dutch Limburg and Liège. This is done as digitization errors occasionally corrupt a kloeke code into a valid kloeke code, yet outside the region that the WLD covers. In total this

process removes 1 657 entries out of the total of 1 732 195 and visibly allows the kloeke codes to respond to their respective locality centers with very high accuracy. The list of rejected kloeke codes during this process can be found in Appendix B. We have visualized all remaining kloeke codes in the WLD in Figure 3:

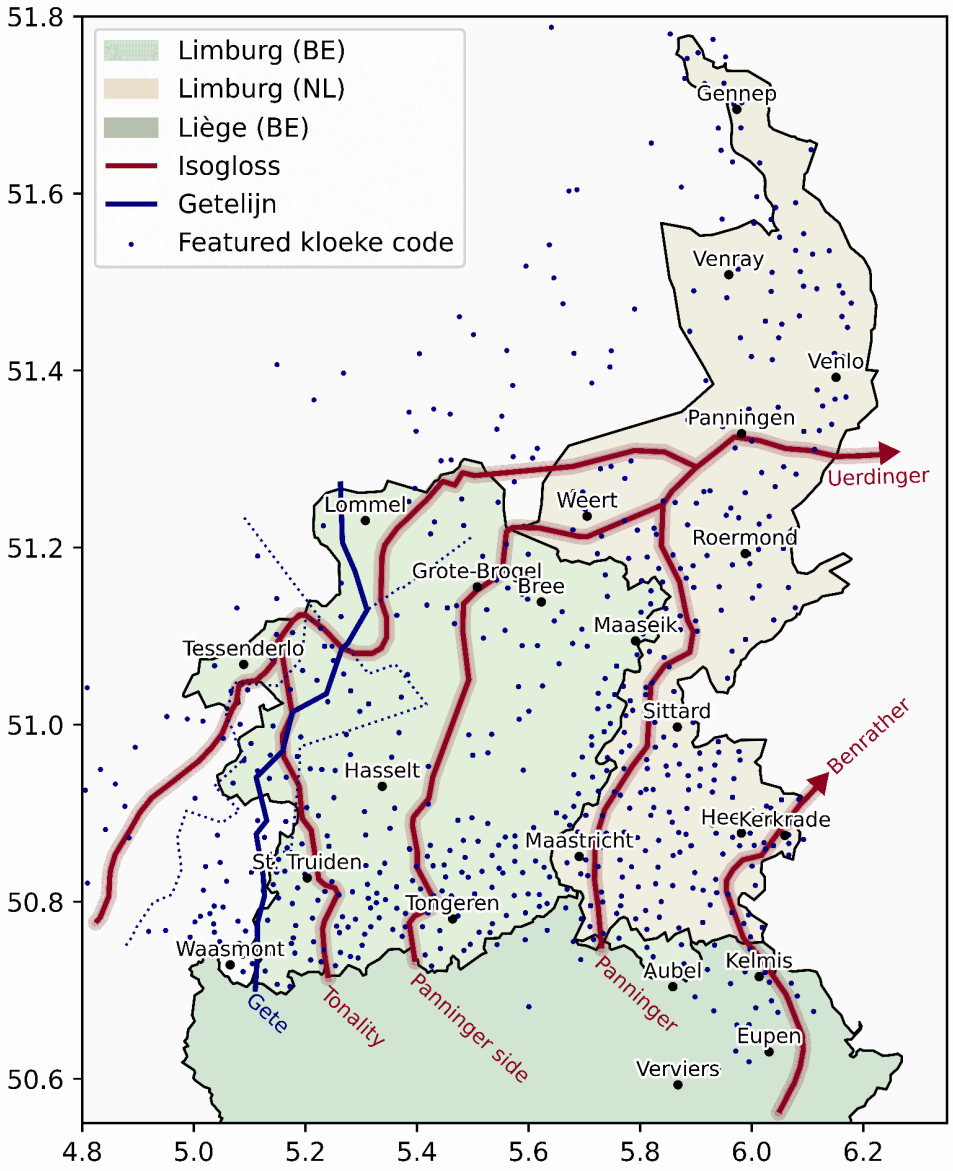


Figure 3: All localities of the WLD within a 20 km radius of both Limburgs and Liège.

Finally, the geographic coordinates (lat, lon) are normalized to (y, x) in the interval $[0, 1] \times [0, 1]$ using a simple rescaling:

$$y = \frac{\text{lat} - \text{min\_latitude}}{\text{max\_latitude} - \text{min\_latitude}}$$
$$x = \frac{\text{lon} - \text{min\_longitude}}{\text{max\_longitude} - \text{min\_longitude}} \tag{1}$$

where min_longitude = 4.7040582, max_longitude = 6.1785077, min_latitude = 50.5932400, max_latitude = 51.9320994.

4.2 Cleaning the keywords and dialect entries

Some entries of the WLD are visibly noisy with meaningless characters (e.g. `ǫpdr¬ ∑ ɪən` or `nę?+f18597ə`), likely due to digitization errors or faulty conversion of unicode characters. The WLD relies on the use of a large number of diacritics to nuance the various Limburgish dialects, meaning that traditional normalization approaches such as converting all characters to lowercase and stripping all non-alphabetic characters are impossible. We therefore implement a custom set of rules to clean the dialect entries:

1. All characters are converted to lowercase.

2. All featured characters are counted. Characters used fewer than 100 times throughout the WLD are assumed to be noise due to their very low frequency. On visual inspection, the words featuring these characters (e.g. `ç, f, §, », π`) are distorted beyond comprehension and therefore are discarded entirely from the WLD. This leaves a total of 118 characters:

	!	"	#	&	(	)	*	,	-	.	/	0	1	2	3	4	5	6	:
;	?	[	]	◌̂	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n
o	p	q	r	s	t	u	v	w	x	y	z	}	~	à	á	â	ä	è	é
ê	ë	ì	í	î	ï	ò	ó	ô	õ	ö	ø	ù	ú	û	ü	ā	ē	ę	ī
ŋ	ō	ő	š	ū	ű	ž	ə	ǫ	Ǫ	ȳ	ə	ɛ	ʔ	◌˂	◌˃	◌˙	◌˱	◌˲	◌̃
◌̄	◌̆	◌̇	◌̋	◌̌	◌̡	◌̢	◌̤	◌̥	◌̰	◌̴	◌̃	◌̩	ẽ	ʼ	·	/	∂		

   Note that the position of the diacritics is exemplified with a ◌: if a diacritic is placed before it is meant to be on the character, if it is placed after it is in the correct position. The first character denotes a space (sentences are part of the WLD dialect entries). Some characters are still superfluous: e.g. `?` or `!` used in expressions, brackets such as `(,),{,}` to give additional info such as gender or plural forms, etc. Additionally, some predictable transcription errors can be reversed such as `"` (which should be `ø`) and additional cleaning can be used, such as converting occurrences of `g` to the proper IPA symbol. The entire sequence of these manual rules can be found in Appendix D. During this step, 8 982 out of 1 730 538 entries are removed.

3. The dialect entries contain sentences. However, most NLP applications require a fixed input dimension. To solve this, we can split dialect entries that are sentences by matching the individual words with their respective Dutch keywords. This is done through a simple algorithm: for each word in the keyword sentence, the Levenshtein ratio [2] w.r.t. each word in the dialect sentence is calculated. The dialect word with the largest Levenshtein ratio (as long as this is above a cutoff value of 0.4) is then taken to be the matched dialect word. The original entry is then removed from the WLD and replaced by these individually matched words. This algorithm comes from the assumption that the keywords resemble the Limburgish word to a reasonable extent (see section 3.2). This sentence splitting increases the WLD by an additional 71 639 successfully split entries, leading to a total of 1 793 195 entries where the keyword and dialect word are matched exactly. Together with 109 333 entries that weren't successfully split but are still useful, the total size of the WLD is now 1 902 528 entries.

[2] See (2). in section 6.1.1

Finally, it is also beneficial to clean the Dutch keywords in order to remove any nonsensical entries. As a result of the splitting in the step above, unnecessary parentheses or other interpunctions are now noise within the Dutch keywords. We clean them using some simple custom rules:

1. If a keyword contains a `(`, `)`, `{`, `}` or `,` that character is removed.
2. If a keyword ends with `-`, `?`, `!` or `:` that character is removed.
3. If a keyword contains `.`, `` `s ``, `-`, ` `, `` `t ``, `[`, `]`, `/`, `<`, `>` that keyword is assumed to be superfluous information (e.g. additional semantic information) and the entry is removed from the WLD.

These steps remove 14 070/1 902 528 entries, most of which are noisy data as a result of the splitting. After these custom data cleaning rules, the total WLD size has now become 1 888 458 entries.

4.3 Geographic representativeness of the WLD

As we can see on the map above, the kloeke codes are more densely distributed in some areas than others. As the main goal of this project is to train NLP applications on the geographic distribution of Limburgish, a geographically unrepresentative dataset would skew any learning towards localities that are more frequently featured in the WLD. We can plot a 2D histogram of all the entries in the WLD (Figure 4.3):

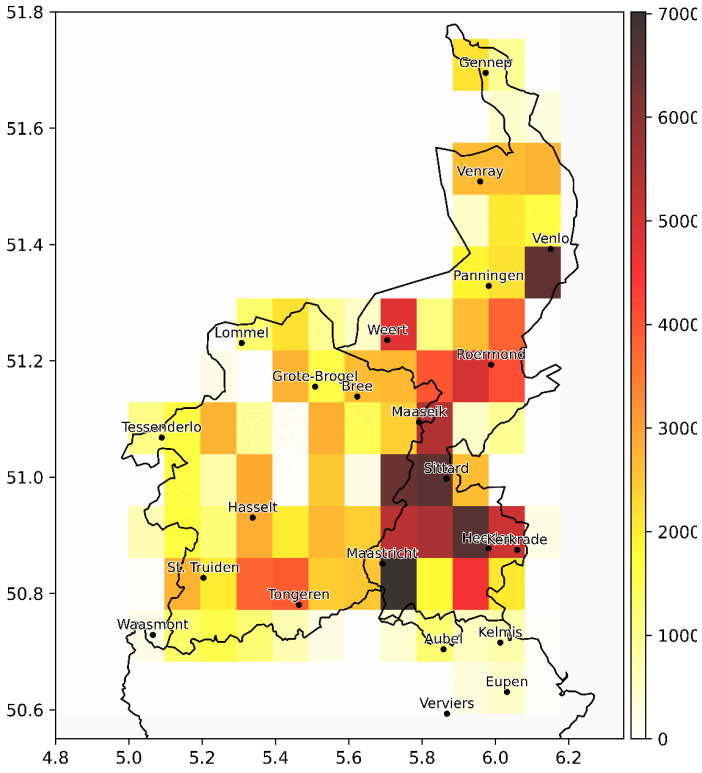


Figure 4: 2D histogram of the frequency of data entries in the WLD.

We can see in the histogram that the WLD is geographically skewed: Dutch Limburg has nearly double the number of entries as Belgian Limburg and specifically the south is well represented around major urban centers such as Maastricht and Sittard. The north of Dutch Limburg is sparser, although the region around Venlo is clearly represented. In Belgian Limburg, the south is slightly more represented than the north, which corresponds to the population density at the time of the data collection. The region northeast of Hasselt is poorly represented as this area (the Donderslag moors) is barely populated, as is the empty spot in the east (High Campine) and northwest (Bosland). Finally, the northeast of Liège is poorly represented, likely due to a lower population density or possibly a lower interest to collect data in that region.

To remove this geographic bias from the WLD, we will undersample the data. Each kloeke code will receive a weight proportional to its frequency in the WLD. Because resampling the WLD per kloeke code would be too granular, we will approximate the rescaled geographic distribution using a 2D Gaussian kernel. Using a bandwidth factor of 0.2, we approximate the distribution (Figure 5). The weight per kloeke code is then taken from its corresponding Gaussian kernel approximation, squared. These weights then represent the probability of each entry in the WLD being sampled. Reducing the total WLD size to 70% or 1 321 921 entries yields the following geographic distribution (Figure 6):

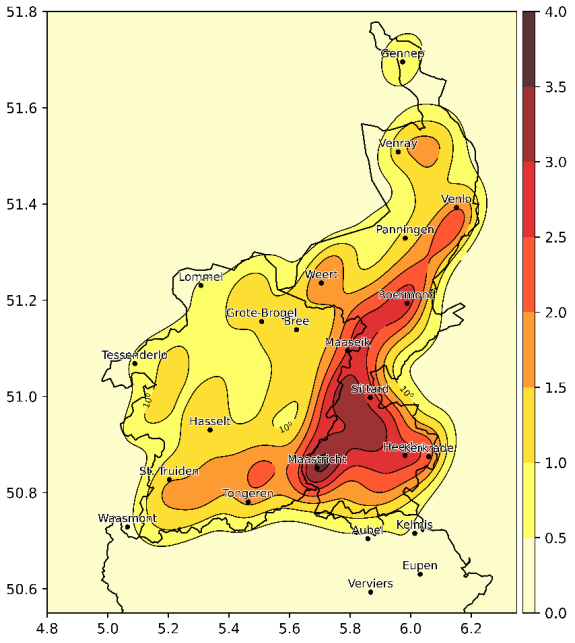


Figure 5: Rescaled Gaussian kernel approximation of the WLD distribution.

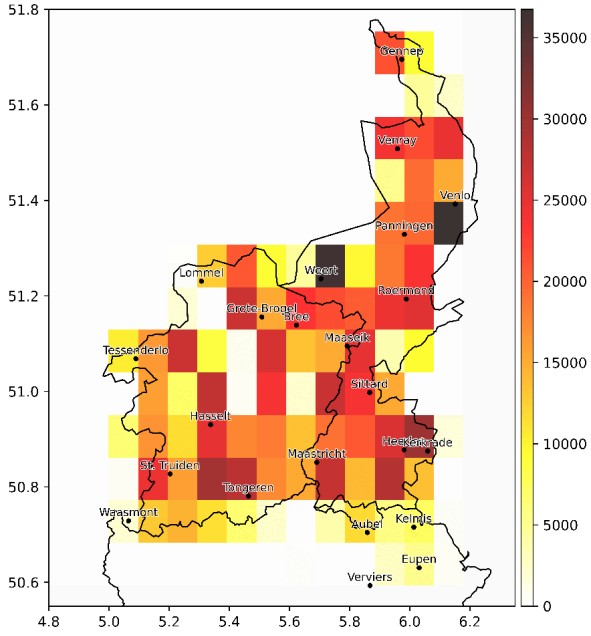


Figure 6: Data entries of the WLD after undersampling.

We can conclude that this undersampling makes the WLD geographically representative, as the data now seems uniformly spread over the covered area. The sparsely populated regions as described above remain poorly represented, as very little data was available before the undersampling.

4.4 Spelling normalization

As is evident from Chapter 2, the spelling conventions in the WLD can vary significantly. To normalize the spelling, we will select a normalized subset of the WLD using an exclusion principle: whenever a dialect word contains bigrams using typical Dutch orthography or the Veldeke spelling, the word is assumed not to be normalized in the morpho-phonological/phonetic spelling (that should be the standard of the WLD).

We note that this preprocessing step is different from the normalization model that we construct in Chapter 5. We will thus narrow the undersampled WLD to a smaller, normalized dataset. Additionally, we will only consider entries that were successfully matched to their keywords to ensure that the selected dataset is meaningful. The excluded bigrams are manually chosen from a combination of Dutch orthography and the Veldeke spelling (see Appendix A):

aa	ee	uu	oo	ij	ei	eu	oe	au	ou	ui	y	sch	sj	ng	nk	uw	ae	ao	äö
ië	eë	i-j	dj	qu	tj	zj	oë	uë	ch	rr	tt	pp	ss	dd	ff	gg	kk	ll	mm
cc	bb	nn	oa	ks	ie	ai	ww	ea	ei										

We find that the normalized WLD now contains 711 637 entries. This is a strict exclusion of half the WLD, but due to the lack of a curated dataset we need to maximize the quality of the dataset that we will use. If we once again plot the geographic distribution in Figure 7 we find that the normalized, undersampled data remain geographically representative:

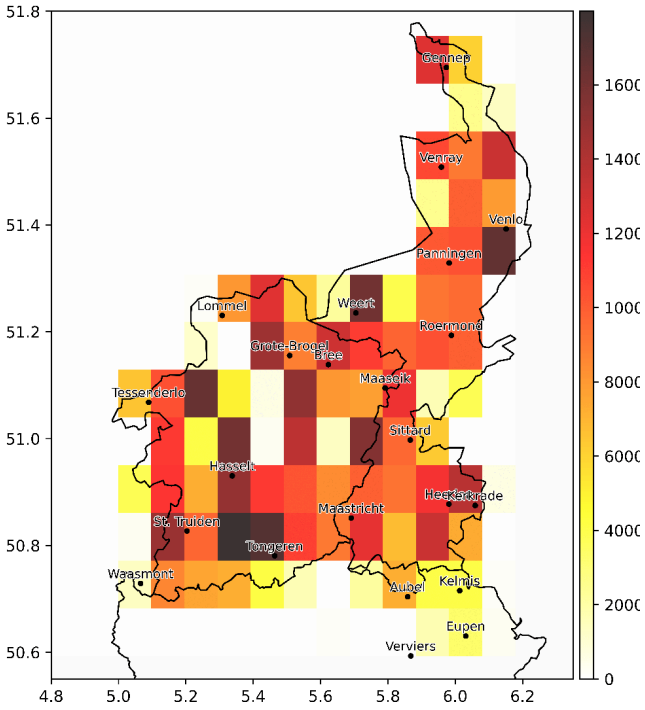


Figure 7: Geographic distribution of the normalized, undersampled WLD.

5 Extremely granular Limburgish dialect identification: a deep learning approach

Language Identification or *LID* is the task of identifying a natural language input as belonging to a specific language. Each of the over 7000 living languages (Ethnologue) has varying distinct properties such as semantics, phonology, stress patterns, tonality, etc. which can be used by humans to distinguish between them (Harper and Maxwell, 2008). LID tasks typically do not focus on classifying acoustic natural language input but textual input instead, which is also what we will consider. LID is crucial in many NLP pipelines that require the selection of specific language models (Apple Machine Learning Research: Natural Language Processing Team, 2023), machine translation, data mining or spam filtering (Ismail, 2020). In the case of distinguishing distant languages, LID is considered a solved problem (Goswami et al., 2020) as various techniques in machine learning have yielded very high results in the past decades, particularly Naive Bayes and Support Vector Machine approaches on word n-grams, although new deep learning techniques have shown better performance on fewer annotated data (Goswami et al., 2020). Recently, recurrent neural network-based architectures such as Long Short-Term Memory networks (LSTM) have achieved state-of-the-art results (Apple Machine Learning Research: Natural Language Processing Team, 2023).

Two factors increase the complexity and difficulty of LID: (1) the identification of short texts and (2) the identification of similar languages. Correctly identifying the language of a short text is an important task in an era where short language data are prevalent: text messages, tweets (Apple Machine Learning Research: Natural Language Processing Team, 2023), search queries, or code-switching within texts or even sentences. Traditional n-gram or machine learning approaches tend to perform worse on these tasks (Balazevic et al., 2016). Secondly, similar languages, varieties, and dialects are difficult to distinguish due to shared phonology, morphology, syntax, and vocabulary and are the bottleneck of state-of-the-art LID applications (Zampieri et al., 2014). This second problem is often defined as *Dialect Identification* or *DID*. DID has sparked increasing interest in the past years as studies have been conducted on areas with different language variety scenarios [3]: areas with standardized written languages, while the spoken languages vary greatly geographically, but are also present in mostly non-standardized written forms on social media, such as Arabic (Shoufan and Alameri, 2015) and Swiss German (Scherrer and Rambow, 2010) or areas with a standardized written and spoken language such as English (Simaki et al., 2017), (Lui and Cook, 2013). Other examples include historical, mainly non-standardized languages such as cuneiform texts (Jauhiainen et al., 2019). Most of these aforementioned approaches still utilize traditional machine learning approaches such as Naive Bayes classifiers using n-grams, but the state-of-the-art results come from deep learning approaches such as autoencoders (Parida et al., 2020), Convolutional Neural Networks (Goswami et al., 2020) and particularly bidirectional LSTMs (Elaraby and Abdul-Mageed, 2018) and transformer-based models (Tonja et al., 2022).

Limburgish represents an interesting case study of DID for very similar languages: the Limburgish dialects vary geographically along well-known isoglosses (see Chapter 1) yet remain mutually intelligible. Most of the variation occurs phonologically and to a lesser degree lexically (Bakkes et al., 2007), (Belemans and Keulen, 2004), (Hermans, 2013). The phonology of Limburgish dialects is often popularly claimed to be so predictable that speakers can easily identify the city or village of origin of another speaker (Dols, 1946). This is supported by Limburgish allowing for a very large degree of phonological variation due to the large number of phonemes: the Maastrichter dialect has 21 monophthongs and 3 diphthongs (Gussenhoven and Aarts, 1999) while the Weerter dialect has 12 long and 10 short monophthongs, and 6 diphthongs in a phonetic five-height system (Heijmans and Gussenhoven, 1998), which corresponds to one of the largest vowel systems in the world (Ladefoged and Disner, 2012). In comparison, Belgium's standard Dutch and the Netherlands' standard Dutch have twelve vowels (Hitch, 2020), Californian English 15 (Ladefoged and Disner, 2012), Cantonese 13 (Luo et al., 2020) and Spanish 5 (Hualde, 2013). Additionally, Limburgish is one of few European languages with lexical tonality (Fournier, 2008); two tones are used as phonemes.

Due to this large phonological variation along extensively studied isoglosses in a relatively compact region of

[3] For an overview see (Jauhiainen et al., 2018).

approximately 70 by 100 km, Limburgish is an ideal case study for extremely granular dialect identification tasks. Additionally, due to the existence of the WLD, a large dataset with exact geographic coordinates for each dialect word is available so that we need not use a classification task for DID, but we can consider a regression task where we ought to find the exact coordinates of a specific text input. This has the advantage of avoiding arbitrary demarcations of dialects and languages, as is typically done in DID and LID tasks. Furthermore, since the WLD consists mainly of single-word entries and a large array of diacritics is used to represent their exact phonology (see Appendix C), it allows for the study of short-text DID to be pushed to extremity by only considering single-word inputs. Considering popular claims about the ease of Limburgish dialect identification, the high phonological variation and their predictability, we expect a regression DID task on Limburgish to be a perfect scenario for testing solutions to the two bottleneck problems in contemporary DID as outlined above.

In this section we will study single-word dialect identification of Limburgish, where the regression task consists of predicting the exact geographic coordinates. We will consider various state-of-the-art architectures, types of embedding, the impact of normalization, etc.

5.1 DID using feedforward neural networks

Feedforward neural networks are the most basic architecture in deep learning, they are universal approximators of continuous functions (as is approximately the case for Limburgish dialects) but not optimal for sequential data such as language input. For a recent review see (Shrestha and Mahmood, 2019). We will consider the preprocessed, resampled, and spelling normalized WLD for this and all deep learning architectures below. To verify our results, we will consider two baselines:

- (1): predicting the geographic center of $(0.35075642, 0.64351419)$ in normalized coordinates, which corresponds to $\approx 51.0629N, 5.6529E$ in Web Mercator projection (1) and lies within the locality Opoeteren. These coordinates are exactly the center of all entries in the (resampled) WLD. Since the WLD is resampled to become geographically representative, the exact geographic center would be the constant solution with the lowest error.
- (2): we will use the results of the most optimal feedforward neural network configuration from this section as a baseline for other deep learning architectures, as we expect those to perform better, as is the case in the literature.

The preprocessed version of the WLD contains 711 637 entries which we will randomly split into an 80% training set, 10% validation set and 10% test set. These datasets are sampled again for each comparison in this section. As most deep learning architectures require inputs of a fixed size, we select an input size and pad entries that are too short. To visualize the lengths of the entries in the normalized WLD we plot a histogram (Figure 8):

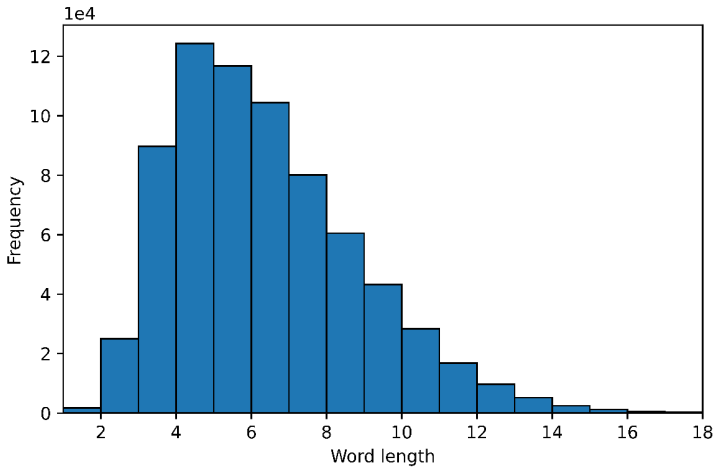


Figure 8: Lengths of words in the WLD after preprocessing, resampling and spelling normalization.

Since 674 414 out of 710 957 entries are fewer than or equal to 10 characters, we only consider these entries for the DID task. As feedforward neural networks only process numerical inputs, we have to choose an embedding for the characters in the words. We consider two types:

1. One-hot encoding: converts each character of the input word into a null vector with the dimension size set to the total number of characters. Only the component corresponding to that character is set to 1. This does not take into account any information regarding the characters.

2. Embed IPA segments as phonological features: embed each (IPA) character into a vector where each component corresponds to a phonological feature (e.g. labial or not, voiced or not, etc.). This takes phonetic information into account of the characters and has been shown to outperform one-hot encodings in some tasks (Mortensen et al., 2016). For this, we use the Python library *PanPhon*, which uses 24 articulatory features.

For each of the two types of embedding we perform a hyperparameter search: we vary the number of hidden layers from $1-3$ and the number of hidden nodes per layer from $10-1000$. Each hidden layer except for the final layer uses a relu activation function. We use Mean Squared Error (MSE) as the loss function and Adam (a stochastic gradient descent algorithm) as the training method. The batch size is 320 and convergence is assumed when the validation MSE has not improved by 10^{-4} over 3 epochs. The implementation is done in Python using the *Keras* librar and the hyperparameter search using the *Optuna* library (Akiba et al., 2019). The hyperparameter search is used to determine the most optimal network architecture. This is done through Optuna's state-of-the-art *Tree Parzen Estimator*, which constructs probabilistic parameter estimators to efficiently search the parameter space after each of the 200 sampled network configurations, varying the number of hidden layers and nodes per hidden layer, while storing the loss on the validation test set upon convergence. We first attempt the hyperparameter search on the one-hot encoding. Since we are left with a total of 92 possible characters, each character is embedded into a 92-dimensional vector. Inputs that are fewer than 10 characters are padded with null vectors. The final vector is flattened to a total input vector of dimension 920.

If we plot the MSEs of the validation set (Figure 9), we find that there is a slight improvement for using more than one hidden layer, which is a generally unexpected result for a feedforward neural network where usually a single hidden layer is sufficient if it is large enough (Heaton, 2008). Having three hidden layers does not seem to improve the errors and we therefore decide on an architecture with two hidden layers. Varying the number of neurons in both hidden layers and plotting the results on the MSE, we find Figure 9:

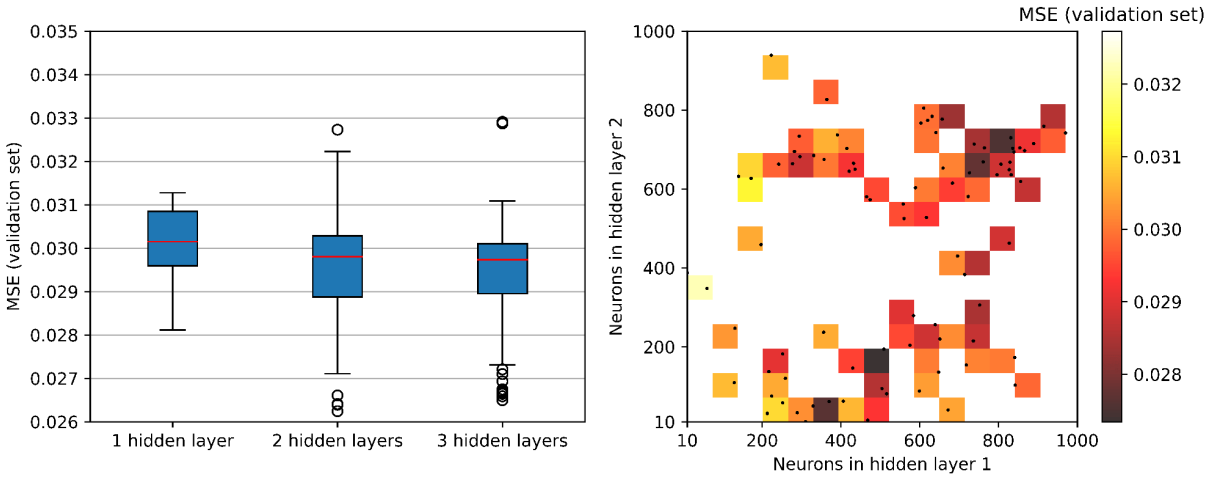


Figure 9: Boxplots of MSE values per number of hidden layers (left) and the MSE values after varying the two hidden layer sizes (right), both determined on the validation set.

The hyperparameter search results and the visualization indicate that the most optimal network architecture consists of two hidden layers with approximately 830 and 670 neurons in the hidden layers respectively (a total of 1.3M parameters). This corresponds to an MSE of 0.0266 on the test set. We note that the size of the first hidden layer approximates the input size of 920. The relatively large size of the network and the improvement due to two hidden layers indicate that features that are relevant for DID require feature extraction that is not linearly separable and thus more complex. To compare the influence of the type of embedding, we repeat this hyperparameter search using PanPhon's phonological embedding. Since each IPA character is converted to a 24-dimensional vector, the padded and flattened input vector is of dimension 240. Visualizing the results as above we find in Figure 10:

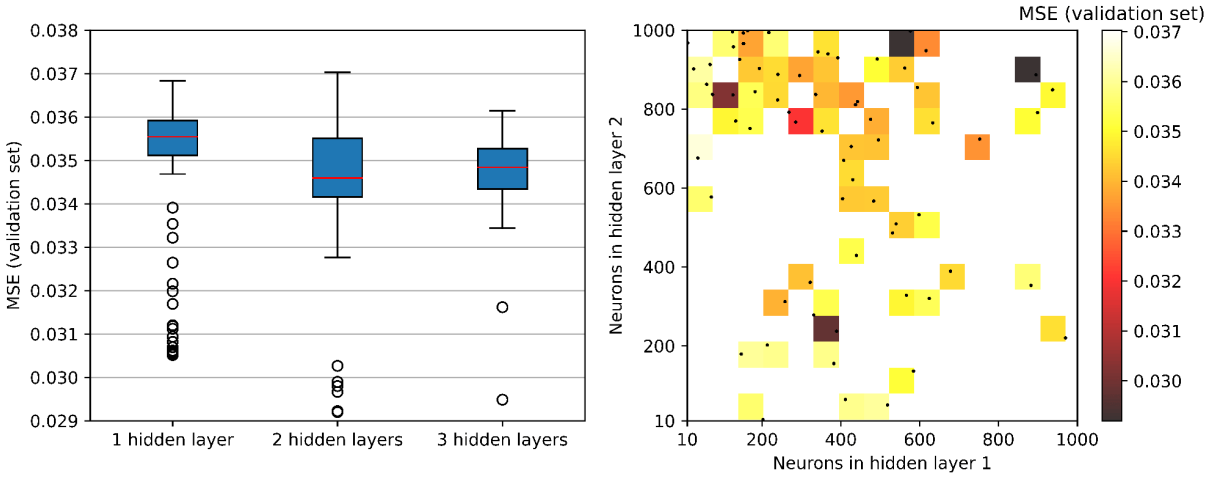


Figure 10: Boxplots of MSE values per number of hidden layers (left) and the MSE values after varying the two hidden layer sizes (right), both determined on the validation set.

We again find that an architecture with two hidden layers improves upon having only a single hidden layer, with no improvement in the case of three hidden layers. The hyperparameter search yields an optimal architecture situated around approximately 900 neurons in both hidden layers (a total of 1.1M parameters). This corresponds to an MSE of 0.0292 on the test set. We notice that although the network is slightly larger in hidden layer size, due to a smaller input size (240 instead of 920), both optimized networks are similar

in size. To verify whether the one-hot encoding yields better results than the phonological embedding, we compute the physical distance (in km) between the predicted coordinates and true coordinates for both types of embedding and visualize them in boxplots (Figure 11):

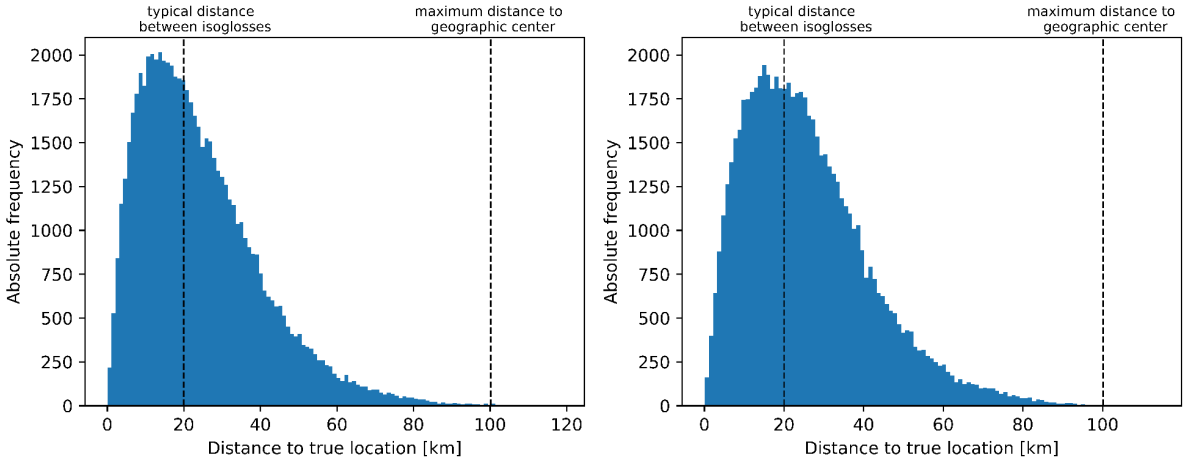


Figure 11: Distances between the predicted location and true location for the one-hot encoding (left) and phonological embedding (right).

Here we have indicated the maximum distance possible between any entry in the WLD and the geographic center near Opoeteren (100.17 km), as well as a characteristic distance between isoglosses (20 km). This characteristic distance is determined by averaging the approximate maximum distance between the isoglosses as in (Bakkes et al., 2007). The mean distance of the one-hot encoded embedded network is 24.38 km, for the phonological embedded network this is 25.97 km. Since these distributions are not normally distributed and dependent, we execute a two-sided Wilcoxon sign-rank test using *SciPy*:

- H_0: both networks perform equally well for the physical distance metric.
- H_1: there is a difference in performance between both networks for the physical distance metric.

We find a p-value of $p < 0.001$ and can therefore reject the null hypothesis. Since the networks do not perform equally, we test whether one network outperforms the other through a one-sided test:

- H_0: the phonologically embedded network performs better than the one-hot encoded network for the physical distance metric.
- H_1: the one-hot encoded network performs better.

We find a p-value of < 0.001 and can therefore reject the null hypothesis, we conclude that the one-hot encoded network performs better. It is likely that, although PanPhon tries to provide a more natural metric for sounds in the International Phonetic Alphabet, this metric does not suffice for Limburgish. As we have discussed in the introductory section of this chapter, Limburgish dialects are very rich in phonology and projecting this phonology onto PanPhon's 24-dimensional vectors might result in a loss of information. For example, the Weerter dialect has 5 vowel heights, while PanPhon only encodes vowels binarily (high/low, back/front).

Repeating the first hypothesis test for the first baseline (the geographic center) and the one-hot encoded network we find a p-value of < 0.001, which means that we can reject the hypothesis that both networks perform better. If we then test whether the first baseline performs better than the one-hot encoded network, we again find a p-value of < 0.001, which means that our one-hot encoded network outperforms the baseline. This is expected since the mean distance for the first baseline is 31.24 km. We plot the distribution of physical distances in Figure 12

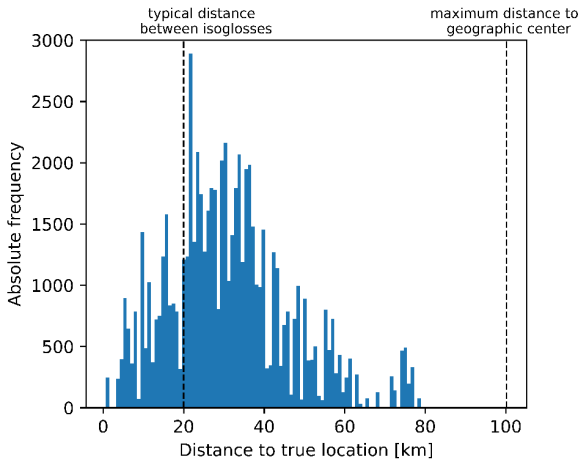


Figure 12: Distances between the predicted location and true location for the first baseline.

After testing the different types of embedding, we now remove a step from the preprocessing part and examine the impact of a spelling normalization scheme. We repeat the training of the optimized one-hot encoded network architecture from above separately on the normalized part of the WLD, which contains 711 637 entries, as well as the WLD before this normalization step, which contains 1 321 921 entries. We split the WLD before the normalization step into an $80-20$ train and test set and first train the optimal network on the entire training set and then train it on the normalized part of the training set (using the same procedure as in the previous section). Both networks are then tested on the test set (which is normalized), plotting the physical distances we find (Figure 13):

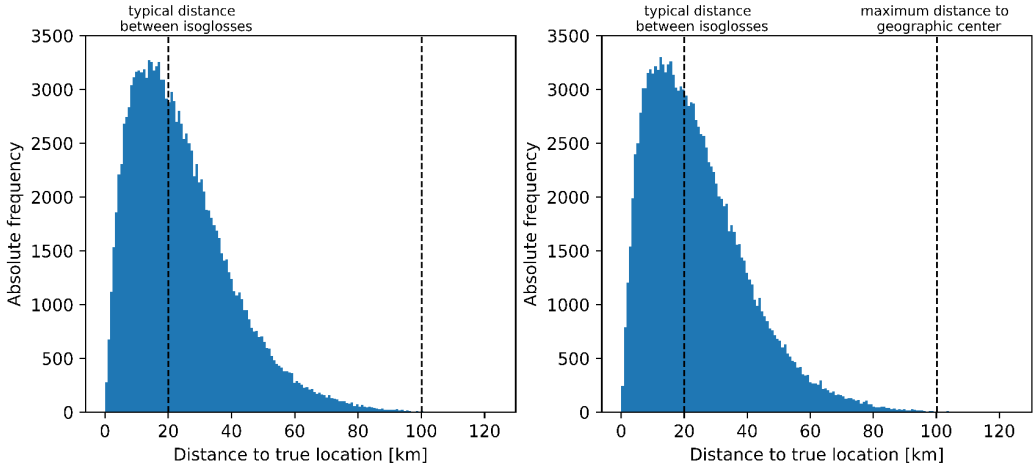


Figure 13: Distances between the predicted location and true location for the network trained on a WLD that is unnormalized (left) and is normalized (right).

The mean physical distance for the network trained on the WLD before normalization is 24.45 km and for the network trained after normalization is 24.39 km. We again perform the same hypothesis test as above and find a p-value of < 0.001 from which we can conclude that there is indeed a difference in performance between training on a normalized version of the WLD and training on the entire WLD before normalization. To test whether the unnormalized trained network performs better than the normalized trained network, we conduct another one-sided hypothesis test and find a p-value of < 0.001 from which we can conclude that the network trained on normalized data outperforms the network trained on the entire WLD.

It is likely that, since the test set only contains normalized entries, the network trained on a normalized dataset performs better on the test set even though it is approximately half the size of the dataset on which the other network is trained. The choice of such a normalized test set is important for our goal of DID: we wish to identify the location of Limburgish words in high-quality phonetic spelling, rather than conventional spelling. A dataset of normalized IPA spelling allows for a more consistent study of Limburgish dialects, as well as not risking the loss of generality of this study for other languages that do not necessarily have a tradition of conventional orthographies.

In conclusion, we find that a one-hot encoding outperforms a phonological embedding of the phonetic characters, likely because the phonological embedding does not sufficiently support Limburgish phonology. A normalized dataset, even though it has fewer entries than an unnormalized dataset, performs better on a normalized test set. The most optimal feedforward neural network architecture for DID in Limburgish consists of an input layer of dimension 920, reflecting the 92 characters in the one-hot encoding and a padded word length of 10, two hidden layers of respectively 830 and 670 neurons, connected through relu activation functions and 2 nodes in the final layer, using a sigmoid activation function. In total, 1.3M parameters are used and the mean accuracy is 24.38 km, while a characteristic distance between 2 isoglosses is approximately 20 km and the maximal distance from any entry in the WLD to the geographic center is 100.17 km. This network is visualized using Keras in Figure 14.

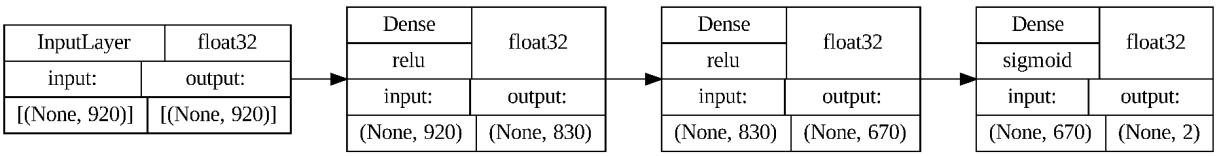


Figure 14: Diagram of the most optimal feedforward neural network for Limburgish DID. *None* stands for the batch size and can be adjusted according to available computing resources.

The physical distances of this network will be used as a second baseline for the next deep learning architectures, together with the first baseline (predicting geographic center). The train-validation-test datasets will be sampled again and frozen as of now, as we have established both baselines. Baseline 1 and baseline 2 yield respectively 31.17 and 24.19 km in mean physical distance on the test set.

5.2 DID using Long Short-Term Memory networks

Many state-of-the-art applications in NLP use a special type of Recurrent Neural Network (RNN): the *Long Short-Term Memory network* or *LSTM*, first described in (Hochreiter and Schmidhuber, 1997). RNNs are preferred for sequential data such as characters in words, but they do not store information over longer time/step intervals. LSTMs solve this by maintaining a "short term memory", whose flow of information is controlled by the individual cells, corresponding to the input dimensions. LSTMs only encode sequential data in one direction and it is therefore preferable to consider an additional LSTM, encoding the other direction. Both outputs are then appended into what is called a bidirectional LSTM layer or *biLSTM*. This currently outperforms other deep learning and machine learning architectures in the case of Arabic DID (Elaraby and Abdul-Mageed, 2018) and for general very short multilingual language identification (Toftrup et al., 2021).

Here we will study biLSTMs for DID in a similar setup as above: using Optuna, we determine the optimal architecture for a biLSTM-based architecture. We vary the number of biLSTM layers from $1-5$, flatten the output and then vary the number of feedforward layers from $1-2$ before the final 2 neurons in the output layer. Between each layer we again use a relu activation function, with the final layer using a sigmoid function. We run Optuna's hyperparameter search 150 times, where each architecture is trained until convergence using the Adam training method in batches of 160 (i.e. >3 epochs where the validation MSE has not increased by 10^{-4}). The input dimension remains 920 as we use the one-hot encoding and we vary the number of cells

in each biLSTM layer from $1 - 500$ and the number of neurons in each feedforward layer from $10 - 1000$. If we plot the MSEs of the validation set per number of stacked LSTM layers in Figure 12 16, we find:

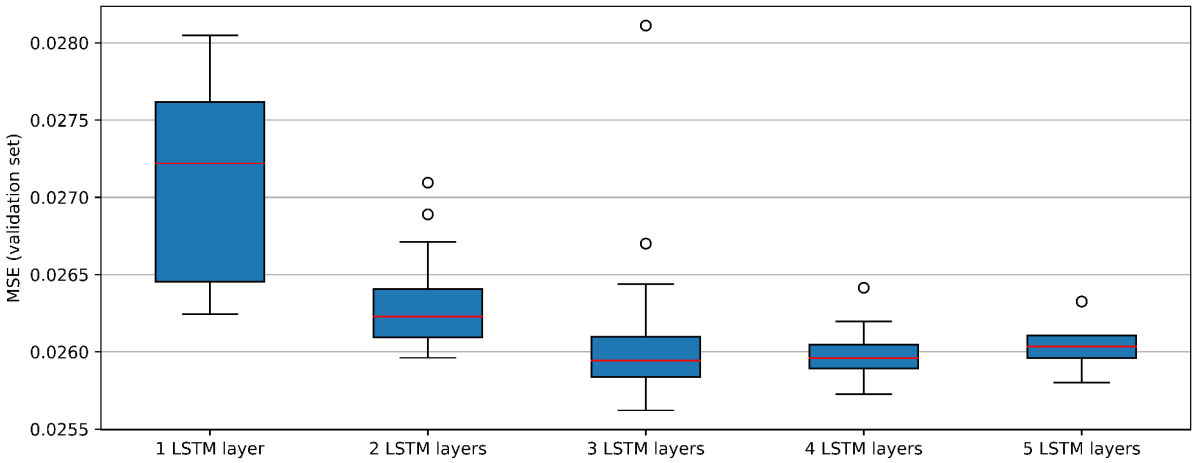


Figure 15: Boxplots of MSE values per number of stacked LSTM layers, determined on the validation set.

We notice that stacking multiple LSTM layers visibly improves the mean MSE on the validation set, although there do not seem to be noticeable improvements after three stacked LSTM layers. This is an expected result, as stacked RNN architectures are known to outperform single-layer RNN architectures for machine translation tasks (Goldberg, 2015). The hyperparameter search determines that the most optimal architecture consists of 3 stacked LSTM layers of approximately 300 units each, a flattening layer, and one feedforward layer of approximately 500 hidden neurons. This corresponds to an MSE of 0.0262 on the test set. To verify whether this architecture yields better results than the two baselines, we compute and plot the physical distances between the predicted coordinates and real coordinates in Figure 16:

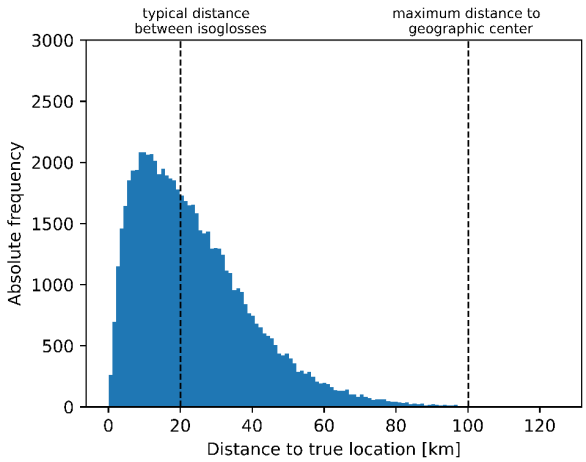


Figure 16: Distances between the predicted location and true location for the optimized LSTM architecture.

The mean physical distance on the test set is 23.88 km, which is an improvement compared to both baselines. To determine statistical significance, we again conduct two hypotheses tests to verify whether the physical distance errors of the LSTM architecture are equally distributed as the two baselines. We find a p-value of < 0.001 when compared against baseline 1 and a p-value of < 0.001 and we can therefore reject both null hypotheses. If we then test whether both baselines yield smaller physical distance errors, we find a

p-value of < 0.001 for baseline 1 and a p-value of < 0.001 for baseline 2 from which we can conclude that the null hypotheses are rejected and the optimized LSTM-based architecture indeed performs better than the optimized feedforward neural network architecture and the geographic center baseline. This was already visually apparent in the distribution of physical distance errors: the peak of the curve is farther away from the typical distance between isoglosses than is the case in the feedforward analysis.

In total, this architecture uses 8.3M parameters, which is significantly bigger than the optimized feedforward neural network. The network is visualized in Figure 17.

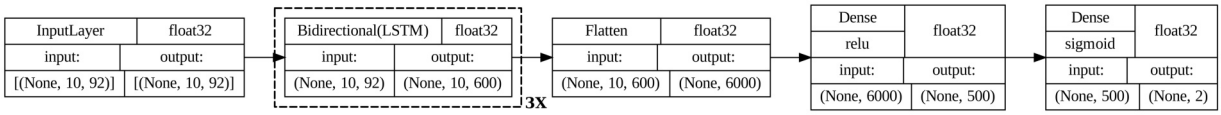


Figure 17: Diagram of the most optimal LSTM-based architecture for Limburgish DID. *None* stands for the batch size and can be adjusted according to available computing resources. The stacked LSTM layer is abbreviated.

5.3 DID using transformers

Most state-of-the-art applications in NLP are currently dominated by transformer architectures (Wolf and et al., 2020). The transformer (Vaswani et al., 2017) architecture scales better with model size and training data than RNN or CNN-based architectures and outperforms LSTMs in capturing long-range features in sequential data through its attention mechanism. Additionally, it allows for parallel training, which significantly reduces the training time for its implementations. Stacked transformers can be found in BERT architectures (Devlin et al., 2019) and GPT language models (Liu and et al., 2023) such as the recent ChatGPT phenomenon. We will consider the same setup as the LSTM hyperparameter search and will replace the biLSTM layers with the encoding block part of the transformer (Vaswani et al., 2017). We vary the stacking of the encoder block from $1-5$, flatten the output and then again vary the number of feedforward layers from $1-2$. For each encoder block, we also vary the latent dimension from $1-1024$ and the number of attention heads from $1-16$. After repeating the hyperparameter search for 100 trials, we obtain the following plot (Figure 18) of the MSEs of the validation set per number of stacked encoder blocks:

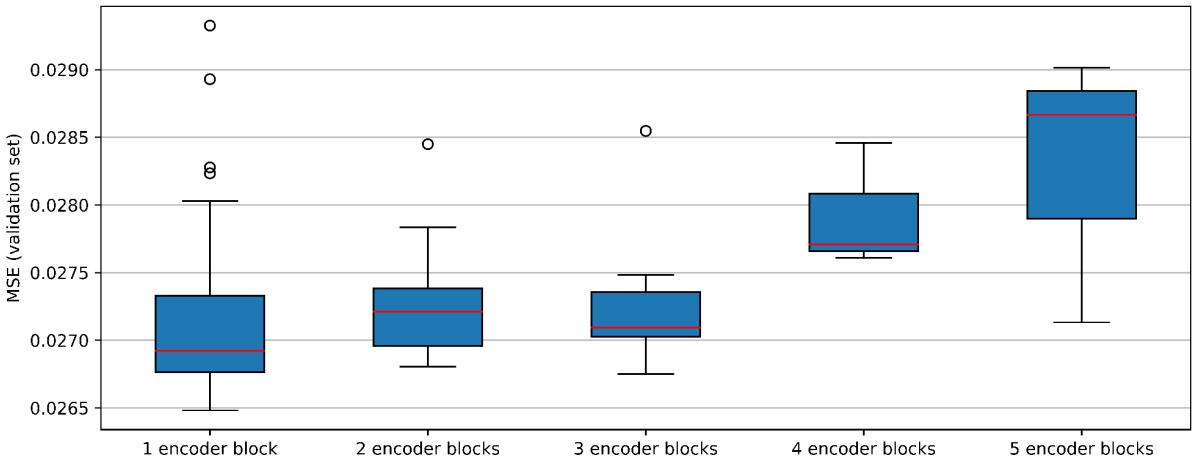


Figure 18: Boxplots of MSE values per number of stacked encoder blocks, determined on the validation set.

We notice that, unlike stacked LSTM layers, there does not seem to be a significant improvement in mean MSE (validation set) as encoder blocks are stacked. We do note that the drastic increase in MSE and variance

after 3 stacked encoder blocks is an artifact of the hyperparameter search, as this part of the parameter space was not as extensively explored by Optuna. The trend in these results is in line with current research: it has been shown that stacking LSTM layers benefits the extraction of higher-level linguistic knowledge, while transformers do not show such typical monotonic behavior (Liu et al., 2019). As our task of short-text DID does not require the extraction of generalized, high-level information such as semantics, we hypothesize that the single-layered attention mechanism of the encoder block is sufficient. It is possible that most information regarding the location of a word is directly encoded in its character representation, to which the attention mechanism has direct pairwise access.

The most optimal architecture as determined by the hyperparameter search consists of one encoder block (8 attention heads and latent dimension of approximately 500) and two feedforward layers of ≈ 1000 and ≈ 600 hidden neurons respectively. This corresponds to an MSE value of 0.0267 on the test set. We again compute the physical distances in Figure 19:

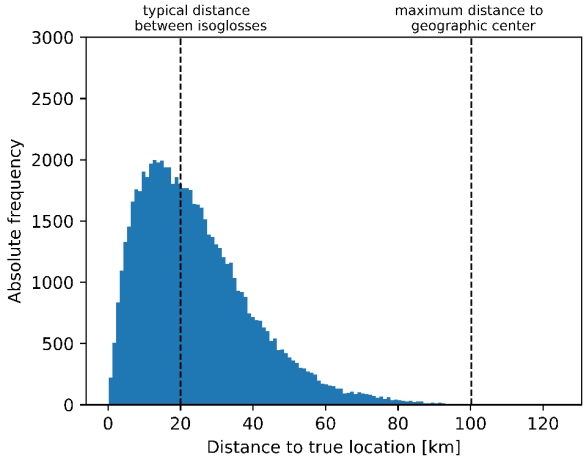


Figure 19: Distances between the predicted location and true location for the optimized transformer encoder-based architecture.

The mean physical distance on the test set is 24.47 km, which is less accurate than the LSTM architecture and baseline 2, although better than baseline 1. We perform the hypothesis tests of equal distribution with both baselines and find a p-value of < 0.001 for baseline 1 and a p-value of < 0.001 for baseline 2. We can reject the hypothesis that the transformer encoder architecture performs equally well as both baselines. We then test whether both baselines perform better than the transformer encoder architecture and find a p-value of < 0.001 for baseline 1 and a p-value of 1 for baseline 2. We can conclude that the transformer encoder-based architecture performs better than the geographic center baseline, but the feedforward-based architecture performs better according to the physical distance metric.

In total, this architecture uses 1.6M parameters, which is significantly less than the LSTM architecture and comparable to the feedforward architecture. The model architecture is visualized in Figure 20.

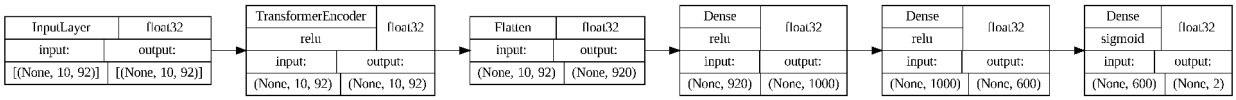


Figure 20: Diagram of the most optimal transformer encoder-based architecture for Limburgish DID. *None* stands for the batch size and can be adjusted according to available computing resources.

5.4 Analysis of results

architecture	feedforward (one-hot)	feedforward (PanPhon)	LSTM	transformer encoder
MSE	0.0266[4]	0.0292 [4]	**0.0262**	0.0267
mean distance [km]	24.19	25.97 [4]	**23.88**	24.47
one-sided p-value baseln. 1	< 0.001	NA	< 0.001	< 0.001
one-sided p-value baseln. 2	NA	NA	< 0.001	1
parameters	1.3M	1.1M	8.3M	1.6M

Table 3: Results of Limburgish DID using different deep learning architectures.

From our results (Table 3) we can conclude that the most accurate architecture for Limburgish short-text dialect identification is LSTM-based. Stacked LSTM layers outperform single-layered LSTM architectures since they likely extract linguistic information more comprehensively. Even though transformers and their attention mechanisms are highly prominent in state-of-the-art applications, they are in this case outperformed by LSTM architectures and no additional increase in performance is derived from stacking transformer encoder blocks.

As Limburgish phonology is complex, it is likely that a phonological embedding is not sufficient to encode the WLD. This could explain why a simple one-hot encoding performs better. A feedforward network trained on a smaller, normalized dataset performs better than a feedforward network trained on an encompassing, larger, unnormalized dataset for the task of DID on a normalized dataset. All considered deep learning architectures perform better than the baseline of simply predicting the geographic center of the data. Only the LSTM architecture performs better than the feedforward network baseline, although it should be noted that the improvement in the mean distance (0.32 km) requires an LSTM architecture with 6× the number of parameters and the feedforward architecture may therefore be a more economical choice in real-life applications.

We will now analyze the behavior and results of the LSTM-based architecture, which yielded the best results. As we have described in the first chapter, the main changes and isoglosses in the Limburgish language occur relatively vertically from east to west. If we decompose the physical distances between the predicted coordinates and true coordinates on the test set into the distances along the latitudinal and the longitudinal axes we find in Figure 21:

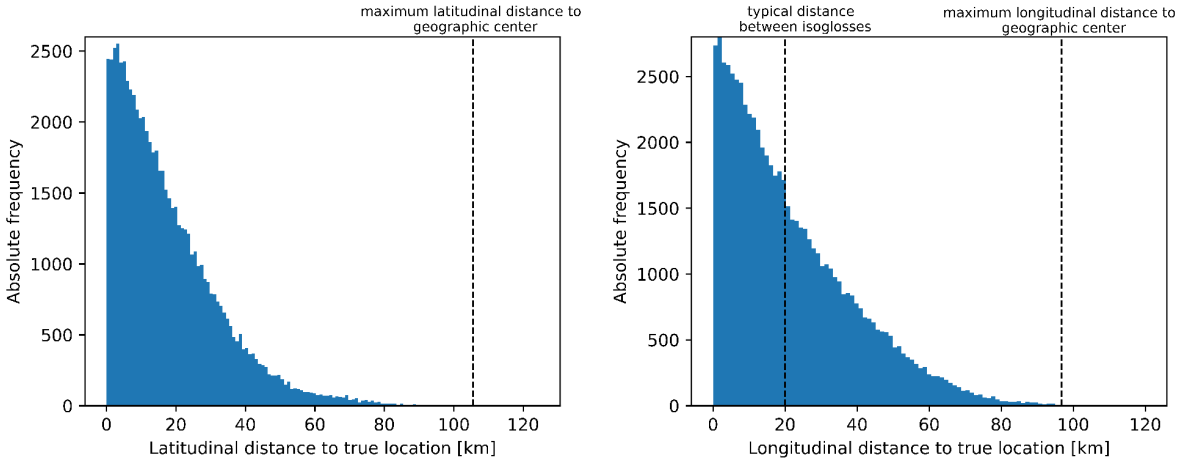


Figure 21: Physical distance errors on the test set, decomposed along the latitudinal and longitudinal axes.

[4] These values are from a differently sampled test set than the other entries.

The mean latitudinal distance is 17.80 km and the mean longitudinal distance is 21.75 km, from which we can infer that on average the variation in the longitudinal direction (i.e. horizontally) is larger than the latitudinal direction, which corresponds to the traditional notion in Limburgish dialectology. We now visualize the physical distance errors in a 2D, normalized histogram (Figure 22).

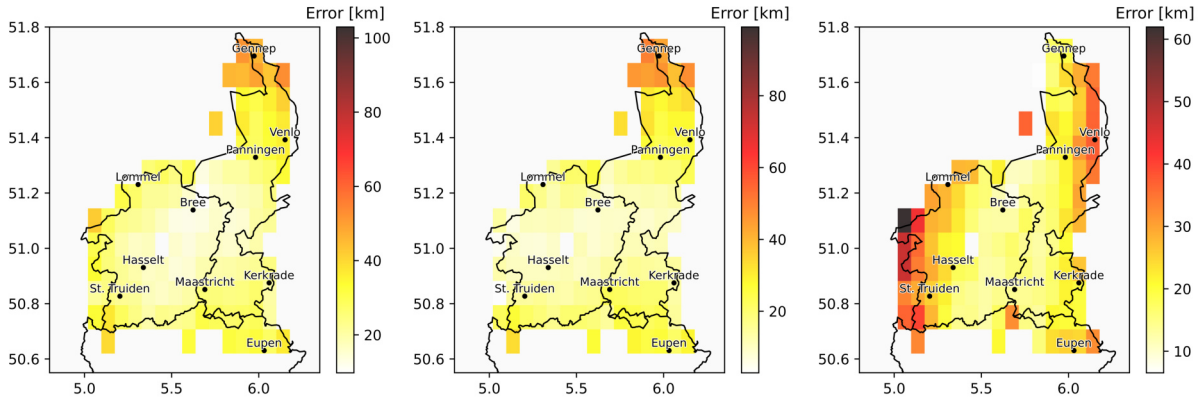


Figure 22: Physical distance errors (left), latitudinal (middle), and longitudinal (right) on the test set.

We notice that the errors are relatively homogeneously spread, although the accuracy of localities near the outer border is slightly larger. The main anomaly is the northern tip of Dutch Limburg (well within the Kleverlandic region which is not traditionally considered Limburgish), which results in a near doubling of the distance errors. The most interesting results can be found in the longitudinal (horizontal) decomposition of the distance errors: the localities near the western and eastern boundaries are significantly less accurate. This again corresponds with the notion of traditional dialectology: the main variation in Limburgish is vertical and not horizontal, and the regions with the largest error lie either outside the traditional Limburgish area or are within transition zones. We further elaborate by plotting the main isoglosses and increasing the resolution by employing a scatter plot in Figure 23.

The largest longitudinal errors correspond closely to the western area beyond the Tonality line and the idealized Gete line. This can be expected as these isogloss bundles form the most distinct boundary region within the Limburgish area. The southern half is less accurate than the northern half, which is a surprising result considering traditional dialectology: the southern half comprises the more continuous West-Limburgish transition zone and is considered Limburgish, while the northern half is considered Brabantian transitioning into Limburgish. These Brabantian-Limburgish dialects have a more isolated position historically and are traditionally considered to have undergone more Brabantian influence, being above the Uerdinger line. In Dutch Limburg, the decrease in accuracy also matches the Uerdinger line into Kleverlandic, although less noticeably. We do notice a strong west-east divide near Venlo that resembles the Meuse river (Figure 24) [5].

This sudden decrease in accuracy does not seem to be confined to either side of the Meuse, nor is this repeated anywhere else along the Meuse. Bakker (Bakker, 2016) found that no isoglosses or dialect boundaries follow the Meuse in this region. If we instead consider the settlements in this region (Bakker, 2016), we find a continuous urban zone along the Meuse, matching exactly with the increase in errors of our LSTM architecture (Figure 25). In his research, Bakker concluded that Venlo was the only major city in the area, radiating its cultural influence along the northern Meuse valley until the 18th century. A century ago this was also proposed by Schrijnen (Schrijnen, 1907) in his article on the "mich-quarter". A thorough historical

[5]The GIS data of the Meuse is courtesy of (Regions).

review can be found in (Crompvoets, 1998). This area around the Meuse represents a discontinuity in the Kleverlandic region which is still actively influenced in a process of "vervenlosing" (Venlo-fication) (Bakkes et al., 2007) (Bakker, 2016).

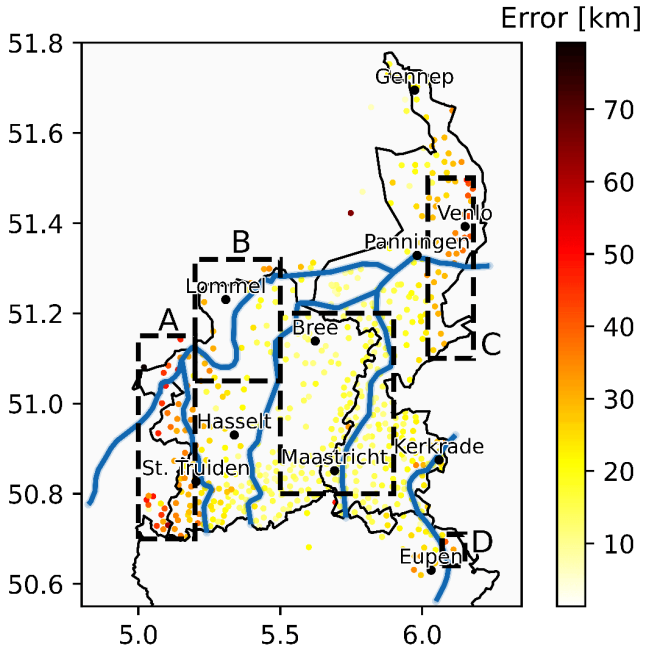


Figure 23: Longitudinal component of the physical distances on the test set.

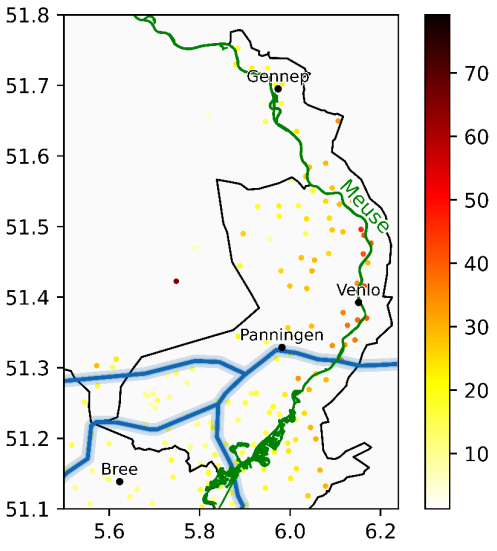


Figure 24: Longitudinal distance errors in the north of Dutch Limburg.

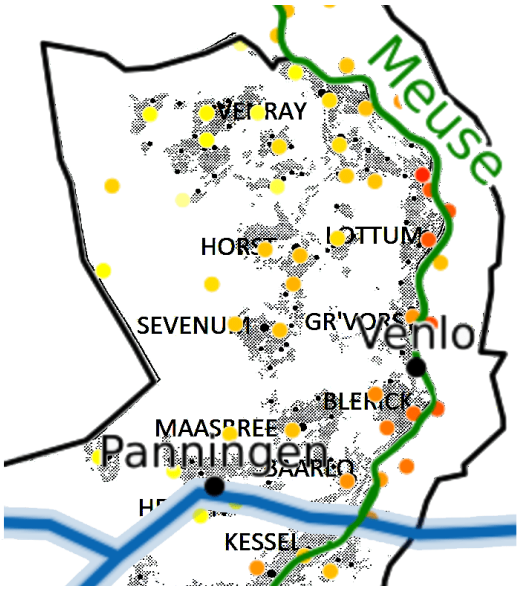


Figure 25: Errors overlaid on a map of settlements ca. 1806, from (Bakker, 2016). The dots in decreasing size represent 250, $25-250$ and <25 houses.

Another interesting result is the northeast of Liège, the decrease in accuracy seems to match the Benrather

line, which might influence the accuracy of the architecture. The overall accuracy of the LSTM architecture seems to be homogeneous for the entire Limburgish area according to traditional dialectology (both Western-, Central- and East-Limburgish), which is a very surprising result.

From these results, we could hypothesize that the LSTM architecture generalizes better on the "smooth" area that is considered to be Limburgish in the traditional sense rather than transitional dialects into other language clusters. These transitions sometimes correspond to discontinuities caused by non-linguistic factors (e.g. the Gete line corresponding to the historical Liège-Brabant border or the river Meuse in Kleverlandic near Venlo). To verify these hypotheses, we will further analyze the characters and word features that lead to this error distribution. We have marked four regions (A, B, C, and D) (Figure 23) that show a noticeable increase in errors, in which we will study the longitudinal distance errors on a character and word level.

5.4.1 Region A

We have selected all the entries in the test set whose true coordinates lie in region A, together with their longitudinal distance errors (rounded to 3 decimals, in km). We retrieve the 10 words with the highest and lowest longitudinal errors respectively:

Word	rǭmǭkər	slẹ̄t	snīēf	afblārə	krøbē̜tər	mø̜̄.l	ke̜rəkho̜f	sxē̜rdər	zwǫwər	ę̄nspanə
Error	0.003	0.037	0.043	0.047	0.085	0.104	0.114	0.122	0.122	0.138

Word	xehä:m	sxāf˛baŋk	rūūzə	pōēl	hętsəls	hèùte	hàspələ	rãm	káá	kêver
Error	109.865	108.361	106.553	106.237	104.949	103.937	102.875	102.69	101.645	101.086

The 10 words with the highest errors yield much poorer results than the 10 words with the lowest errors. This is likely caused by a larger presence of variation in the training data: we find frequency means of 206 and 151 respectively. Because of this, the model might have trained better in differentiating the variation. This is a trend that occurs for all regions, as well as the entire test set. We do notice that the words with the highest errors tend to have more characteristics uniquely specific to the West-Limburgish transitional dialects: the long ä in xehä:m, the long ūū in rūūzə or èù in hèùte, etc. The words with the lowest errors do not have such typical characteristics. To study the influence of phonology on the errors in more detail, we count for each character what the mean longitudinal distance error is. We only evaluate characters that have more than 10 occurrences and consider the 10 characters with the highest mean longitudinal distance error (the test set contains 6646 words in region A):

Character	ù	`	ò	ˇ	ǭ	˛	.	z	ē	v
Error	30.4	26.9	23.6	23.5	23.4	23.0	23.0	22.5	22.5	22.4
Frequency	15	23	36	23	227	67	171	409	338	648

The ù and by extension ` are unique West-Limburgish transitional sounds, they are featured in the test set in words such as brèùn (brown) and krùise (cross). The ǭ is not necessarily uniquely West-Limburgish transitional, but the context in which it is used is uniquely typical, i.e. it replaces the long vowel in certain words such as dāx (day) or xrāf (grave). We illustrate with some examples in the test set using this sound: dǭak (roof), wǭt (chisel cut), vǭt (vat). The ē again corresponds to some uniquely local phenomena: long vowels where the rest of the Limburgish area has diphthongs. We find this in for example the words ē̜zər (iron), bē̜təl (chisel) and hē̜fkē̜r (horse wagon). Finally, the z in some instances corresponds to words that have an r-deletion, which is a typical Brabantian phenomenon and present in dialect words such as zwętə (black). All these 10 characters can be linked to uniquely West-Limburgish phonology, with the exception of ò, ˇ, ˛ v. From these results, we think that is likely that these Brabantian influences in combination with the discontinuous Gete bundle result in poorer performance of the architecture.

5.4.2 Region B

Word	hǭəsvę̜lt	zawtən	mek	smē.t	updű.n	zē.vələ	nen	sxę̜lən	krūzəlōr	vē̜rəkə
Error	0.003	0.01	0.012	0.014	0.028	0.029	0.043	0.043	0.044	0.026

Word	tīpələ	dǫrs̩vlūr	kêver	trêøtere	xláns	xrôetə	afzāxə	krōnrat	sxāf̩baŋk	lǫri
Error	92.635	87.663	86.457	85.454	84.728	83.63	82.787	81.986	81.935	81.254

We again find the same pattern as for region A, the words with the lowest errors have significantly more variation; a frequency mean of 619 in comparison to 96. We repeat the character analysis:

Character	`	î	ö	û	ù	-	ò	ī	ū	˘
Error	32.5	28.2	27.6	27.1	26.7	25.3	24.3	23.9	23.9	23.8
Frequency	13	12	23	14	11	10	15	68	76	24

The largest errors are associated with ` and come from the words se`dęl (serradelle plant, 49.02 km) and w`èx (gone, 50.25 km) with the remainder of occurrences having errrors below 18 km. Both words are very common without much variation and likely caused the high error. The î is associated with verhûîze (to move, 67.07 km), zîên (to be, 60.82 km) and bûîke (to burp, 61.85 km). All of these examples contain typical Brabantian diphthongs or triphthongs where Limburgish would have monophthongs. The ö is associated with words such as böl (pouch, 73.66 km), dörə (to dare, 68.52 km) which are again widespread without much variation. The û is once again associated with Brabantian diphthongs and triphthongs such as bûîke and verhûîze (both same as above). The other characters cannot be associated to any unique Brabantian-Limburgish phonology. We conclude that the slight increase in mean longitudinal errors in this region cannot be attributed to local phonology and that the network generalized better in this region than is the case for the West-Limburgish transitional dialects.

5.4.3 Region C

Word	smālt	haŋk	laŋkęr	ɛiɻkəmbuəm	xêrt	éérbéés	entə	bezwāōrd	plistərə	huəməs
Error	0.02	0.026	0.026	0.167	0.173	0.184	0.226	0.26	0.287	0.311

Word	zwøɻm	pərtōl̩	dràəxən	kīs	standə	sxøtstǫk	jaske	īəmər	stō?ə	sxɛrəp
Error	117.884	110.443	109.043	101.557	100.036	99.898	99.703	97.5	96.281	94.585

The words with the lowest errors now have a comparable frequency to the words with the highest errors, 213 and 194 respectively. This means that we can directly compare the words based on their features. The words with the highest errors all contain features which are unique to Kleverlandic: the l in smālt (lard) which in the rest of the Limburgish regions is vocalized or velarization: haŋk (hand) instead of a -nd as is the case in the rest of Limburgish, buəm (tree), huəməs (high mass) instead of other variants which feature o or ou, the differentiation between long vowels and stretched vowels such as in bezwāōrd which in the rest of Limburgish would be a monophthong. The words with the highest errors do not contain such typical features and vary little in the Limburgish area. The only words that contain features that could indicate a northern origin are jaske (diminutive of jacket) due to its lack of umlautization or stō?ə due to the use of a glottal stop. Analyzing the characters:

Character	ü	î	˘	é	.	ī	:	ó	h	ǭ
Error	28.6	26.3	24.3	24.1	23.2	23.2	22.9	22.6	22.4	22.4
Frequency	12	13	74	65	98	113	13	38	467	145

The ü is an outlier as only one word contains a significant error snüpke (candy, 65.65 km) as it is nearly uniform for the entire Limburgish area. The î corresponds to the words kóefmîês (crested tit, 82.45 km) and lîêf (body, 52.28 km) and are not typically Kleverlandic (they are generally short vowels) but originate in the city dialect of Venlo (Bakkes et al., 2007). The é corresponds to loépe (to run, 81.81 km), vérk (work, 80.78 km) and sléeèj (rough, 67.44 km). The paragoges in loépe and sléeèj originate in Venlo as most Kleverlandic dialects do not have the paragoge. The v in vérk seems to be a misspelling and is likely an outlier. The . corresponds to mu.nd (mouth, 82.50 km), āfdā.k (canopy, 60.40 km) and hu.d (hat, 58.84 km). The first word is again typical of Venlo but the other two are too general and contain no regional characteristics. The ī can once again be strongly associated to the influence by Venlo: bīst (beast, 80.51 km), blīzə (chaff, 90.13 km) both of which in Kleverlandic generally contain ē. The remaining characters except for the : and ǭ can also be attributed to the influence by Venlo. We conclude that it is likely that the irregular/discontinuous influence of Venlo in contrast with the general evolution of Kleverlandic dialects results in poor generalization by the network and therefore higher mean longitudinal errors.

5.4.4 Region D

Word	jrāvə	miənə	ējəmāt	kɛtsštǫk	lē.m	blo:mə	hˈøi̯	ejəmāt	elfmeter	poraj
Error	0.997	1.978	3.15	7.243	7.508	9.748	10.15	10.164	10.909	12.966

Word	hoų̯.f	fē̯lt	rījə	ōd̯ər	zęi̯døx	høltərtøl	hāx	veŋər	hō.f	lant
Error	75.147	74.86	68.581	68.266	66.882	63.47	60.291	57.0	55.802	53.29

Since region D only contains 48 words in the test set, we will not perform a character analysis but instead consider the feature of the words with the lowest and highest errors. We notice that the words with the lowest errors are significantly lower than the other regions: this is expected as the studied region is mostly beyond the Benrather line and due to these dialects being Ripuarian and not generally considered Limburgish they have a very distinct phonology and are underrepresented in the WLD. The variation of the words is again relatively equal, 311 for the words with the lowest errors and 497 for the words with the highest errors, which allows us to compare the words based on their features. The words with the lowest errors are clearly Ripuarian in origin: the `j` instead of `g` in `jrāvə` (to dig) and `ējəmāt` (pickled) but also Ripuarian vocabulary such as `kɛtsštǫk` (daisy). The words with the highest errors are again words with little variation `rījə` (to drive), `zęi̯døx` (milk strainer filter) or `hāx` (hedge). Interestingly, typical Ripuarian/High German vocabulary also results in high errors, possibly due to low frequencies: `fē̯lt` (field) and `høltərtøl` (elder plant). We conclude that it is likely that the model performs poorly in this region due to the distinct Ripuarian properties and their underrepresentation in the WLD, rather than due to the existence of any discontinuities.

5.4.5 General character analysis

We will now analyze which words, characters, and features result in good and poor performance by the model, this time for the physical distance errors (and not solely the longitudinal errors). We find the 10 words with the lowest and highest errors in the entire test set:

Word	slōi̯	spatoren	kǭf	botsə	kē̯	kərnēnə	hǭk	ontbeje	plōx	xəpløx̯ˌdə
Error	0.035	0.054	0.087	0.099	0.106	0.117	0.123	0.129	0.139	0.154

Word	lantɛr	bǫjəmwesə	bŭlt	sløę̄i̯	dø:pə	trę̄xtər	āxtpǫndər	màjònéés	ste̯l	afsè.jd
Error	118.538	117.478	115.06	110.46	109.796	109.485	108.276	105.745	105.4	105.131

We notice that the words with the lowest errors have extremely low errors in comparison with the four regions that we have studied. This is because listing the words with the lowest errors at large scales (such as the entire test set) will bias the results towards words that lie close to the original coordinates due to chance. For the words with the highest errors we find that they are all words that have very little phonological variation in the Limburgish area, which likely makes it more difficult for the model to learn any geographic patterns. We present the characters that correspond to the highest errors with at least 10 occurrences:

Character	c	ú	à	ä	-	ô	ù	õ	a	d
Error	25.9	25.7	25.4	24.4	24.0	22.8	22.6	22.5	22.2	22.0
Frequency	159	28	289	153	129	333	115	24	13017	7660

We notice that the `c` corresponds to words that are poorly normalized such as `victoria` (76.46 km), `cent` (61.51 km) `decimeter` (57.36 km) as the `c` is not part of Limburgish phonology. These words therefore are mostly loanwords from English and Romance languages, where the c is more commonly used and are therefore impossible for the model to locate. The `ú` is barely used and mostly corresponds to words that are not fully normalized such as `véúrmònd` (legal guardian). Similarly, the `à` corresponds to mostly poorly normalized words such as `màjònéés` (mayonnaise, 105.74 km).

The `ô` is the first character that can be meaningfully interpreted: it is featured in words such as `pĭětôêt` (very dead, 85.65 km), `buskrôêt` (gunpowder, 71.03 km) and `môêdər` (mother, 65.56 km). This character is mostly associated with Brabantian influence and the Venlo-Mich quarter region, as is the case for the first two words (from Niel-Bij-St-Truiden and Venlo respectively). Similarly, the `ù` is associated with words of

Venlo influence such as `hèùj-wéér` (hay weather, 69.78 km) and `rèùstə` (to rest, 79.90 km). The remaining characters `a` and `d` are among the most common characters, hence their place in the list. If we then consider the characters associated with the lowest errors:

Character	ű	š	ʔ	ő	ˑ	ˇ	.	û	˙	˝
Long. error	9.1	12.6	12.8	13.0	13.4	14.3	14.7	14.8	15.0	15.0
Frequency	68	2684	82	46	106	579	3946	195	29	13

We immediately notice that the `š` has a low error, this is likely because it is strongly associated with the important Panninger isogloss. We also notice the glottal stop in third place, which is only common in the small Brabantian-Limburgish area and therefore a typical local characteristic. We further notice that half the characters featured are diacritics, which indicates that diacritics indeed play an important role in identifying Limburgish dialects based on phonology. To verify this claim, we count the percentage of diacritics per word and plot the physical distance errors in Figure 26.

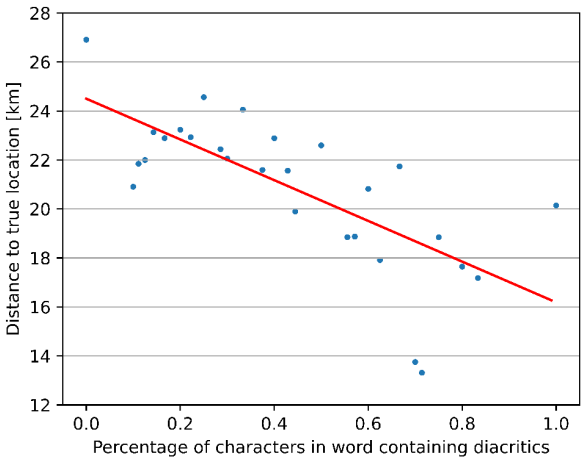


Figure 26: Physical distances with respect to percentage of diacritics.

We notice a clear decrease in distance errors as the percentage of diacritics increases, with the highest error corresponding to 0 diacritics. Computing a linear regression, we find an R^2 value of 0.52 from which we can conclude that a reasonable linear relation exists between diacritics and accuracy.

5.4.6 General word analysis

Now, we will analyze the results on a word level. Firstly, we expect that longer words will result in lower physical distance errors, which is the case in general DID. We visualize the distance errors in function of the word lengths in Figure 27.

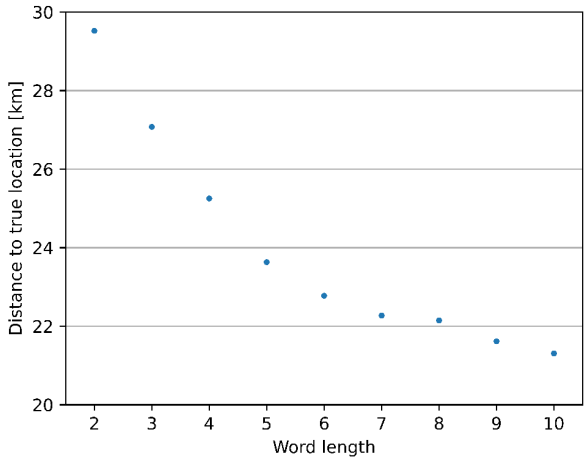


Figure 27: Physical distances with respect to word length.

We find that longer words clearly result in a lower mean distance error. Secondly, we would like to analyze whether more frequent words result in lower errors. Since no elaborate frequency data exists for Limburgish, we will assume that the concepts of the WLD are relatively similarly distributed as in Dutch and use the *SUBTLEX-NL* frequency list (Keuleers et al., 2010). We consider all entries in the test set that have a keyword that exists in the top 1M words in SUBTLEX-NL and plot their errors against their respective zipf[6] values (Figure 28).

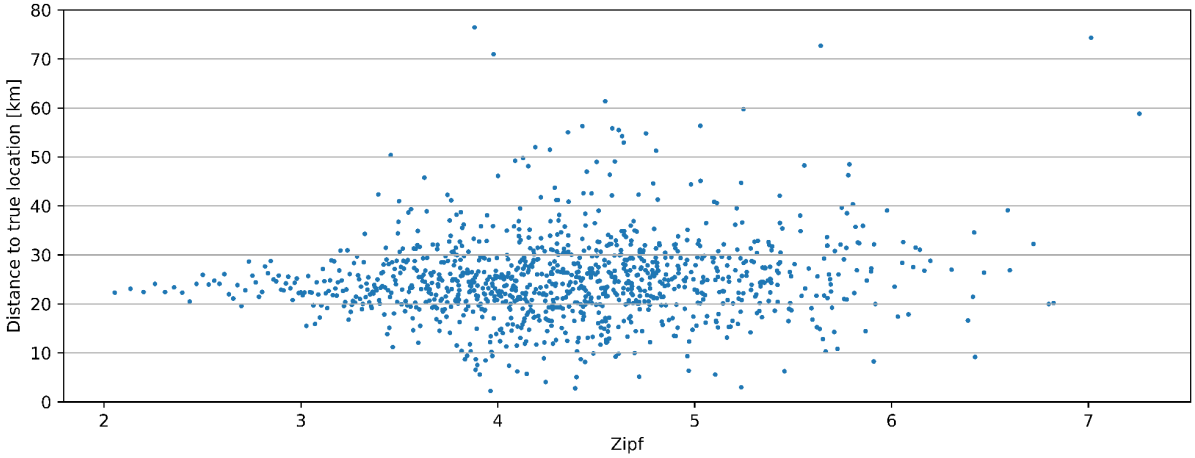


Figure 28: Physical distances with respect to zipf frequency.

We do not notice any correlation and since $R^2 = 0.015$ we can conclude that there is no linear relation between the accuracy of the model and the frequency of words. One possible cause could be that the frequencies of

[6]The Zipf scale describes a logarithmic scale for the frequency of vocabulary in natural language settings. Zipf values of 1 and 7 represent very low and very high-frequency words, respectively.

concepts do not sufficiently carry over from Dutch to Limburgish as such frequency datasets measure the frequency of individual words and not abstract concepts. Additionally, frequency lists are not known to be cross-lingual. For example, even for the 14 most common words on the Dutch, German, and Limburgish Wikipedia (Table 13), as found using the Python module (Semenov), we already find that the most frequent words are not aligned:

Dutch	de	van	in	het	een	en	is	op	voor	met	werd	door	zijn	hij
German	der	die	und	in	von	im	den	des	mit	das	er	wurde	dem	als
Limburgish	de	in	en	vaan	is	van	op	mèt	zien	es	die	ouch	veur	te

Table 4: Fourteen most frequent words on the Dutch, German and Limburgish Wikipedia projects.

Next, we are interested in the influence of the semantic domain on the physical distance errors (Table 5). We use the fact that the volumes of the WLD are organized in volumes covering semantic domains.

WLD volume	Part	Accuracy [km]
Miner	II	18.083
Cultivation of cereal crops	I	21.242
Pasture cultivation	I	21.266
Farm, yard and vegetable garden	I	21.532
Miller	II	21.667
Agricultural vehicles	I	21.689
Horse harness	I	21.774
Birds	III	21.820
The home	III	22.131
Cultivation of tubers and other crops	I	22.486
Farm, commercial buildings	I	22.850
Food and drink	III	22.857
Cattle, milk and butter, animal husbandry in general	I	23.269
The horse	I	23.318
Lumberjack, carpenter, etc.	II	23.345
Small livestock and poultry	I	23.695
The material and abstract world	III	23.896
Blacksmith, plumber, coppersmith	II	24.039
Personality and feelings	III	24.208
Clothing and body care	III	24.229
Fertilizing and plowing	I	24.358
Harrowing and dragging	I	24.379
Other animals	III	24.426
Movement and health	III	24.709
Flora	III	24.977
Bricklayer, carpenter, etc.	II	25.028
Tailor, seamstress, etc.	II	25.137
Social behavior, school and education	III	25.266
Party and entertainment	III	25.621
Butcher and baker	II	25.705
Family and sexuality	III	25.970
Farmlands	I	25.992
Beer brewer and syrup maker	II	26.035
Beekeeper, straw or bentgrass weaver	II	26.361
Church and faith	III	26.839
The human body	III	27.219
Shoemaker, saddle/harness maker	II	28.100
Potter, brick maker, etc.	II	28.791
Turf extractor and ore miner	II	30.095

Table 5: Mean distance errors per WLD volume.

We notice that the volume with the lowest mean distance error (by a margin) is mining. This is to be expected, as this relates to the very specific, relatively new jargon of the 19 Limburgish mines (Crompvoets and van de Wijngaard, 1989). This jargon often only exists in isolated localities and has a large underlying German and Walloon basis, which likely makes their geographic identification easier. This is to be contrasted with the least accurate volume: ore miner and turf extractor. This semantic volume does not include mining jargon from the recent Limburgish mines but rather the very local jargon of the zinc mines near Kelmis/Vaals which have been operating since the Middle Ages (Crompvoets and van de Wijngaard, 1989). This jargon has a large Ripuarian basis and is not spread outside of the narrow mining region, which could result in a discontinuity as we have discussed in the sections above. The turf extraction vocabulary is mostly located in the north of Dutch Limburg, which has a specific discontinuity for turf extraction: the region saw the creation of so-called "turf colonies", where workers from all over the Netherlands would migrate to, influencing the dialects (Crompvoets and van de Wijngaard, 1987). This behavior seems to indicate that if discontinuities exist (either in vocabulary or phonology), but are spread over a large enough region, then the model accuracy

increases (as is the case for recent mining jargon). More localized discontinuities, even if they are relatively older such as turf extraction in the north of Dutch Limburg or zinc mines in the south-east of Dutch Limburg, instead result in a decrease in accuracy.

We further notice that terms generally related to farming (part I of the WLD) have the lowest errors when compared to the two other categories: this vocabulary is generally the oldest and likely preserves the geographic phonological variation better. Furthermore, these agricultural terms are widespread with a high regional diversity, containing more geographic information. The volumes with the highest errors are more recent terms with less phonological variation such as church and faith, the human body, family, and sexuality. Here we also find non-agricultural professions such as beekeeper, potter, etc.

Finally, we would like to study the influence of other factors such as sentiment and age of acquisition. We would expect that words with a low age of acquisition result in a higher error in the DID task: a lower AoA is correlated with shorter, less complex words (Botarleanu and et al., 2022) which in turn can encode less phonological variation. (Moors and et al., 2012) computed the age of acquisition values for 4300 Dutch words as well as the norms of activity/arousal, power/dominance, and valence of these words. We will also investigate whether words with either a very positive or negative sentiment can be correlated with the accuracy of the model, which serves as a proxy for the phonological variation of the Limburgish words. We will again consider the WLD test set entries whose keywords can be matched with Dutch words from the dataset (Moors and et al., 2012) and find Figure 29. Here we make the strong assumption that word-based norms of arousal and power as well as age of acquisition can be transferred from Dutch to Limburgish.

We notice no correlations between valence, arousal, dominance, and the mean physical distance errors of the dialect identification task. It is possible that no such correlation exists, or that we simply cannot transfer sentiment values from Dutch words to Limburgish words. We do notice that a very low age of acquisition (below 5 years) seems to result in a higher distance error, possibly supporting our original hypothesis. Apart from the initial increase in distance error, we do not notice any correlation.

5.4.7 Conclusion

In conclusion, we have found that longer words, containing more diacritics, result in more accurate dialect identification. A manual analysis reveals that prevalent words with little phonological variation are more difficult to locate geographically, while certain typical dialect markers increase results. Specifically for Limburgish dialect identification, we find that the horizontal distance error is larger than the vertical distance error, indicating that horizontal variation is larger than vertical variation, in line with notions in traditional Limburgish dialectology. Certain border regions such as the transitional region to Brabantian, the Kleverlandic dialects, and the Benrather dialects result in poorer dialect identification. This is attributed to discontinuous phenomena in these regions which are not consistent with the rest of the studied area. Other discontinuous phenomena, such as multilingual, specific mining jargon result in more accurate dialect identification, likely due to a larger frequency of vocabulary and geographical spread.

Semantically, mining jargon and agricultural terms yield the best dialect identification results as opposed to abstract concepts and professional terminology. There does not seem to be a correlation between any sentiment and the accuracy of dialect identification, although it is unclear what the effect of using a Dutch dataset for a Limburgish task is. We found a weak correlation between an early age of acquisition value and a higher dialect identification error, but no further correlation.

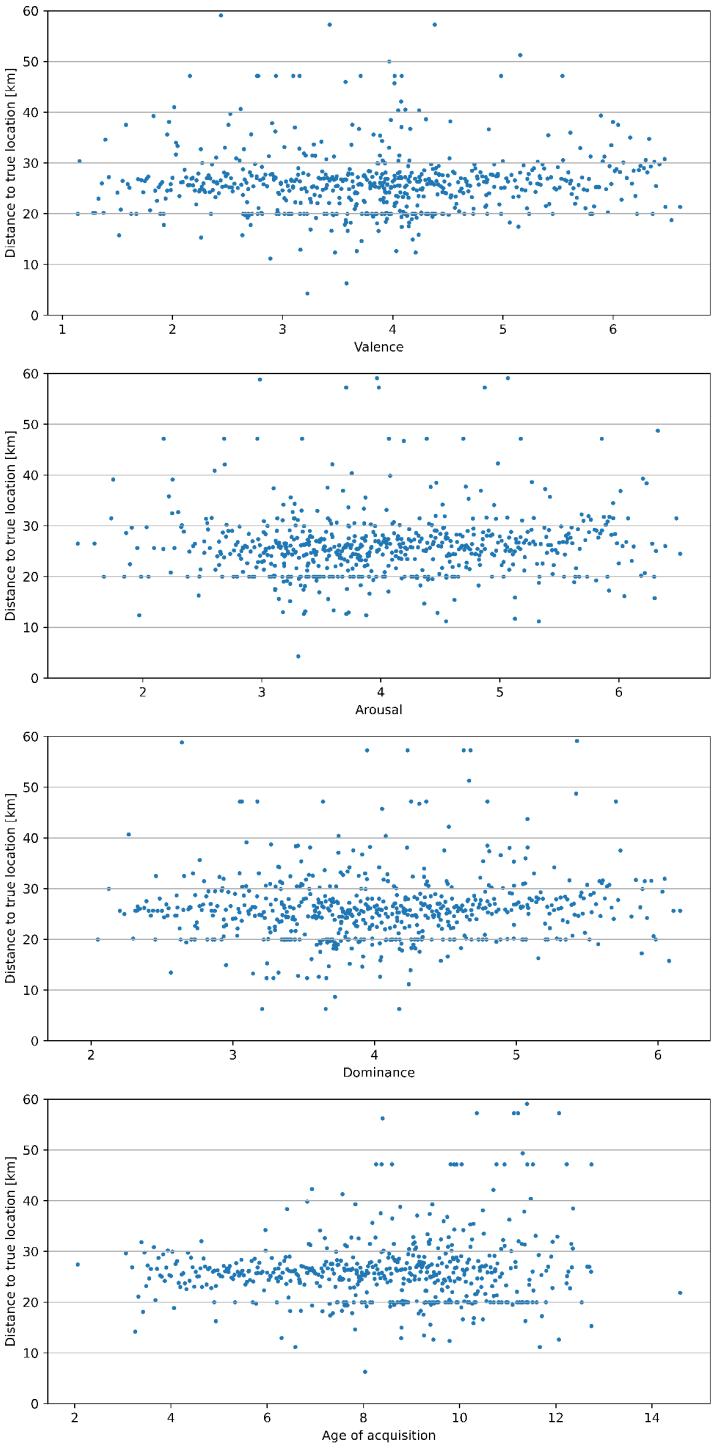


Figure 29: Physical distances with respect to norms of valence, arousal, dominance, and age of acquisition.

6 A novel adaptation of the transformer: the Geographically Embedded Transformer (GET)

Transformer-based architectures dominate seq2seq tasks in NLP such as machine translation, summarization, text generation, etc. However, the native transformer architecture (Vaswani et al., 2017) or its many adaptations (Tay et al., 2022) do not generalize well for language variation or multilingual data. The XLM model (Lample and Conneau, 2019) solves this by modifying the input of the transformer into a cross-lingual vector, such that cross-lingual training is possible. In for example the case of machine translation, the first half of the input vector consists of the input language and the second half of the target language, leading to parallel training. This approach contends with the performance of the original BERT architecture (Devlin et al., 2019). BERT has been adapted to a multilingual setting where it has been pre-trained on 104 languages (mBERT) (Face, b) while XLM has been partially adapted into (XLM-RoBERTa) (Face, a), using a BERT architecture for the masking part, on over 100 languages. XLM-RoBERTa is the current state-of-the-art in multilingual classification, sequence labeling and question answering (Conneau and et al., 2020)(Face, a).

XLM-RoBERTa and all previous implementations however suffer from a "curse of multilinguality" (Conneau and et al., 2020): even though low-resource language models can be leveraged using high-resource languages that are similar enough in a multilingual training setup, expanding the model to more languages results in an overall decrease in model accuracy. This remains an open problem in the task of large multilingual machine translation or *universal neural machine translation* (*UNMT*) in a cross-lingual setting, where high-resource languages significantly outperform low-resource languages. However, low-resource languages can be oversampled to significantly increase their performance, at the deterioration of high-resource languages in a transfer-inference trade-off (Arivazhagan and et al., 2019). Additionally, low-resource languages can only be leveraged through the inclusion of similar high-resource languages, although excluding language families that do not have sufficiently large corpora.

These approaches provide an additional dimension to training language data using transformers, although currently only for countably many languages with sufficiently large corpora. Linguistic data, however, does not only vary per a limited number of fixed language classifications (i.e. English, French, Swahili) but also temporally (age of speaker or present time), geographically (dialects), by register (formal to non-formal), code-switching (multilingual speakers), etc. Many of these factors represent continuous variables that are difficult to reconcile with transformed-based architectures or any current NLP architectures.

We therefore propose a modification to the normal transformer architecture (Vaswani et al., 2017): by enhancing the input of the encoder and decoder blocks with additional linguistic information, the performance of many NLP tasks could be increased. We will test this modified architecture in the case of geographic language variation in the Limburgish language, as the WLD dataset allows for a very granular geographic study of the phonological variation of Limburgish. We will from now on differentiate the normal transformer architecture from the *Geographically Embedded Transformer* or *GET*. The modification to Vaswani et al.'s transformer is visualized in Figure 30.

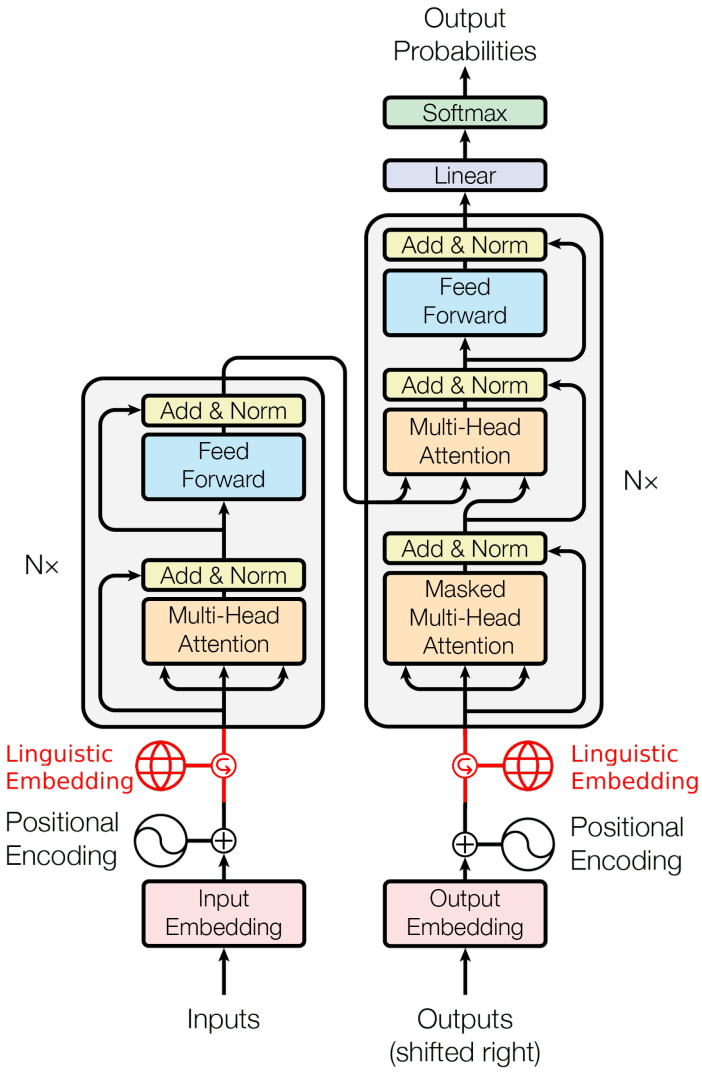


Figure 30: Transformer architecture diagram taken from Vaswani et al., 2017 (Vaswani et al., 2017) and modified (in red) to include the embedding of additional linguistic information.

In the normal transformer model, both the encoder and decoder block take the embedded input and apply a positional encoding: since the attention mechanism considers the input tokens separately, this encoding ensures that the information of the original position is not lost. The only architecture that would allow the encoding or additional embedding of extra linguistic information therefore necessarily has to occur after the positional encoding and before the attention mechanism, and cannot be passed as a separate token if the linguistic information concerns the entire input vector (e.g. age of speaker, geographic information of dialect, formality, present time). We choose to implement this additional information by expanding the dimension of the input embedding and including the additional information there.

For our implementation, we again consider words in the WLD that are fewer than 10 characters and we embed each character into a dimension N after the input embedding and positional encoding, yielding vectors of dimension $(\text{batch size}, 10, N)$. After the linguistic embedding, the dimension will be $(\text{batch size}, 10, N+2)$, where we have appended two additional rows in the last index, corresponding to the normalized (x, y) coordinates of each WLD entry. Only the first column contains the coordinates by design, as the output embedding during the decoding phase necessarily starts with only a single [START] token and we wish to

keep the embedding sparse. The input vector after the linguistic embedding can be visualized as:

$$\begin{bmatrix} e_{1,1} & e_{1,2} & e_{1,3} & e_{1,4} & e_{1,5} & e_{1,6} & e_{1,7} & e_{1,8} & e_{1,9} & e_{1,10} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ e_{N,1} & e_{N,2} & e_{N,3} & e_{N,4} & e_{N,5} & e_{N,6} & e_{N,7} & e_{N,8} & e_{N,9} & e_{N,10} \\ y & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ x & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where $e_{i,j}$ represents the floats of the embedded vector after the input embedding and positional encoding. We build this architecture using TensorFlow's functional API.

The characters in each WLD entry are vectorized to an integer between 0 and the total character vocabulary size minus one, 93, which currently consists of the characters

-	.	:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
q	r	s	t	u	v	w	x	y	z	-	à	á	â	ä	è	é	ê	ë	ì
í	î	ï	ò	ó	ô	õ	ö	ø	ù	ú	û	ü	ā	ē	ę	ī	ŋ	ō	ő
š	ū	ű	ž	ə	ǫ	ǭ	ȳ	ə	ɛ	ʔ	◌˂	◌˃	◌˙	◌˱	◌˲	◌̃	◌̄	◌̆	◌̇
◌̋	◌̌	◌̨	◌́	◌̛	◌̣	◌̥	◌̭	◌̰	◌̩	ẽ	·								

as well as the start token $ and end token □. The start tokens are used as the first input for the decoding block and the end token is used to indicate when the decoding should stop to any search algorithm used in conjunction with the transformer.

We will consider various tasks in NLP using GET such as normalizing the geographic variety of spelling conventions, highly precise dialect machine translation and the reproduction of traditional dialectology maps.

6.1 Geographic spelling normalization

The entries in the WLD can vary significantly along spelling conventions. There is reason to believe that the conventions are geographically based, as local Limburgish dictionaries and traditions can set the standard for spelling in specific regions. On other occasions the overarching Veldeke spelling is used although its conventions can vary, since the finalized version of 2003 (Bakkes et al., 2023) was not available to the respondents at the time of the surveys that constitute the WLD and its use was less frequent than it currently is. Sometimes the spelling contains noise, as diacritics can merely indicate that a character's pronunciation differs from Dutch orthography (Weijnen et al., 1983). This normalization task is particularly challenging because there is no reference for "correct spelling", as is the case in typical spelling correction tasks. Additionally, it is a priori not clear which spelling variation constitutes actual phonological variation and which only noise from incorrectly recording certain dialect words into phonetic transcription.

As there is no need to undersample the dataset to make it geographically representative for the normalization task, we will repeat the preprocessing of the WLD without the undersampling step. The WLD now consists of a total of 1 888 458 entries which are split into a normalized subset of 982 133 entries and an unnormalized subset of 805 208 entries. By construction, we assume that the normalized subset contains only "correct" entries in IPA notation, while the unnormalized subset contains the remaining entries, consisting of other conventional spellings. The dataset for this task is generated as follows: for each word in the unnormalized dataset, all entries in the normalized dataset that match the Dutchified keyword are selected. All matches that are located less than 0.5 km from the unnormalized word's location are then selected. The entries of this new normalization dataset then consist of these unnormalized words matched with normalized words within their proximity. Because these words are geographically very close, we can assume that the variation in spelling is due to convention and that they have the exact same phonology, only now in correct phonetic spelling. Finally, all words that are more than 10 characters are omitted and the words are vectorized. The total size of the normalization dataset is 118 440 entries. The dataset could be enlarged by increasing the search radius, but this results in less certainty regarding the discrimination between phonological variation and spelling variation. All entries except for five come from the exact same locality (no distance between

them), and the average distances between the five unnormalized and normalized words' localities is 0.39 km. This dataset has very few unnormalized-normalized pairs, as 0.5 km is below the typical distance between two localities in the WLD. We also generate a second dataset by increasing the radius to 6 km (a typical distance between localities), resulting in 278 867 entries where 159 445 are from differing localities, with an average distance per unnormalized-normalized entry of 3.47 km.

Below in Table 6 we list some examples of the first normalization dataset, where both the unnormalized and normalized words come from the same locality:

Unnormalized word	Normalized word	Locality	Translation
vief	viêf	Echt	five
kroedwès	krutweš	Tungelroy	folkloristic herb
waere	wĕre	Heel	to become
aafdoeë	āfdūə	Vlijtingen	to mow the grass
schoppen	sxopə	Houthalen	to kick

Table 6: Examples of the normalization dataset.

In the first entry we see that an unnormalized `ie` becomes `iê`. This diacritic is not contained in the WLD's set of diacritics (see Appendix C) and likely only expresses that the pronunciation is different from Dutch orthography. The second example correctly normalizes Dutch `oe` →`u` and `d` →`t`. It corrects the missing diacritic on the `s` but incorrectly loses the diacritic on the `e`. The third and fourth examples correctly normalize the `ae` and `oeë` in Veldeke spelling and the last example correctly converts Dutch orthography to IPA.

6.1.1 Normal transformer as baseline

To test whether our GET transformer outperforms the normal transformer architecture with respect to the normalization task, we first train the traditional transformer on this dataset. We split the normalization dataset of radius 0.5 km into $80-10-10$ train, validation, and test datasets and perform a hyperparameter search using Optuna. We vary the number of stacked encoder and decoder blocks from $1-5$, where for simplicity we assume the same set of parameters: the embedding dimension varied from $1-1024$, the latent dimension from $1-1024$ and the number of attention heads from $1-16$. We run 100 iterations of different parameters until convergence, i.e. more than 5 epochs without a strict 10^{-3} improvement in accuracy. We use the Adam training method and Sparse Categorical Crossentropy as metrics and a batch size of 512. We visualize the results of the hyperparameter search and first consider the influence of stacking encoder and decoder blocks in Figure 31.

We notice that stacking more than 2 layers of encoder and decoder blocks does not significantly improve results, for which we refer to the analysis of stacking encoder/decoder blocks in section 5.3. We also note that the strongly diverging results in 3,4, and 5 stacked encoder/decoder blocks are a result of the Optuna algorithm. Considering the attention heads, we find in Figure 32 that the number of attention heads does not seem to significantly impact the results, which could indicate that a single attention head mechanism is sufficient for the task of character-based NMT in Limburgish. The results for dimensions of the embedding and latent spaces can be found in Figure 33 and Figure 34.

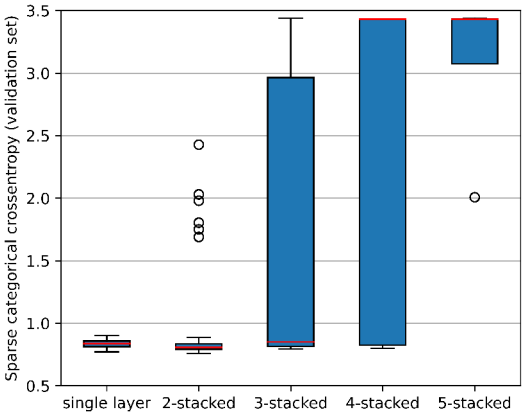


Figure 31: Boxplots of Sparse Categorical Crossentropy loss per number of stacked encoder blocks, determined on the validation set.

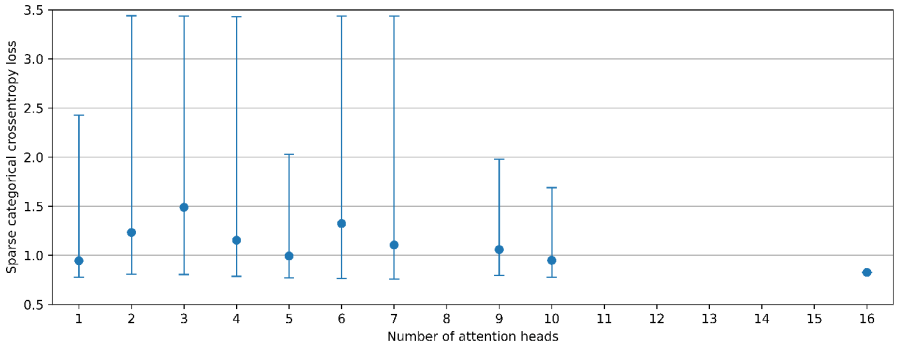


Figure 32: Influence of the number of attention heads on the loss, determined on the validation set.

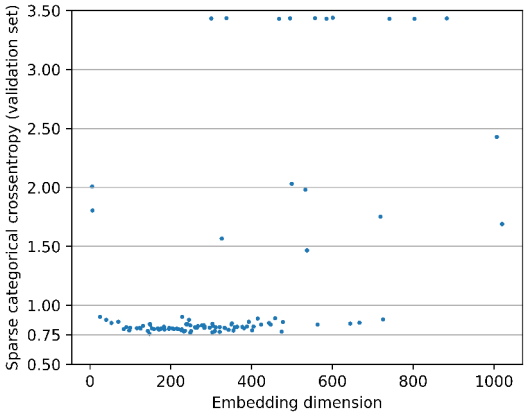


Figure 33: Relation between the embedding dimension and the loss on the validation set.

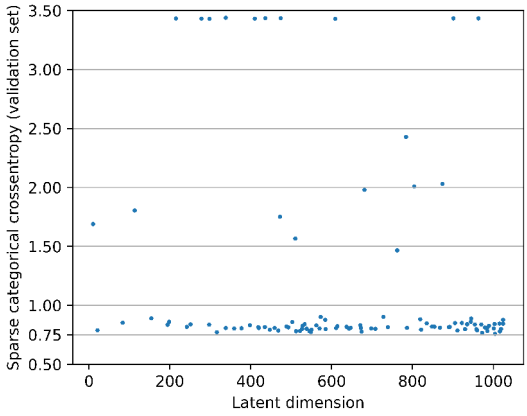


Figure 34: Relation between the latent dimension and the loss on the validation set.

We notice that the optimal embedding dimension is similar to the size of the character vocabulary (92 without start and stop characters) while the optimal latent dimension is close to the maximum of its range

in the hyperparameter search, indicating that better results could be derived from an even more extensive hyperparameter search. Finally, we find that the optimized architecture consists of 2 layers of stacked encoder and decoder blocks, an embedding dimension of ≈ 150, a latent dimension of ≈ 1000, and 7 attention heads. This results in a total model size of 5.1M parameters and a Sparse Categorical Crossentropy loss of 0.758. We further evaluate the performance using two additional metrics:

- **Levenshtein ratio**: the Levenshtein ratio between two words s_1 and s_2 is defined as

$$1 - \frac{\#\text{insertions} + \#\text{deletions}}{\text{len}(s_1) + \text{len}(s_2)}. \tag{2}$$

  This ratio is a character-based measure of similarity between two words, normalized for the lengths of the words (unlike the typical Levenshtein distance). Two identical words have a Levenshtein ratio of 1, the minimum ratio is 0. These ratios are computed using the *Levenshtein* library in Python.

- **CharacterF**: *character n-gram F-score* or *ChrF* (Popović, 2015) is the machine translation equivalent of the traditional F-score and is appropriate for tasks that require character-level translation. As it relies on character n-grams, it is more sensitive towards morpho-syntactic phenomena. ChrF is defined as (Popović, 2015)

$$\text{chrF}\beta = (1 + \beta^2)\frac{\text{chrP} \cdot \text{chrR}}{\beta^2 \cdot \text{chrP} + \text{chrR}} \tag{3}$$

  where chrP and chrR stand respectively for the precision and recall of character n-grams. β can be tuned to assign more importance to recall than precision but we set $\beta = 1$. We use 3-grams as these correspond closely to human judgment (Popović, 2015).

We expect both of these metrics to undervalue the performance of our training; the WLD is very rich in diacritics and situations can arise where two diacritics are phonologically close, but are counted as an insertion in the Levenshtein ratio and invalidate an n-gram in the ChrF metric due to being two different characters. To mitigate this undervaluation, we will also report the Levenshtein ratio and ChrF value for both words after their diacritics are removed. This is done using Python's *Unidecode* (Unidecode) library which transliterates unicode into ASCII characters and outperforms traditional preprocessing methods such as simply stripping accents. In the table below (Table 7) we will show how these metrics behave for the normalization dataset:

Unnormalized - Normalized	ChrF	Levenshtein ratio	ChrF (no diacritics)	Levenshtein ratio (no diacritics)
vief - viêf	0	0.75	1	0.75
kroedwès - krutweš	0	0.4	0.18	0.53
waere - wĕre	0	0.8	0.4	0.8
aafdoeë - āfdūə	0	0.33	0.25	0.33
schoppen - sxopə	0	0.46	0	0.46

Table 7: Sample of the normalization dataset and their evaluation according to the ChrF and Levenshtein metric (both with and without diacritics).

We notice that ChrF is generally too strict and in particular for the first entry (the normalization consists of only an additional diacritic). The non-diacritic ChrF metric is too tolerant for this example but corresponds more closely to our judgment for the other entries. The Levenshtein ratio seems to be even more tolerant than the non-diacritical ChrF and the non-diacritical Levenshtein ratio seems to generally be the most tolerant metric. We will consider ChrF to be the extreme lower bound and the non-diacritical Levenshtein ratio as the extreme upper bound, with the non-diacritical ChrF functioning as the main human judgment metric for the normalization task.

To establish a notion of how poorly normalized the unnormalized words are, we compute these four metrics for the entire normalization dataset (first row in Table 8 below). Additionally, we will establish an upper bound

for these metrics based on the inherent spelling variation of the dataset: even though the unnormalized-normalized pairs are chosen within a small geographic radius, there is a priori no notion of how much the spelling of the normalized words can vary beyond phonological variation. We will therefore compute these metrics for all normalized words within a 6 km radius of each paired unnormalized word (second row in Table 8 below):

	ChrF	ChrF n.d.	Lev.	Lev. n.d.
Unnormalized-normalized	0.112	0.242	0.599	0.710
Normalized variation	0.440	0.589	0.751	0.84

Table 8: Expected upper and lower boundaries for the evaluation metrics.

We expect any functional normalization scheme to at least match the unnormalized-normalized metrics and approach the normalized variation as closely as possible.

6.1.2 GET performance compared to the normal transformer

We will now train both the normal transformer architecture and the GET architecture using the set of parameters found in the hyperparameter search to verify whether the GET architecture brings any improvements to the normal transformer architecture. In an industry context, a separate hyperparameter search for the GET would likely yield better results for the GET architecture, but equal model size is necessary to study whether a statistically significant improvement can be attributed to embedding additional linguistic information, as is the case in GET. We visualize the results in Figure 35:

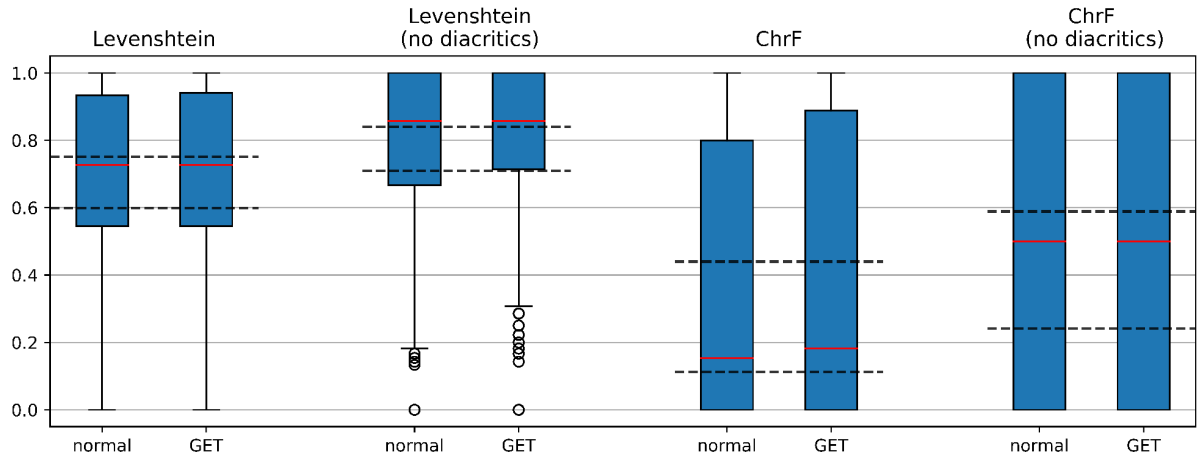


Figure 35: Boxplots of different metrics on the test set, the expected lower and upper bounds are represented by dotted lines. Left and right boxplots represent the normal (Vaswani et al., 2017) and GET architecture respectively.

In general we notice that the GET architecture seems to perform slightly better than the normal transformer architecture. Both architectures' median results approach the expected upper boundary from the normalized variation in the case of the Levenshtein distance, with and without diacritics. In the case of ChrF, we notice a larger variance in the results, with the median of the ChrF with diacritics only slightly above the expected lower boundary. Especially the ChrF results call for a hypothesis test to verify any improvements made by the GET architecture: we again first consider a two-sided Wilcoxon hypothesis test to verify whether the distances for each metric and both architectures are equal, and if rejected, test whether the normal architecture yields better results than the GET architecture using a one-sided test. We summarize the p-values as well as the mean values of each metric below in Table 9 (rounded to 3 decimals):

	ChrF	ChrF n.d.	Lev.	Lev. n.d.
Normal transformer	0.353	0.506	0.713	0.817
GET	**0.363**	**0.516**	**0.718**	**0.821**
p-value test 1	< 0.001	< 0.001	< 0.001	< 0.001
p-value test 2	< 0.001	< 0.001	< 0.001	< 0.001

Table 9: Results of the normal transformer and GET on the normalization task.

From the mean values and hypothesis tests we can conclude that the GET architecture outperforms the normal transformer in the normalization task on Limburgish words in the WLD. The mean Levenshtein values, both with diacritics and without, approach the expected upper boundary (indicating the inherent chaos in spelling variation). The ChrF values, both with and without diacritics, remain slightly removed from the expected upper boundaries of 0.440 and 0.589, respectively, indicating that further improvements could be made to the normalization scheme. Since we found in the manual analysis of the normalization examples that the non-diacritical ChrF value corresponds most closely to human judgment and we attain a mean value of 0.363, we can conclude that this normalization scheme is reasonably effective at normalizing Limburgish words into phonetic notation. Even though we have chosen the same parameters for the GET architecture as was found during the hyperparameter search of the normal transformer architecture, this GET model uses a total of 5.2M parameters due to the extra 2 dimensions caused by the coordinate embedding. This increased model size does not skew the results in favor of the GET as these extra dimensions only allow for a heterogeneous interaction between the coordinates and the embedded characters of the dialect words in the attention mechanism, and do not allow for any larger inference of information further in the network. Likewise, larger network sizes for the normal transformer do not result in an increase in performance, as the hyperparameter search has shown.

To test the influence of the geographic radius that was used to determine entries featuring normalized variants of an unnormalized word, we have also generated a dataset with radius 6 km instead of 0.5 km. The results of training the GET architecture on this dataset are shown in Table 10 below:

	ChrF	ChrF n.d.	Lev.	Lev. n.d.
GET (6 km dataset)	0.356	0.505	0.713	0.815

Table 10: Results of the GET on the normalization dataset with a larger radius (6 km).

We do not notice an improvement in mean metric values. Even though this dataset is almost 3 times larger than the dataset constructed using a 0.5 km radius, it is possible that the larger radius allows for actual phonological variation to enter the data, instead of just spelling variation.

Further improvements could be made to this normalization scheme by either increasing the model size and repeating the hyperparameter search for the GET architecture or manually curating a test set to accurately measure the performance of the normalization scheme. Furthermore, the curation of a large golden standard paired dataset of normalized and unnormalized words/sentences could allow for a more extensive study of normalization tasks on non-standardized languages, but unfortunately due to time and resource constraints and the very low resource nature of Limburgish, these were outside of the scope of this thesis.

6.1.3 Geographic analysis and some examples

We now visualize the non-diacritical ChrF values and the coordinates of their corresponding unnormalized words in Figure 36. We do not notice any visual relation between the performance of the normalization scheme and the location of the unnormalized-normalized word pairs. Finally, we will present 30 randomly chosen unnormalized words and their respective normalized predictions by the GET in Table below.

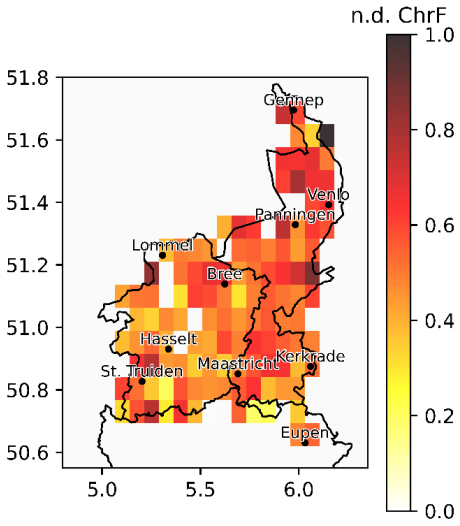


Figure 36: 2D histogram of the non-diacritical ChrF values plotted on the coordinates of their corresponding unnormalized words (test set).

	Unnormalized word	Prediction	Target	non diacr. ChrF	Translation
1	sjnaps	snaps	snàps	1.0	schnaps (drink)
2	zeik	zę̨i̯.k	zę̨i̯.k	1.0	fecal sludge
3	daavekot	dāvəkǫt	dāvəkòt	1.0	dovecote
4	balkebrie	balkəbri	balkəbrī	1.0	traditional meat dish
5	sjollek	šoləkˌ	šo̯ləkˌ	1.0	type of apron
6	steik	stekˌ	stēkˌ	1.0	jolt of pain
7	volle	vǫlə	vǫl	0.667	full
8	strooie	stroi̯ə	strōən	0.5	straw (material)
9	vaon	vāōn	vōn̨	0	flag
10	meute	møt̄	moôte	0	effort
11	kool	kiəl	kīəl	1.0	cabbage
12	hèndichə	hendix	hendixe	0.889	handy
13	kwartsche	kwartse	kwę̄rtšə	0.2	quarter
14	hiemël	hi:məl	hi:məl	1.0	heaven
15	krappele	krapələ	krápələ	1.0	scratching (meat)
16	lintteeke	lintēkə	lentēkə	0.6	scar
17	beikem	bę̨i̯kəm	bę̨kəm	0.286	buckling (fish)
18	sjei	šę̨i̯ˌ	šę̨i̯ˌ	1.0	vagina (horse)
19	tijlxat	tę̨jlxāt	tilxat	0.444	entrance hole (beehive)
20	tweede	twēdə	de	0	second
21	boeteram	būtəram	bōēteram	0.182	sandwich
22	kievvel	kivvəl	kivəl	0.571	hoop (toy)
23	ekeboem	ekbum	ēkəbum	0.286	oak tree
24	preuvə	prèùve	prēūvə	0.75	to taste
25	bingel	bindel	bę̨ŋəl	0	clapper (bell)
26	moel	mul	mul	1.0	mouth (derogatory)
27	duuzelix	duzelix	duzelix	1.0	dizzy
28	riep	rip	rip	1.0	overripe
29	krolle	kro̯lə	krolə	1.0	curly hair
30	áfzéttə	ifzetə	afzɛtə	0.75	to rip off/defraud

Table 11: Examples of unnormalized words and their predictions according to the GET.

We note the following behavior:

- In entries 3, 4, 5, 14, 15, 16, 18, 19, 21, 22, 26, 27, 28, 29 the words are successfully normalized into phonetic notation. E.g. the long `aa` in entry 3 becomes `ā`, the `ie` becomes `i` in entry 4, the `sj` becomes `š` in entry 5. Sometimes different diacritics are predicted: `o̧` instead of `ò` in entry 3. This is likely because `ò` is not a feature of the phonetic notation of the WLD (Appendix C) and the predicted diacritic matches the expected sound of the region. Likewise, in entry 21 it avoids predicting the non-existing phonetic notation `ōē`. The GET does make errors, it predicts `ęj` in entry 19 which is a conventional notation and not phonetic.
- In entries 2, 6, and 11 the predictions closely correspond to the target normalized words, but clearly change the sounds of the unnormalized words. In entry 2, the diphthong `ęi̯` is predicted, which is correct (see Appendix C) and in entry 6, the diphthong `ei` is changed to an `e`, which is likely correct since the locality (Loksbergen) lies above the `e:`→`i` isogloss (Leenen et al., 1947). We visualize all sounds for the keyword "steek" based on the entries of the WLD in Figure 37 We notice that variants of `ei` occur irregularly throughout the Limburgish area, but particularly in the West-Limburgish transitional area. Since other localities near Loksbergen use both phonemes, it is unclear whether the choice the GET architecture makes is accurate. In entry 11, the prediction converts the Dutch phoneme `o` correctly into a `i`-variant. This is particularly interesting as the locality (Gerdingen) is situated within the delabialization area within Limburgish (Bakkes et al., 2007) (i.e. variants outside this zone use the phoneme `ø` or `u`).

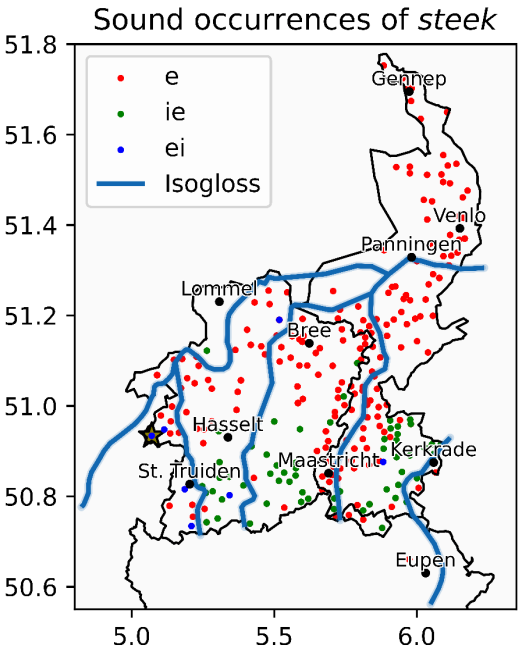


Figure 37: Occurrences of sounds for the keyword "steek" in the entire WLD, diacritics are removed for simplicity. Loksbergen is marked with a star.

- In other instances, such as entries 7, 8, 17, 20 the model correctly predicts the normalization while the target is incorrect, therefore lowering the overall ChrF value.
- In entries 9, 10, 12, 13, 23, 30 the model predicts partially correct results but does predict incorrectly normalized characters. In entry 9, conventional spelling is predicted while in entries 10 and 12 the prediction is more accurate than the original target, but an end-schwa is omitted. Other characters such as the `i` instead of `a` in entry 30 are completely incorrect.
- In entry 24, the end-schwa is incorrectly normalized and conventional spelling is used, making this prediction inaccurate. Entry 25 seemingly tries to correct a spelling mistake but in doing so fails to correctly predict the phonetic normalization, this is likely due to the ambiguity of the input word.
- In entry 1, the `š` is normalized to a `s`, even though this is a direct High German loanword. This normalization reflects the rule of the `s` *to* `š` Panninger isogloss within Limburgish.

In conclusion, we find from this manual analysis that the GET model is overall successful in normalizing words from various Limburgish spelling conventions to IPA notation. The model errs occasionally and in those instances uses conventional spelling or predicts partially correct results. In some ambiguous situations, the model performs better than the target normalization. Only in a few instances does the model make a critical error such as omitting a character or substituting a character for a sound that is far removed from the correct sound, which means that this normalization scheme is a reliable method for improving the normalization of a dataset.

6.2 Dialect Neural Machine Translation

In the case of spelling normalization, the GET architecture uses coordinates as the linguistic embedding to improve the results of the normalization. The coordinate embedding can also be leveraged in other NLP applications such as *Dialect Neural Machine Translation* or *DNMT*, where the GET architecture is trained on pairs of words in the WLD, together with their coordinates.

While the first generation of NMT focused on single language pairs, research quickly shifted to multilingual NMT (Dabre et al., 2020) as learning multiple languages allows the model to better generalize due to a phenomenon called *knowledge transfer* (Dabre et al., 2020). However, most cases of multilingual NMT use parallel corpora and if not available employ methods such as *Zero-Resouce Translation* where the low resource language is paired with a high resource language either through transfer learning, pivot learning (the high resource language is used as an intermediate in a low resource-low resource pair) or zero-shot translation (no parallel data is available). For a recent survey, see (Dabre et al., 2020). Naive multilingual NMT scale quadratically in the number of parameters per included language (Arivazhagan and et al., 2019) and therefore other approaches have been proposed such as using a single unified encoder and decoder for each language pair task (Ha et al., 2016), reducing the model size to one comparable to that of a single language pair NMT model. In this approach, a language token is appended in front of all words in a single language pair context. An alternative approach using such language tokens is (Johnson and et al., 2017), where a language token is added at the start of each sentence in an NMT task on 12 language pairs.

These approaches differ from the GET architecture since they either treat the language-labelled words as separate words or treat the language token as a single word in a sentence. The GET architecture instead allows for continuous variables to be embedded directly in the attention mechanism of the NMT model without increasing the embedding space and does not limit the NMT model to discrete language pairs, as is the case in for example a language continuum such as Limburgish. Here, we will study the case of word-to-word NMT (i.e. character-based) on pairs of words that have a matching keyword, as these words are cognates under phonological variation. The reasons for doing so are twofold:

- No parallel corpus of the Limburgish dialects exists and Limburgish itself can generally be considered a low resource language, as even monolingual corpus data is scarce (See Chapter 1).
- Most of the variation within Limburgish is phonological and to lesser extents lexical and grammatical, as they are closely related varieties (see the first chapter for a more detailed discussion).

Similar studies such as (Honnet et al., 2018) (Scherrer and Ljubeić, 2016) have previously applied character-based NMT on Swiss German, which in the former study is leveraged into full NMT using phrase-based statistical machine translation methods. Due to time constraints, we will only study word-to-word NMT on the Limburgish dialects, which could be leveraged into full Limburgish dialect machine translation in future research.

6.2.1 Model training

Similar to the case of spelling normalization, there is no need to undersample the WLD and we will again use the normalized subset of the WLD after all preprocessing steps without the undersampling, yielding a total of 805 208 entries. We will generate a new dataset consisting of pairs of dialect words and their respective coordinates. Since we only consider the task of modeling the phonological variation of words in Limburgish, we will randomly select pairs with a common Dutchified keyword, ensuring that they are cognates. Since the number of generated pairs grows quadratically with the proportion of the WLD that corresponds to that keyword, some keywords that are anomalously common in the WLD are highly overrepresented with this method. We therefore remove the 20 keywords with the largest corresponding entries, since these are mostly superfluous adverbs or prepositions such as `de` (the), `op` (on), `van` (of), `in` (in), etc. Secondly, to prevent frequent target words from skewing the learning, we will resample the entries corresponding to each keyword according to the same method as in section 4.3, this time while maintaining the total size of the keyword subset. As an example of such a resampling, we show the entries corresponding to the keyword “koe” (cow) before resampling (Figure 38) and after resampling (Figure 39):

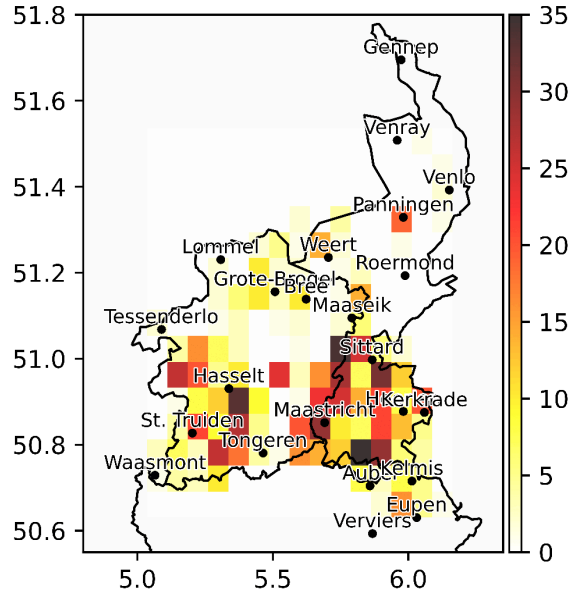


Figure 38: Entries ("koe") before resampling.

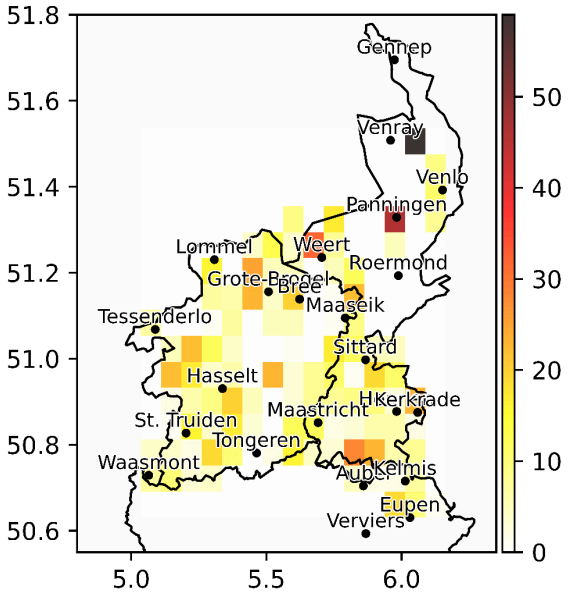


Figure 39: Entries ("koe") after resampling.

Certain words are particularly unique vocabulary and are omitted from the new dataset if their corresponding keyword has fewer than 10 total entries. To reduce the total size of the newly generated dataset, a 10% subset is sampled of the subset corresponding to a keyword and all pairs (excluding pairs from identical coordinates) are generated. We again only consider words with fewer than 10 characters and find a new dialect translation dataset of 20.2M entries. We finally split this dataset into a $80 - 10 - 10$ train, validation, and test set.

Due to resource constraints, we cannot perform an extensive hyperparameter search to optimize the network. Instead, we have manually varied the parameters and compared the choice of parameters to the literature (see above). We decide on an embedding dimension of 256, a latent dimension of 1024, 8 attention heads, and no stacked encoder or decoder blocks. This results in a total of 7.6M parameters. The model is again trained using the Adam optimizer with a Sparse Categorical Crossentropy loss until convergence, i.e. 5 epochs without an improvement of 10^{-3} of the validation loss. Since we do not have access to a curated dialect translation dataset, we will again use the Levenshtein and ChrF metric (with and without diacritics) as evaluation metrics and compare to the expected upper and lower boundaries from the normalization task of Table 8 in section 6.1.1. We visualize the results in Table 12 and Figure 40:

ChrF	ChrF n.d.	Lev.	Lev. n.d.
0.407	0.485	0.687	0.736

Table 12: Performance of GET on the dialect translation task.

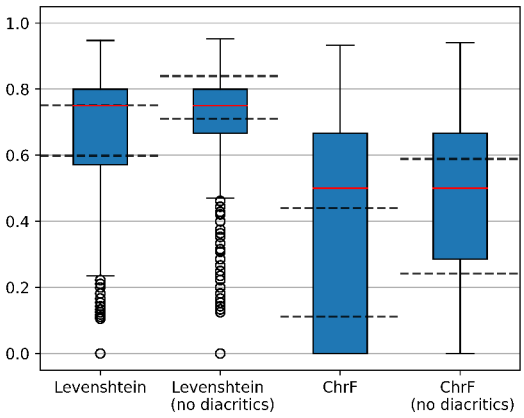


Figure 40: Different metrics on the test set, the expected bounds are represented by dotted lines.

We notice that the mean metrics approach the expected upper boundaries, although noticeably less than was the case in the normalization task. From the boxplots we can deduce that a large variance in the results exists, which might be caused by low-quality entries in the generated dialect translation dataset, as was the case for some examples in the generated normalization dataset.

6.2.2 Geographic analysis

We visualize the non-diacritical ChrF metric for the origin coordinates (Figure 41) and the target coordinates (Figure 42):

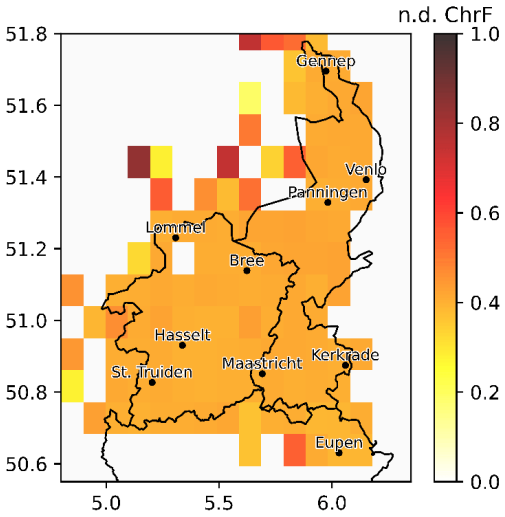


Figure 41: 2D histogram of the non-diacritical ChrF metric on the test set, plotted per coordinate of origin in the translation task.

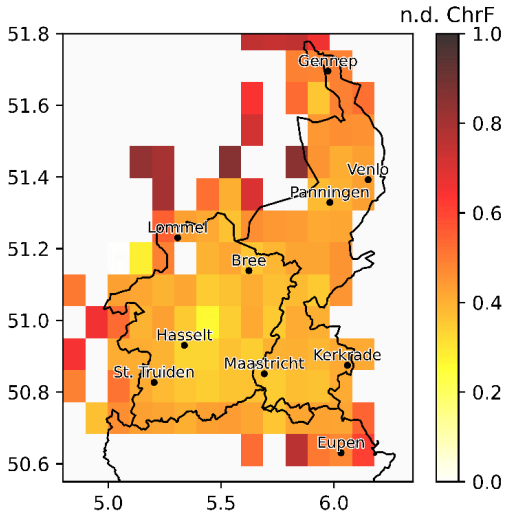


Figure 42: 2D histogram of the non-diacritical ChrF metric on the test set, plotted per coordinate of target in the translation task.

We clearly notice that translating both from and to the area outside the region that was selected by the editors of the WLD (see Figure 6) results in a better non-diacritical ChrF metric. Data outside this region is very infrequent, which likely allows the model to overfit. We also notice that the performance of the model is geographically very homogeneous for the origin coordinates in the Limburgish area when compared to the target coordinates. Interestingly, the metric on the target coordinates seems to behave slightly worse in

the geographic center while the boundaries are more pronounced. It is possible that centrally located data suffers from more ambiguity caused by the surrounding localities than more isolated localities. To investigate whether there are any regional patterns in performance, as we did in 5.4, we will isolate specific rectangular regions in either the origin (Figure 43, Figure 45) or the target coordinate (Figure 44, Figure 46) maps and investigate which regions yield higher or lower non-diacritical ChrF values.

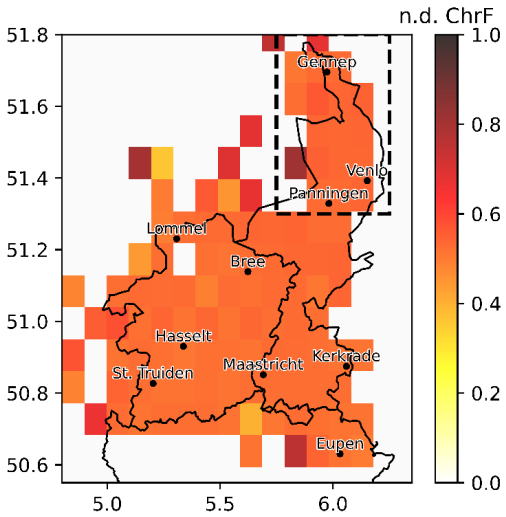


Figure 43: 2D histogram of the non-diacritical ChrF metric on the test set, confined to the target coordinates in the rectangular region, plotted per coordinate of origin in the translation task.

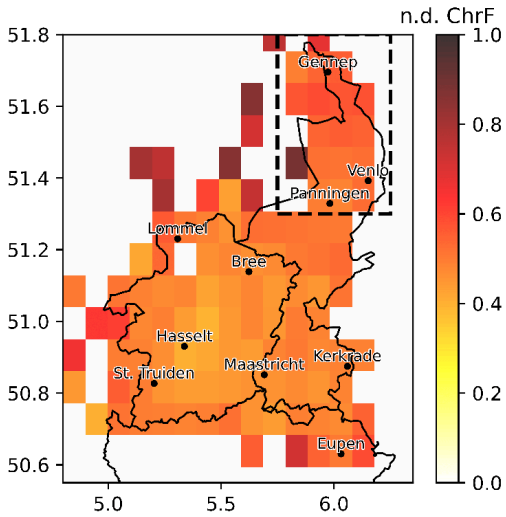


Figure 44: 2D histogram of the non-diacritical ChrF metric on the test set, confined to the origin coordinates in the rectangular region, plotted per coordinate of target in the translation task.

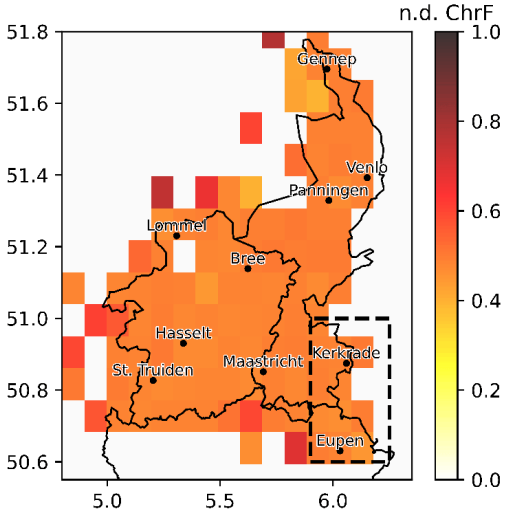


Figure 45: 2D histogram of the non-diacritical ChrF metric on the test set, confined to the target coordinates in the rectangular region, plotted per coordinate of origin in the translation task.

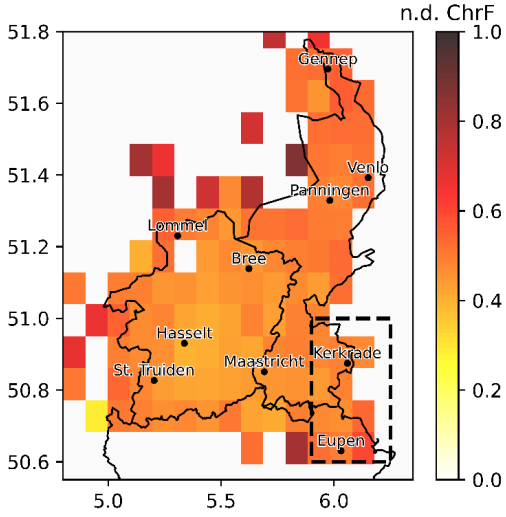


Figure 46: 2D histogram of the non-diacritical ChrF metric on the test set, confined to the origin coordinates in the rectangular region, plotted per coordinate of target in the translation task.

As we confine either the origin or target coordinates to a narrower region, the non-diacritical ChrF score increases for the entire map as the rectangle is located less centrally. Even if we isolate the Kleverlandic and

Ripuarian regions, which are not traditionally considered to be Limburgish and are generally phonologically further removed from the rest of the studied region, we cannot find any regional patterns. Other choices of regions (not shown here) do not reveal any regional patterns either, as was the case in dialect identification.

From this we can conclude that the GET model and more generally the dialect translation task succeeds at overcoming any linguistic discontinuities/transitions, although infrequent data likely gets overfitted and centrally located data seems to suffer from a decrease in performance.

6.2.3 Some examples

Below in Table 13 we show some examples of dialect translation on individual words:

	Input word	Prediction	Target	non diacr. ChrF	Translation	Localities
1	špat	spat	spat	1.0	osteoarthritis (horse)	Q252p-L364p
2	briə.kə	brē̤.kə	brę̄.kə	1.0	to spread manure	Q157p-Q182p
3	moder	mojər	mojər	1.0	mother	L317p-Q177p
4	seŋəl	seŋəl	seŋəl	1.0	girth (horse)	L163a-L270p
5	wēi̯	wę̄i̯	wę̄i̯	1.0	whey	Q030p-Q255p
6	rempəl	rimpels	rumpels	0.6	wrinkles	L245p-L269p
7	sop	sop	sop	1.0	soup	L299p-L215p
8	xeld	xē̤lt	xēld	0.5	money	Q203p-Q199p
9	boṭərham	botəram	botəram	1.0	sandwich	L282p-L215p
10	werk	werk	werk	1.0	work	Q020p-Q072p
11	dōōn	dōēn	af̬dūn	0	to do	L328p-L217p
12	bęsəl	bęsəl	bę.səl	0.286	bushel (hay)	Q072p-Q082p
13	bø̄səl	bø̜səl	bøsəl	1.0	bushel (hay)	Q284p-L298p
14	vęlt	vɛlt	vɛəlt	0	field	L421p-L248p
15	hôre	hore	hore	1.0	male pigeon	P171p-L209p
16	sxøp	sxøp	sxøp	1.0	grain shovel	K357p-P175p
17	weχ	wex	wex	1.0	road	P171p-L210p
18	hǭtə	hǫwtə	hǫwtən	0.857	wooden	Q003p-L368p
19	werk	werk	werk	1.0	work	K353p-K353p
20	kát	kat	kat	1.0	cat	L354p-L215p
21	knøpəl	knø̄pəl	knøpəl	1.0	club	Q253p-Q253p
22	bǫtər	botər	botər	1.0	butter	P186p-Q029p
23	slęi̯p	slęĭ̯.p	slęî̯.p	1.0	field drag	L321p-L317p
24	kamp	kāmp	käm	0.667	teasel (plant)	Q204a-L313p
25	wilde	wel	wøl	0	wild	L371p-P051p
26	nak	nek	nek	1.0	neck	Q082p-L246p
27	hūs	hōēs	hôêês	0.4	house	L366p-L248p
28	kap	kap	kap	1.0	cap (clothing)	Q253p-Q113p
29	mɛrt	mert	mert	1.0	market	L360p-L383p
30	distele	destələ	destəl	0.889	thistle	Q119p-P048p

Table 13: Examples of input words and translation predictions by GET.

We note the following behavior:

- In all entries except 6, 24, 25, and 27 the translation either matches the target or performs better than the target. In entry 6, the word is pluralized even though the input word is not, and the sound `i` instead of `u` is incorrectly predicted since the locality (Blerick) uses the `u` (see Figure 47). We find that the incorrect pluralization results from a typographical error in the WLD, the keyword "wimpers" only appears in plural, and `rempəl` is supposed to be entered as `rempəls` in the WLD. Since this is a single occurrence, the model assumes that the omission of an `s` is a local variation, as it functions as a phonological translation model and has no information about semantics or grammar. In entry 24, the model correctly predicts a `p` even though the target omits it as it is likely an anomaly (see Figure 48). In entry 25, the model predicts the sound `e` instead of `ø` for Lummen, which is correct as the latter sound is likely a result of poor normalization (see Figure 49). Finally in entry 27, the model predicts the sound `ōē` instead of `ôêê`, where neither are phonetic conventions of the WLD but the first convention is used much more frequently in other entries in the WLD.

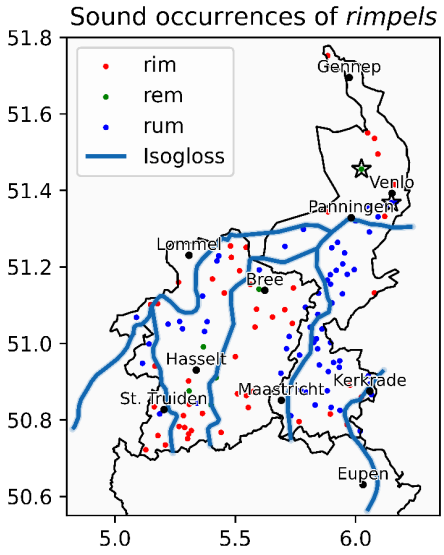


Figure 47: Occurrences of sounds for the keyword "rimpels" in the WLD, diacritics are removed for simplicity. Meterik and Blerick are marked with stars.

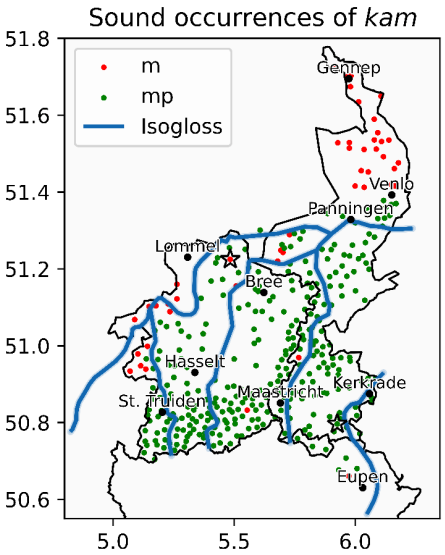


Figure 48: Occurrences of sounds for the keyword " kam" in the WLD, diacritics are removed for simplicity. Mechelen (NL) and Sint-Huibrechts-Lille are marked with stars.

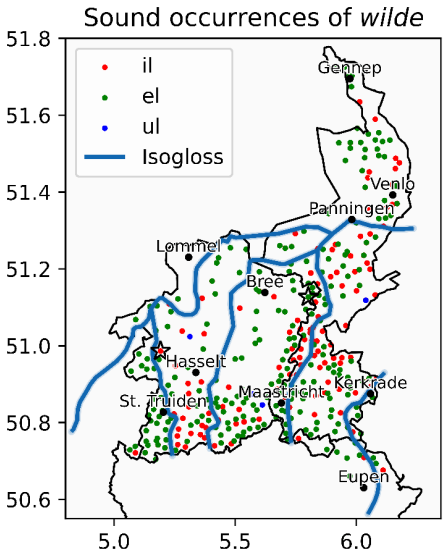


Figure 49: Occurrences of sounds for the keyword "wilde" in the WLD, diacritics are removed for simplicity. Ophoven and Lummen are marked with stars.

- In entries 8, 11, 18, 24, and 30 the model not only translates correctly, but performs better than the target word in terms of normalization. In entry 8, the correct phonetic notation `t` instead of `d` is used, while in entry 11 the target misses the entire preposition of the verb. Likewise in entry 30, the

end-schwa is not omitted which is the case for the target word.

- Entries 1, 2, and 3 are particularly interesting as the model seems to have learned the correct phonological variation associated with the traditional Limburgish isoglosses. In entry 1, the `š` is translated to `s` as the origin is Moresnet and the target Meeuwen, meaning that we have crossed the Panninger isogloss. In entry 2, the origin is Jesseren and the target Nerem, which are only $\approx$ 10 km apart but lie on opposite sides of an isogloss (see Figure 50). In entry 3 the model again correctly captures the phonological variation from `d` to `j` (see Figure 51).

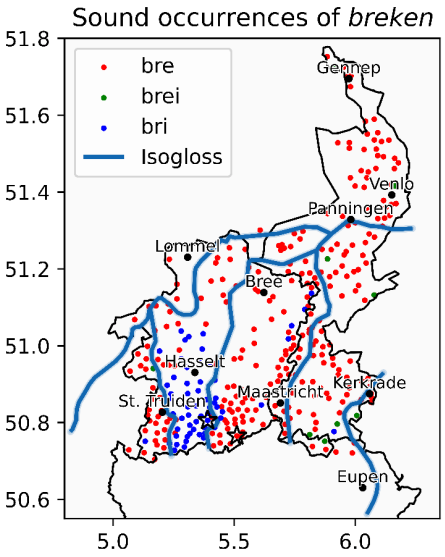


Figure 50: Occurrences of sounds for the keyword "breken" in the entire WLD, diacritics are removed for simplicity. Jesseren and Nerem are marked with stars.

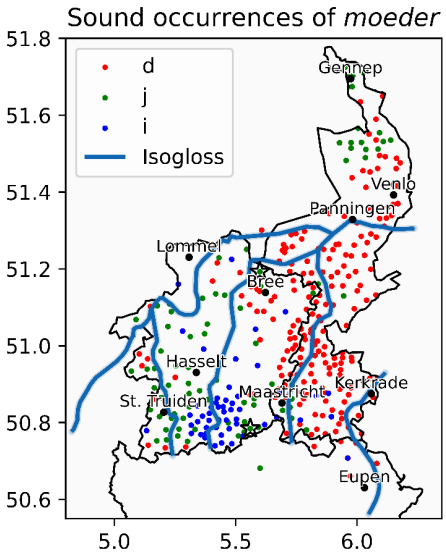


Figure 51: Occurrences of sounds for the keyword "moeder" in the entire WLD, diacritics are removed for simplicity. Bocholt and Millen are marked with stars.

In conclusion, the GET architecture is successful in the task of dialect translation: it takes into account isoglosses and in addition sometimes normalizes the translated words. In a few instances it fails, likely because the dataset it was trained on is not sufficiently normalized. As is the case in the normalization task, a manually curated dialect translation dataset could increase the performance of the dialect translation. Due to resource constraints, larger networks could not be trained and neither could we perform a hyperparameter search, which would likely yield even better results.

This translation model only translates phonological variation, but could be bootstrapped in combination with a word-based translation model to a full dialect translation pipeline, as has been done before for Swiss German (Honnet et al., 2018). This model currently does not take into account Part-of-Speech, grammar, or semantics (as an example we saw the wrong pluralization of a word) and could therefore benefit from a word-to-word translation model.

6.2.4 Variation map generation

We have seen in the section above that the model succeeds in learning some phonological variation rules in Limburgish. This translation task can be leveraged to generate familiar maps from dialectology: by fixing an existing dialect input word (and corresponding coordinates), we can vary the target coordinates over the Limburgish area and visualize the translated words to generate all phonological variation of a certain word. We will only consider the coordinates within a 20 km radius of the border of both Limburgs and Liège. Here

we compare such maps with traditional dialectology:

Ripuarian variation:
A typical isophone attributed to the Benrather line is the x → j such as in xo:t →jo:t (good). We fix the input word xōt, derived from the locality Maastricht and find the variation map (Figure 52) .We notice that the variation map indeed predicts the x → j isophone, although not matching exactly with the Benrather line. It also predicts the transition of variants of o:→u in the west, although incorrectly in the north of Belgian Limburg. The variation map generates a large number of differently spelled phonemes, especially outside of the area with frequent data in the WLD. To compare with the entries in the normalized WLD, we find Figure 53. We notice that despite the very large variation of used spelling conventions, the generated variation map approximates the phonological variation reasonably well.

Central-Limburgish
The Central-Limburgish area is demarcated by the Panninger line and Panninger sideline on the east and west-side respectively. To study whether the GET has implicitly learned these isoglosses, we generate for the Panninger line the variation maps for sp→šp by studying the variation of the keyword "spelen" (to play) and fixing the input word špele from locality Roermond (Figure 54). For the Panninger sideline, we study the isogloss sx→š by generating the variation of the keyword "school" (good-looking) and fixing the input word šo.l from the locality Bree (Figure 58).

For the Panninger line, we notice that the sound s is incorrectly predicted for the entire Limburgish area, apart from a region in the northeast of Liège. This pattern is incorrectly learned by the GET as a result of an error in the normalization procedure: we have removed entries that contain the conventional sj spelling, where we expected phonetic notation š. However, the conventional notation in this instance is much more common than the frequency of the phonetic notation over the entire area, meaning that this normalization filter only allows for variants of the type sp (see Figure (55)), which is not endogenous to regions east of the Panninger line and results in the model incorrectly predicting these phonemes. This is likely caused by the surveys that are the basis of the WLD: the dialects in Liège were curated separately from the ones in both Limburgs, which relied more on Dutch orthography. Apart from this sound, the GET does correctly predict the variation of the first vowel in "spelen": ø north of the Uerdinger line in Kleverlandic while an e in the remainder.

We try another isophone to retrieve the Panninger line in the North of Belgian and Dutch Limburg: nd→nj. We study the variation of "wandelen" (to walk) for the fixed input word wāndələ from locality Genk and find Figure 56. On comparison with the variation of the nd (Figure 57), we find that the predictions accurately follow the real variation. On Figure 57 we have also extracted the predicted isophone, which closely corresponds to the actual isophone. We note that the western boundary of the isophone region does not closely correspond to the Panninger line: this is because the Panninger line is an idealized line of isophones (see Chapter 1 for a more thorough discussion). We again find that a large number of conventional spellings are predicted.

For the Panninger sideline we find that the GET-generated variation map (Figure 58) closely corresponds to the real variation (Figure 59): consider the general sx→š isophone, the use of ou instead of o in the southwest between Tongeren and St. Truiden and the u instead of o in Ripuarian beyond the Benrather line. If we extract the general sx→š and compare it to the idealized Panninger sideline (Figure 59) we find that the predicted isogloss very closely corresponds to the isogloss present in the WLD. We also find that in general the predicted isogloss (Figure 59) closely corresponds to the Panninger sideline, although slightly diverging at some points. We again note that the Panninger sideline is an idealized isogloss, and is based on administrative boundaries and other isophones. We see for example a large divergence between Hasselt and Bree, where the sparsely populated Donderslag moors are. Due to the lack of data in this region, the Panninger sideline was arbitrarily drawn. In the north, near Panningen, we also notice that both the predicted isogloss and the entries in the WLD do not correspond to the idealized Panninger sideline. This is likely due to many of these isoglosses being determined before the collection of the WLD (see in Chapter 2 how the Uerdinger and Panninger sideline in Dutch Limburg were based on SGV data, although data collection for the WLD

continued for decades after).

Brabantian-Limburgish and West-Limburgish
The Brabantian-Limburgish and West-Limburgish dialects are demarcated by the Uerdinger and Tonality lines respectively. A typical phenomenon in both regions is the deletion of r (by this we mean the guttural, but in the WLD notation) in words such as "wortel" (root). To study this isophone, we generate the variation map for the fixed input word wǫ.rtəl from locality Achel (Figure 61). We find that the generated map closely approximates the real variation (Figure 62): we particularly note the deletion of r beyond the Uerdinger and Tonality line, but also in the south. Near Tessenderlo, the glottislag is predicted, which is a phoneme exclusive to this Brabantian-Limburgish region. The isophone ø→o west of the Uerdinger line is also correctly predicted, as is the Ripuarian isophone t→ts near the Benrather line.

Delabialization
An important trait of some Limburgish dialects is "delabialization" ("ontronding"): y→i, ø→ɪ (both in IPA notation). This isophone is interesting as it is not associated with either the Rhinelandic fan, the Cologne Expansion or the Brabantian Expansion but rather with social factors (Bakkes et al., 2007). Delabialization occurs in a long region in the middle of Belgian Limburg, but due to its strongly differentiating effect with Dutch and other varieties of Limburgish the region has been gradually shrinking since the 1940's (Bakkes et al., 2007). Delabialization also a phenomenon in other regions such as Brabantian (e.g. in the Brabantian dialect of Leuven the city name becomes "Leive"), Ripuarian, and Alemannic. To investigate whether the GET model predicts delabialization, we study the isophone by using as fixed input word dīər from locality Grote-Brogel. We find that the generated variation map (Figure 63) closely corresponds to the real variation (Figure 64) for the delabialization region, apart from the localities around Hasselt (Hasselt itself does have delabialization). We also visualize the extraction of the predicted isophone in Figure 64.

Figures

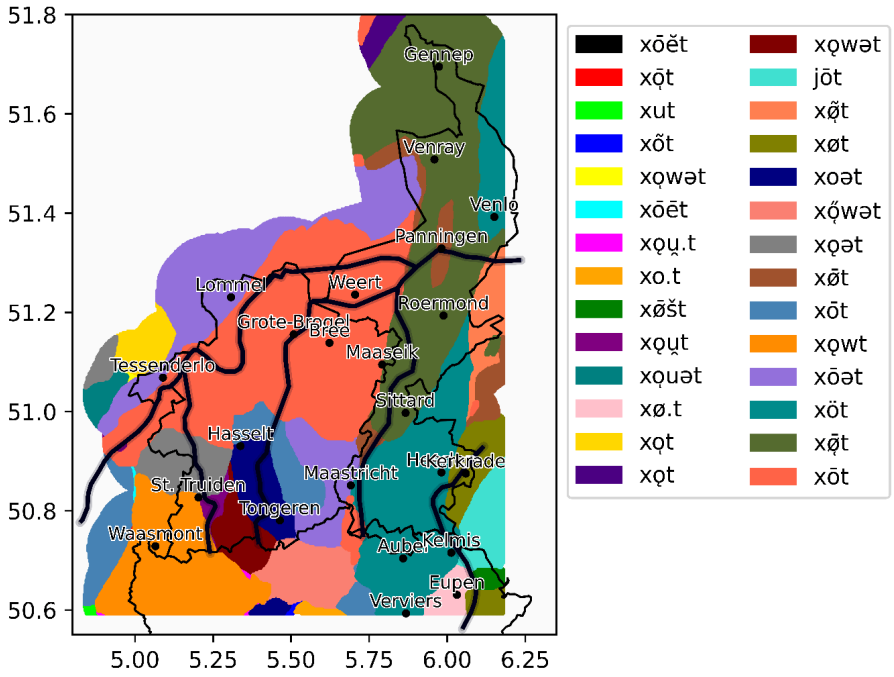


Figure 52: GET-generated variation map of “goed” (good) with Maastricht as fixed input.

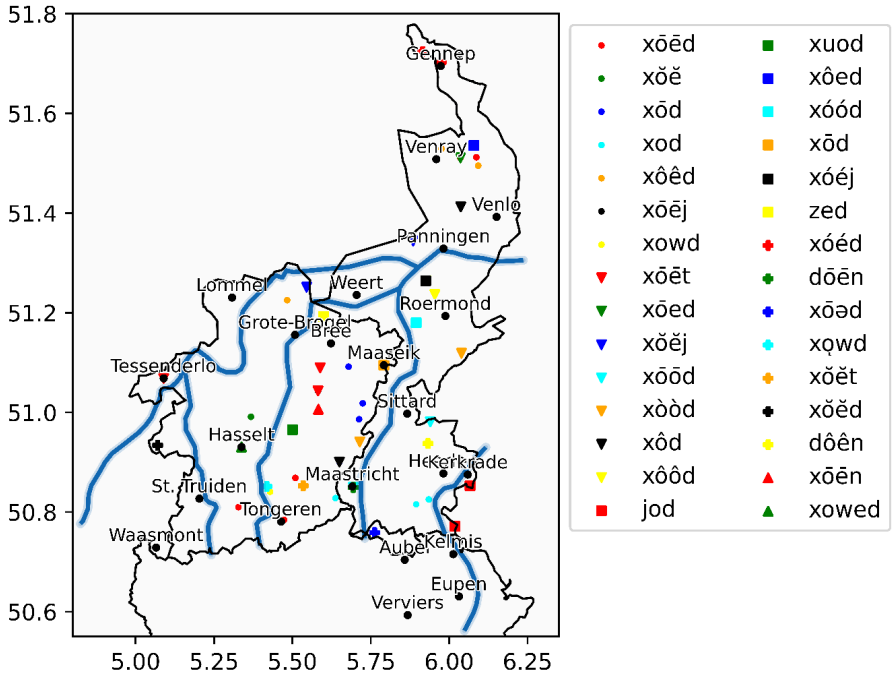


Figure 53: Entries in the WLD for keyword “goed” (good). Only the 30 most common spelling variants are plotted.

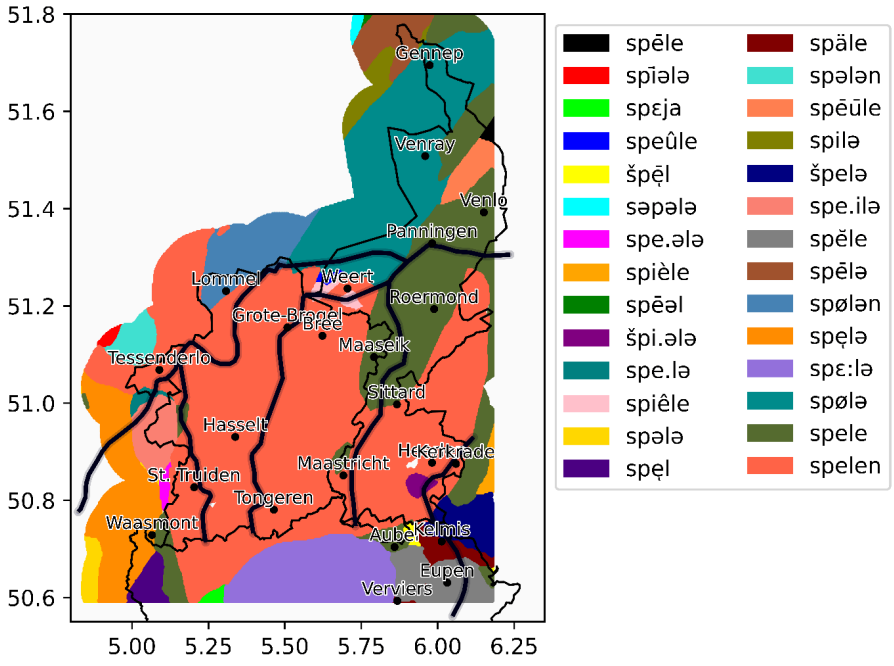


Figure 54: GET-generated variation map of “spelen” (to play) with Roermond as fixed input.

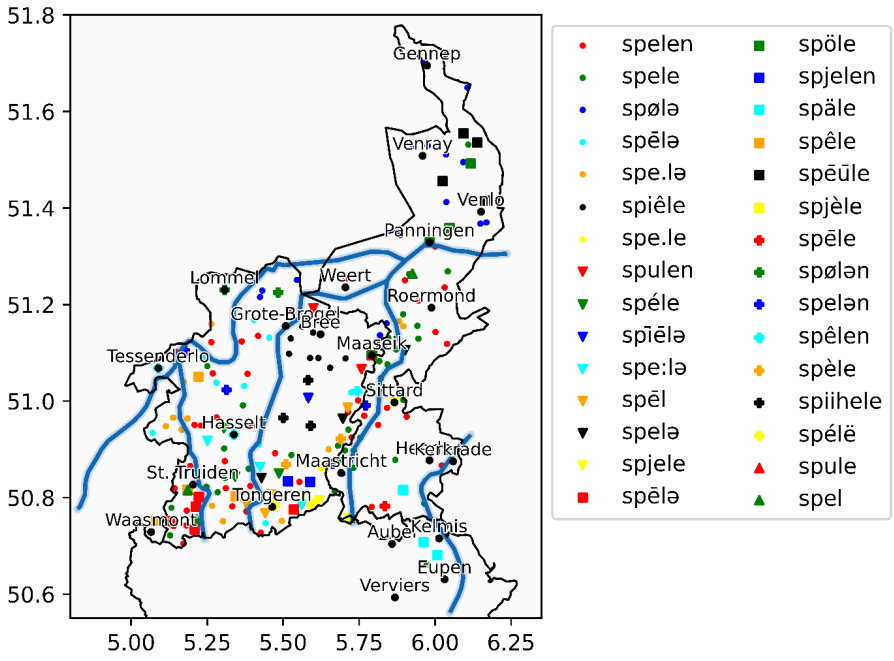


Figure 55: Entries in the WLD for keyword “spelen” (to play). Only the 30 most common spelling variants are plotted.

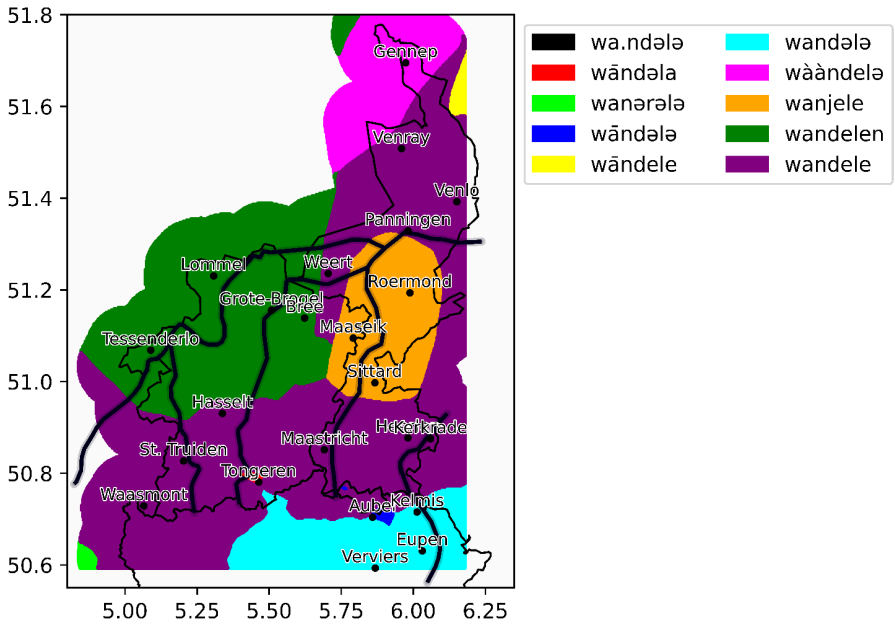


Figure 56: GET-generated variation map of “wandelen” (to walk) with Genk as fixed input.

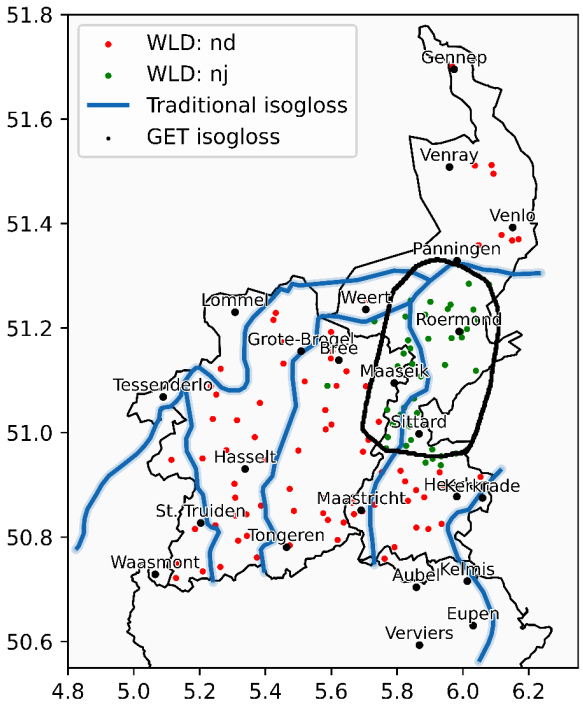


Figure 57: The isophone extracted from the GET prediction, as well as simplified entries in the WLD for keyword “wandelen” (to walk).

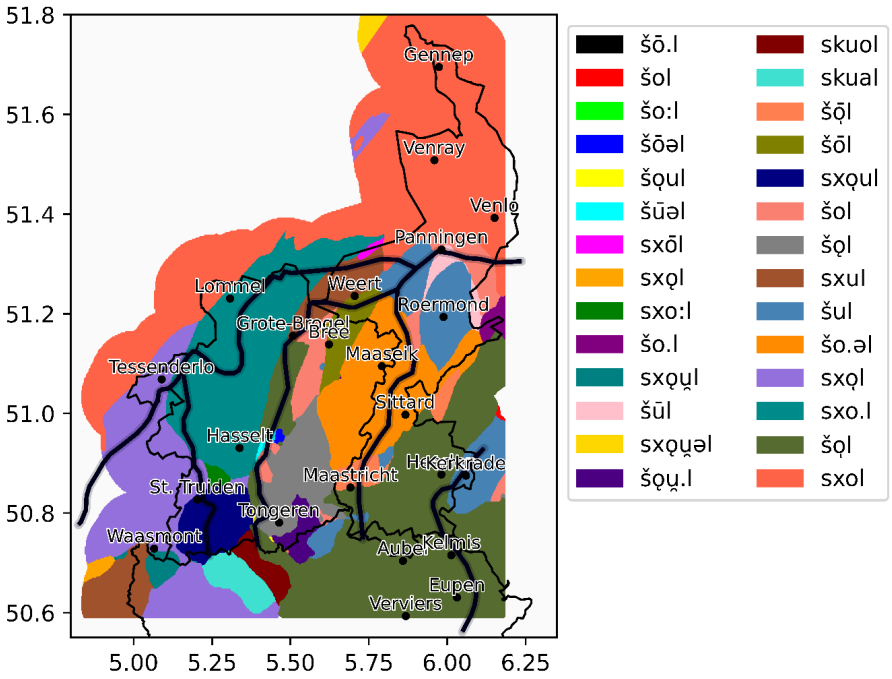


Figure 58: GET-generated variation map of “school” (school) with Bree as fixed input.

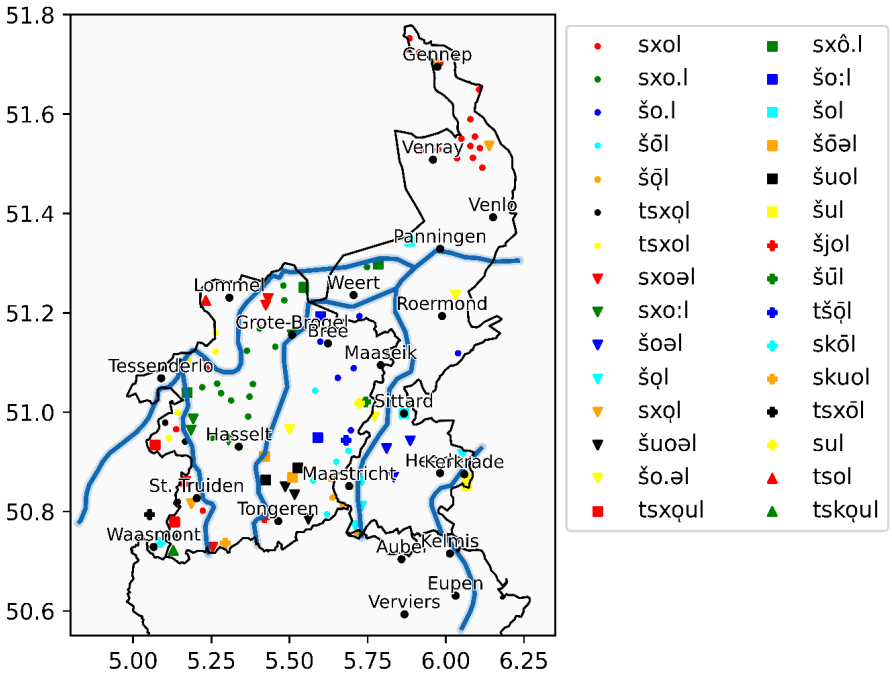


Figure 59: Entries in the WLD for keyword “school” (school). Only the 30 most common spelling variants are plotted.

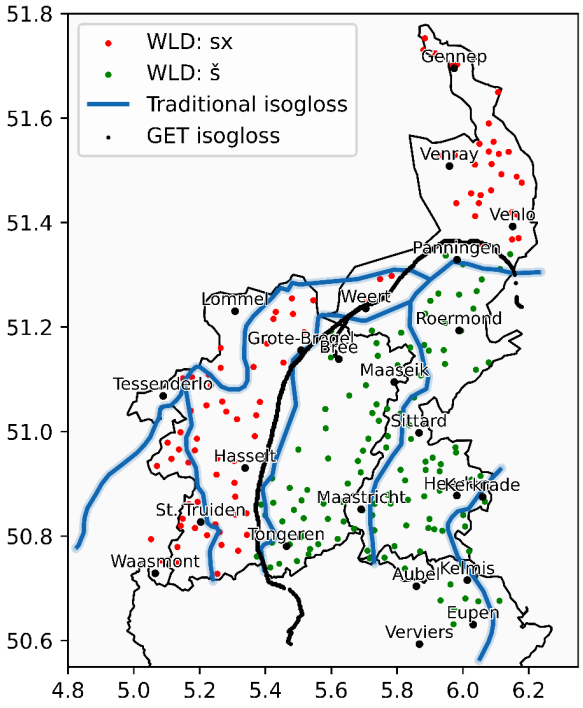


Figure 60: The isophone extracted from the GET prediction, as well as simplified entries in the WLD for keyword “school” (school).

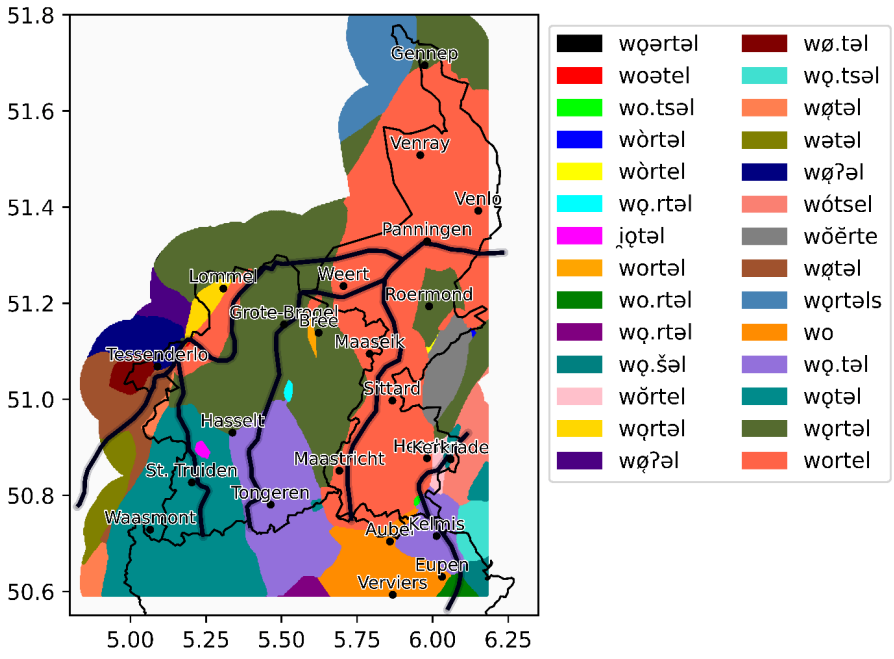


Figure 61: GET-generated variation map of “wortel” (root) with Achel as fixed input.

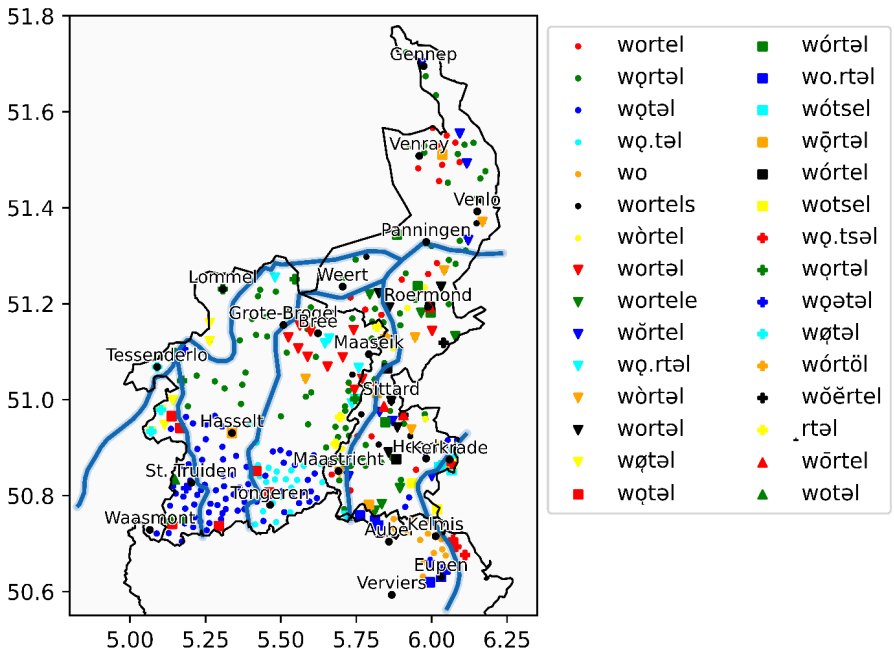


Figure 62: Entries in the WLD for keyword “wortel” (root). Only the 30 most common spelling variants are plotted.

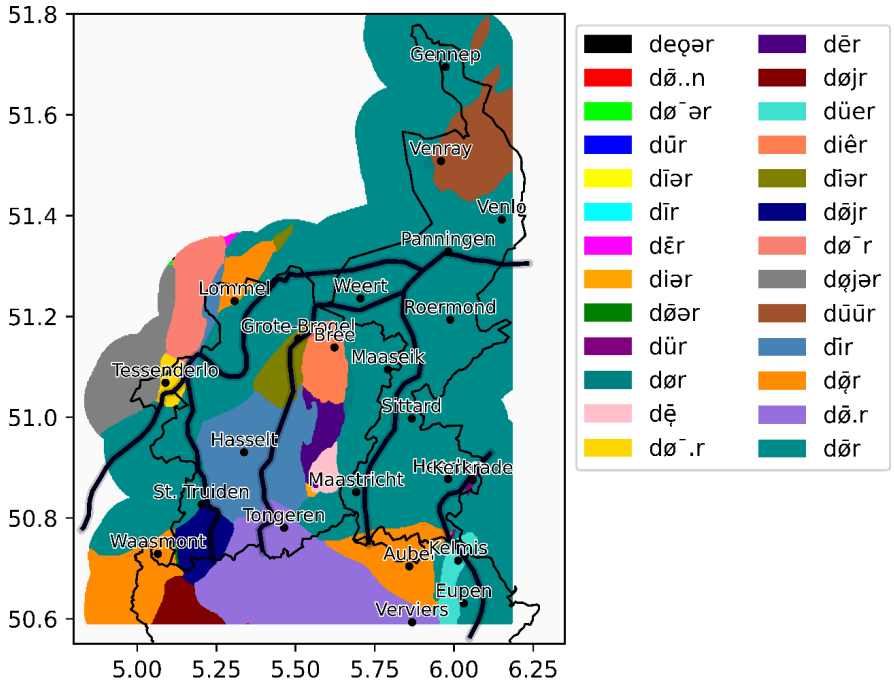


Figure 63: GET-generated variation map of “deur” (door) with Grote-Brogel as fixed input.

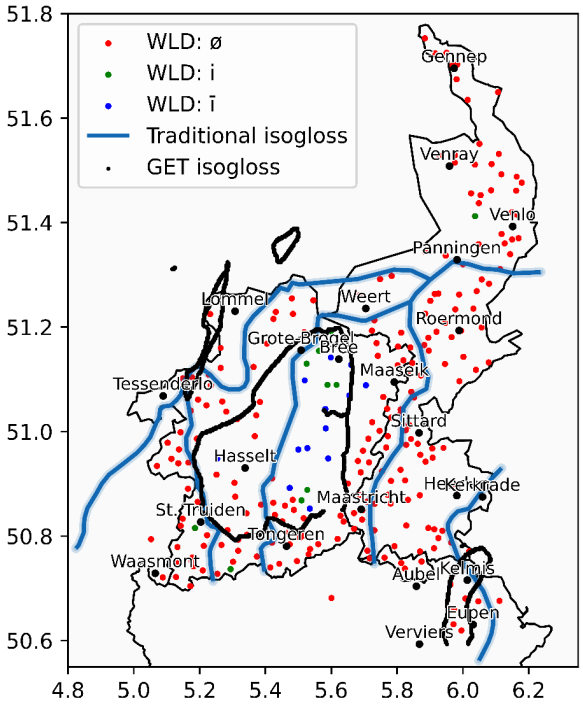


Figure 64: The isophone extracted from the GET prediction, as well as simplified entries in the WLD for keyword “deur” (door).

6.3 Conclusion

We have proposed a modification to the traditional transformer architecture, i.e. the embedding of additional, continuous linguistic information. We have studied the case of the Geographically Embedded Transformer (where we have embedded geographic coordinates), which is comparable in size and training to the normal transformer, but performs better in the task of spelling normalization for dialects of Limburgish if they are geolocated. Due to a lack of curated training data it is difficult to evaluate the performance of GET spelling normalization, although a manual analysis reveals good results.

The GET architecture can also be used for the task of extremely granular dialect translation and is, to the best of our knowledge, the first ever application of a geographically continuous machine translation model. Due to a lack of a curated dataset it is again difficult to estimate the performance, although a manual analysis reveals good results. Unlike the case of dialect identification, we find that geographic discontinuities in phonology do not impact results of the dialect translation task. The GET succesfully predicts known phonological variation in Limburgish and can be used to generate language variation maps, which approximate traditional dialectology maps. The machine translation task is clearly impacted by the poor quality of normalization of the WLD, and an improved normalization scheme could yield better results. The task can additionally be improved through a more extensive hyperparameter search and curation of high-quality datasets, which due to a lack of resources and time constraints were outside the scope of this thesis.

7 Conclusion

We have curated the WLD through various preprocessing steps which allowed us to generate datasets for different NLP tasks: short-text dialect identification, dialect normalization, and dialect translation.

For short-text DID, we have performed, to the best of our knowledge, the most granular identification task yet. We investigated multiple deep learning architectures and found that stacked LSTM architectures outperform feedforward neural networks, transformer encoders, and our baseline. The errors of the DID task were interpreted linguistically: horizontal errors are larger than vertical errors which is in agreement with Limburgish dialectology. Additionally, we analyzed four regions with the highest errors: we found that localized discontinuities in the case of a transition to Brabantian, influence by Venlo and the isolated position of Ripuarian, can be attributed to an increase in errors. We studied in detail the case of Venlo, for which we found additional evidence to an old hypothesis in Limburgish dialectology. We found a linear correlation between the error of dialect identification and the length of words, as well as a general negative correlation between the frequency of diacritics in a word and the mean error, indicating that diacritics are an essential marker used by the trained model for the DID task. We found no noticeable correlations between norms of valence, arousal, dominance, and the mean error, although we found a weak correlation between a low age of acquisition and a higher error. We also found no relation between the frequency of words in natural language contexts and the DID performance. A semantic analysis revealed that agricultural terms result in better DID, while general and professional terms in poorer DID. The best-performing semantic category is jargon related to the Limburgish mines, while the lowest category is jargon related to very localized zinc mines and turf extraction, which had a discontinuous influx of non-endogenous vocabulary.

We proposed a novel adaptation of the transformer: by embedding continuous linguistic information such as geographic coordinates we can outperform the normal transformer in a dialect normalization task. We evaluated four metrics on their interpretability and determined upper and lower boundaries on this zero-resource task. We manually analyzed the dialect normalization predictions and found a close correspondence to results in traditional Limburgish dialectology.

We used the GET architecture on the task of dialect machine translation and found that it generalizes well and achieves good results, both according to manual analysis as well as using the metrics and boundaries we evaluated. Furthermore, a novel approach allows for the creation of language variation maps, which approximate traditional dialectology maps. This architecture is, to the best of our knowledge, the most granular machine translation implementation so far.

The applications that we have studied tackle fundamental problems of non-standardized, low-resource NLP: by normalizing the phonological variation and implementing dialect translation, existing datasets can be enhanced and normalized, which opens doors for other NLP pipelines. Within the Germanic languages, Limburgish has specifically few corpora and datasets, qualitative curation could kickstart more research as many applications are bottlenecked due to a lack of qualitative, annotated data. We encountered the limitations of time and resource constraints, especially in the training of dialect translation and normalization, and additional resources could lead to even better performance of these applications.

Future research could implement the normalization and DID models in order to generate larger datasets and corpora. This could improve the status of Limburgish or other non-standardized languages as low-resource languages. In combination with a corpus, the character-based dialect machine translation model could be enhanced into a full machine translation model capable of translating between any pair of Limburgish dialects.

Availability

The trained models, generated datasets, figures, and GIS data will be made available on the GitHub repository `https://github.com/AndreasJCSimons/LimburgishNLP`.

Appendix A: Veldeke spelling

Limburgish spelling conventions are unusual as Limburgish has been going through an atypical process of codification. A general consensus on Limburgish orthography has existed since the late 19th century, mostly centered around Dutch Limburg (Bakkes et al., 2023) (Assendelft, 2019). However, most codification materials such as dictionaries are (ideologically) based per locality, leading to a codification of Limburgish not towards a single standard language but a *multidialectal space* (Assendelft, 2019), (Cornips et al., 2016): the Limburgish dialects co-exist as equally important varieties of the Limburgish language ("unity in linguistic diversity" (Cornips et al., 2016)), yet are simultaneously purported to be distinct and mutually intelligible, from each other as well as other languages such as Dutch and German.

From 1928 onwards, however, the so-called *Veldeke spelling* [7] is periodically published with the goal of writing all Limburgish dialects in a standard orthographic system. After the Netherlands recognized Limburgish as a regional language under the *European Charter for Regional or Minority Languages*, the *Raod veur 't Limburgs* was created as an advisory body for Limburgish, which officially adopted the Veldeke spelling in 2003. Regardless, local spelling conventions continue to exist on e.g. social media (Cornips et al., 2022), where Limburgish is an active minority language (Jongbloed-Faber et al., 2017), although the Veldeke spelling is becoming increasingly normative (Cornips et al., 2016) and other codification methods such as digital spelling correction are becoming prevalent (Limburgish Academy, b). This means that during the compilation of the surveys that constitute the WLD, Limburgish engaged in ongoing orthographic standardization, which is clearly noticeable in the source material. Certain sources in the WLD such as Goossens' agricultural terms (mostly Belgian Limburg), the RND (Dutch Limburg and Liège) and SGV are transcribed in high-quality phonetic notation (Weijnen et al., 1983), while sources compiled from locality-specific dialect dictionaries typically contain local spelling conventions. The Veldeke orthography is also persistently used in survey answers. Overall, roughly half of the WLD consists of such local spelling conventions or the Veldeke spelling. Below, we will provide a summary of the orthography of the 2003 Veldeke spelling (Bakkes et al., 2023) which was used during the normalization preprocessing step of the WLD:

The goal of the Veldeke orthography is to provide spelling conventions that can be used for the phonology of all Limburgish dialects. Due to the large number of vowels in the Limburgish dialects, the Veldeke spelling contains a large number of diacritics, although it tries to use as little diacritics as possible and relies on familiarity with Dutch orthography wherever possible. It distinguishes 12 short vowels, 16 long vowels, and 58 diphthongs and triphthongs containing a paragoge ("naslag" in Dutch). It also uses the 4 Dutch diphthongs. Orthography that matches with standard Dutch is put in bold:

[7] of the eponymous organization Veldeke Limburg (°1926) which is active in Belgium and the Netherlands and aims to promote the Limburgish dialects.

short vowel	example
ie	
i	
è	mais (French), Ben (English)
e	
uu	
u	
ö	Köln (German)
oe	
ó	between bot en boet (Dutch), IPA: ʊ̞
o	
a	
á	De Smákt (Northern Limburgish)

long vowel	example
iê	bier (Dutch)
ie	
ee	
e	
ae	prêtre (French), fair (English)
àè	pet (Dutch) but longer
uu	
u	
eu	
äö	sœur (French)
oê	boer (Dutch
oe	
oo	
o	
ao	four (English)
aa	
â	car (English)

The long paragoged vowels consist of adding an additional sound to a monophthong or diphthong, usually the schwa. Below we present all existing combinations:

ieë	ieè	iej	iew	ië	eë	i-j	eej	iw	eew	èë	aeë	aej	èw	aew	eë
àèë	ej	aej	ew	àèw	uuë	uuè	uuj	uuw	uë	euë	uj	euj	uw	euw	öë
öä	äöj	öw	äöw	oeë	oeè	oej	oew	óë	óa	ooë	ój	ooj	ów	oow	oë
oa	aoë	oj	aoj	aow	aë	aaë	aj	aaj	aaw						

Some diphthongs are taken from Dutch orthography: **ei**, **ij**, **ui**, **ou**, **au** although Limburgish does not distinguish between **ou** and **au** or **ei** and **ij** and the latter of each option is therefore discouraged. Limburgish mostly uses Dutch orthography for consonants with some exceptions:

consonant	example
gk	garçon (French)
tj, dj	hóndj (dog): palatalization (“mouillering”)
nj, lj	Spanje (Dutch): palatalization

Other typical examples of Limburgish orthography are the use of “zj” (IPA: ʒ) and “sj” (cf. German schön).

Appendix B: List of kloeke codes

Below we provide a list of all kloeke codes featured in the WLD after the first geographic preprocessing step, the frequency of their usage and their geographic coordinates in the EPSG:4326 standard (after our correction algorithm). The coordinates are rounded to 3 decimals but the entire dataset is made digitally available:

Kloeke	Locality name	Coordinates (N, E)	Freq.
Q168p	'S-Herenelderen	50.809, 5.503	2385
Q036a	Aalbeek	50.901, 5.852	2
P179	Aalst	50.781, 5.213	1
P179p	Aalst	50.781, 5.213	1981
L282p	Achel	51.255, 5.480	6391
L191p	Afferden	51.635, 6.014	2473
L192b	Aijen	51.584, 6.043	429
L372a	Aldeneik	51.103, 5.805	508
P120p	Alken	50.875, 5.308	3184
P120	Alken	50.875, 5.308	1
Q077a	Alt-Hoeselt	50.828, 5.492	339
L318d	Altweert	51.249, 5.693	1887
L318e	Altweerterheide	51.220, 5.679	1790
Q102	Amby	50.862, 5.732	1
Q102p	Amby	50.862, 5.732	5935
L244c	America	51.437, 5.980	1514
Q038p	Amstenrade	50.938, 5.933	2527
L250p	Arcen	51.476, 6.179	3193
L417p	As	51.006, 5.583	10270
L333p	Asenray	51.199, 6.057	1846
L331a	Asselt	51.227, 6.013	117
L263p	Asten	51.404, 5.746	2
Q258p	Astenet	50.688, 6.035	27
P169p	Attenhoven	50.767, 5.093	74
Q249p	Aubel	50.704, 5.859	485
L295	Baarlo	51.331, 6.093	2
L295p	Baarlo	51.331, 6.093	7307
Q279p	Baelen	50.631, 5.971	505
L324p	Baexem	51.226, 5.885	2837
Q196a	Banholt	50.790, 5.810	651
P221p	Batsheers	50.738, 5.281	40
L327p	Beegden	51.191, 5.920	2884
L204p	Beek	51.542, 5.637	4
L359p	Beek	51.155, 5.603	1274
Q019p	Beek	50.927, 5.811	3442
L300p	Beesel	51.269, 6.042	4990
L297p	Belfeld	51.311, 6.113	2495
Q106p	Bemelen	50.847, 5.764	419
Q113b	Benzenrade	50.863, 5.979	101
P053p	Berbroek	50.950, 5.209	1369
Q163p	Berg	50.797, 5.504	1315
L429a	Berg	51.016, 5.787	1459
Q103p	Berg/Terblijt	50.862, 5.780	3084
L192p	Bergen	51.59, 6.079	1339
L279p	Bergeyk	51.267, 5.396	17
L265c	Beringe	51.337, 5.948	638
K358p	Beringen	51.050, 5.221	7122
L378q	Berkelaar	51.123, 5.872	159
P187p	Berlingen	50.820, 5.311	1094
P024p	Betekom	50.986, 4.781	1

Kloeke	Locality name	Coordinates (N, E)	Freq.
L162p	Beugen	51.674, 5.940	2
K318p	Beverlo	51.089, 5.234	7024
Q072p	Beverst	50.892, 5.473	4673
P176b	Bevingen	50.798, 5.177	849
Q083p	Bilzen	50.868, 5.510	11550
P113p	Binderveld	50.861, 5.167	951
Q029p	Bingelrade	50.969, 5.941	1415
P095p	Binkom	50.872, 4.884	1
L115z	Bisselt	51.759, 5.904	7
K218p	Bladel	51.367, 5.215	1
Q121c	Bleijerheide	50.853, 6.066	7695
L269p	Blerick	51.368, 6.149	10207
L215p	Blitterswijck	51.531, 6.109	3949
L317p	Bocholt	51.192, 5.600	8982
Q211p	Bocholtz	50.818, 6.007	3526
L183p	Boekel	51.603, 5.673	1
L269b	Boekend	51.378, 6.117	2570
L287p	Boeket	51.289, 5.729	2377
P224p	Boekhout	50.746, 5.233	741
K361a	Boekt Heikant	51.015, 5.297	1180
Q002c	Bokrijk	50.953, 5.421	449
P051a	Bolderberg	50.989, 5.273	105
Q160p	Bommershoven	50.787, 5.380	1472
L423a	Booien	51.034, 5.754	3
Q011p	Boorsem	50.941, 5.715	3450
Q096a	Borgharen	50.879, 5.688	2629
Q156p	Borgloon	50.802, 5.344	6961
L281p	Borkel	51.298, 5.443	28
P218p	Borlo	50.742, 5.181	2820
L428p	Born	51.034, 5.810	3520
L357p	Bos	51.163, 5.543	52
L289h	Boshoven	51.261, 5.684	1029
L331b	Boukoul	51.215, 6.045	1791
L187p	Boxmeer	51.649, 5.945	16
L360p	Bree	51.142, 5.599	13750
L247p	Broekhuizen	51.488, 6.163	1604
L247z	Broekhuizenvorst	51.496, 6.157	209
Q159p	Broekom	50.782, 5.332	1104
L434a	Broeksittard	51.002, 5.895	1326
Q035p	Brunssum	50.960, 5.979	7236
P178	Brustem	50.802, 5.223	1
P178p	Brustem	50.802, 5.223	1383
L426p	Buchten	51.042, 5.811	4879
L285p	Budel	51.274, 5.574	36
L323p	Buggenum	51.234, 5.977	1972
Q096p	Bunde	50.897, 5.733	1530
P182p	Buvingen	50.753, 5.175	1035
Q191p	Cadier	50.828, 5.768	707
L245a	Castenray	51.489, 6.035	2548
Q121a	Chevremont	50.876, 6.066	3129

Kloeke	Locality name	Coordinates (N, E)	Freq.
L159p	Cuijk	51.730, 5.880	53
K354p	Deurne	51.039, 5.096	13
L244p	Deurne	51.470, 5.790	14
L097p	Dieden	51.820, 5.608	1
Q071p	Diepenbeek	50.911, 5.420	10592
Q071	Diepenbeek	50.911, 5.420	3
L381c	Diergaarde	51.083, 5.965	8
P041p	Diest	50.985, 5.053	2
L431p	Dieteren	51.076, 5.843	2437
Q242p	Diets-Heur	50.745, 5.484	761
L421p	Dilsen	51.035, 5.725	3070
Q027p	Doenrade	50.968, 5.907	6163
L259p	Dommelen	51.356, 5.430	1
Q001b	Donk	50.939, 5.135	1
P049	Donk	50.939, 5.135	1
L204a	Donk	51.604, 5.687	1
P049p	Donk	50.939, 5.135	1629
L415a	Dorne	51.044, 5.625	90
P111p	Drieslinter	50.844, 5.047	2
P115p	Duras	50.833, 5.148	1341
Q176p	Eben-Emael	50.793, 5.667	28
L381p	Echt	51.107, 5.872	10236
Q193a	Eckelrade	50.806, 5.766	249
L290a	Egchel	51.313, 5.970	717
Q086p	Eigenbilzen	50.874, 5.576	7021
Q198p	Eijsden	50.772, 5.710	4103
L288c	Eind	51.278, 5.773	2344
L430p	Einighausen	51.003, 5.826	2955
L430	Einighausen	51.003, 5.826	2
Q007p	Eisden	50.986, 5.713	5616
Q007	Eisden	50.986, 5.713	2
L353p	Eksel	51.168, 5.405	8451
L353	Eksel	51.168, 5.405	1
L419p	Elen	51.066, 5.758	2485
P162p	Eliksem	50.784, 5.012	2
L320a	Ell	51.220, 5.795	6050
L363p	Ellikom	51.130, 5.525	2263
Q017p	Elsloo	50.950, 5.771	953
L025p	Elst	51.917, 5.855	2
P185p	Engelmanshoven	50.772, 5.256	297
K353b	Engsbergs	51.038, 5.067	7
Q207p	Epen	50.777, 5.912	4310
Q284p	Eupen	50.631, 6.031	3309
K358a	Eversel	51.026, 5.240	39
Q119p	Eygelshoven	50.894, 6.060	2078
Q262p	Eynatten	50.694, 6.082	305
Q202p	Eys	50.825, 5.935	9660
P166p	Ezemaal	50.777, 5.002	1
Q106a	Gasthuis/'t Rooth	50.839, 5.785	2
P107p	Geetbets	50.892, 5.114	4
L371a	Geistingen	51.136, 5.819	2858
L240p	Geldrop	51.423, 5.561	1
Q021p	Geleen	50.986, 5.842	9447
Q004p	Gelieren Bret	50.968, 5.527	1758
P186p	Gelinden	50.765, 5.261	2327
Q087p	Gellik	50.884, 5.609	828
Q251p	Gemmenich	50.746, 5.997	968
Q251	Gemmenich	50.746, 5.997	2
K314a	Genendijk	51.084, 5.106	6
Q003p	Genk	50.965, 5.500	9287

Kloeke	Locality name	Coordinates (N, E)	Freq.
L164p	Gennep	51.701, 5.968	8235
Q173	Genoelselderen	50.802, 5.537	1
Q173p	Genoelselderen	50.802, 5.537	876
L360a	Gerdingen	51.147, 5.588	635
Q018p	Geulle	50.924, 5.754	5486
Q019z	Kelmond	50.923, 5.801	269
L214a	Geysteren	51.551, 6.050	2649
P175p	Gingelom	50.752, 5.132	3014
Q245p	Glons	50.752, 5.544	4
Q002a	Godschei	50.948, 5.398	1371
P155p	Goetsenhoven	50.768, 4.949	1
Q153p	Gors-Opleeuw	50.824, 5.392	1309
P116p	Gorsem	50.835, 5.163	394
P190p	Gotem	50.804, 5.313	52
L326p	Grathem	51.192, 5.859	3857
P108p	Grazen	50.873, 5.127	279
L425	Grevenbicht	51.044, 5.770	1
L425p	Grevenbicht	51.044, 5.770	4657
L244b	Griendtsveen	51.444, 5.889	811
L119p	Groesbeek	51.774, 5.931	14
L032p	Groessen	51.932, 6.029	4
Q193p	Gronsveld	50.811, 5.731	8132
Q019b	Groot Genhout	50.927, 5.823	145
P184p	Groot-Gelmen	50.783, 5.266	1764
P184	Groot-Gelmen	50.783, 5.266	1
Q156a	Groot-Loon	50.793, 5.364	166
L356p	Grote-Brogel	51.156, 5.508	1891
Q170p	Grote-Spouwen	50.833, 5.554	1609
L249p	Grubbenvorst	51.420, 6.148	2151
L366p	Gruitrode	51.089, 5.589	5120
Q079p	Guigoven	50.840, 5.399	1099
Q203p	Gulpen	50.816, 5.895	8947
P195p	Gutschoven	50.772, 5.317	2409
L429p	Guttecoven	51.014, 5.818	5803
Q121d	Haanrade	50.889, 6.072	2159
L322p	Haelen	51.237, 5.954	5715
P048p	Halen	50.948, 5.114	4672
L320c	Haler	51.188, 5.780	1849
P161p	Halle	50.813, 5.124	3
P173p	Halmaal	50.804, 5.152	886
L286p	Hamont	51.251, 5.545	8613
Q160a	Haren	50.681, 5.601	220
Q002p	Hasselt	50.930, 5.338	14598
Q261p	Hauset	50.708, 6.071	79
L352p	Hechtel	51.124, 5.364	4534
L352	Hechtel	51.124, 5.364	2
Q110p	Heek	50.878, 5.869	1439
L328p	Heel	51.180, 5.895	7333
Q105	Heer	50.840, 5.725	1
Q105p	Heer	50.840, 5.725	2398
Q113p	Heerlen	50.891, 5.976	15319
Q112a	Heerlerheide	50.918, 5.966	3279
P197p	Heers	50.751, 5.302	4340
P197	Heers	50.751, 5.302	3
Q094p	Hees	50.846, 5.612	1585
Q081a	Heesveld-Eik	50.895, 5.499	672
L261p	Heeze	51.383, 5.572	2
L246c	Hegelsom	51.437, 6.048	247
L165p	Heijen	51.674, 5.981	4220
L121p	Heiland	51.754, 5.953	1

Kloeke	Locality name	Coordinates (N, E)	Freq.
P219a	Heiselt	50.732, 5.228	48
Q164p	Heks	50.769, 5.357	1066
L413p	Helchteren	51.057, 5.383	3338
L291p	Helden	51.320, 6.000	5537
L265a	Helenaveen	51.389, 5.918	1
L237p	Helmond	51.476, 5.662	1
P191p	Hendrieken	50.799, 5.327	37
Q158a	Henis	50.799, 5.470	712
Q254p	Henri-Chapelle	50.677, 5.932	395
K316p	Heppen	51.109, 5.228	2334
L419a	Heppeneert	51.079, 5.795	1
Q278a	Herbesthal	50.661, 5.995	67
Q174p	Herderen	50.807, 5.574	1021
Q257p	Hergenrath	50.709, 6.032	94
P050	Herk-De-Stad	50.941, 5.166	1
P050p	Herk-De-Stad	50.941, 5.166	5344
L384p	Herkenbosch	51.155, 6.066	781
Q243p	Herstappe	50.727, 5.426	199
L330	Herten	51.181, 5.965	1
L330p	Herten	51.181, 5.965	11856
P121a	Herten	50.833, 5.333	120
L079p	Herwen	51.885, 6.101	1
Q187a	Heugem	50.830, 5.706	2384
Q175a	Heukelom	50.808, 5.620	2
K360p	Heusden	51.038, 5.281	2854
L292p	Heythuysen	51.250, 5.901	5477
Q085p	Hoelbeek	50.873, 5.558	107
Q039p	Hoensbroek	50.924, 5.929	8303
P188p	Hoepertingen	50.811, 5.284	6195
P188	Hoepertingen	50.811, 5.284	2
Q077p	Hoeselt	50.850, 5.487	8498
Q208b	Holset	50.776, 5.986	20
L426z	Holtum	51.047, 5.821	2557
Q121z	Holz	50.870, 6.093	48
Q250p	Hombourg	50.723, 5.921	156
Q193b	Honthem	50.817, 5.797	8
K217p	Hoogeloon	51.397, 5.268	1
Q165p	Hopmaal	50.772, 5.379	1188
Q165	Hopmaal	50.772, 5.379	1
L325p	Horn	51.208, 5.946	4918
L246p	Horst	51.453, 6.054	5503
L269a	Hout-Blerick	51.360, 6.127	1579
L414p	Houthalen	51.031, 5.372	7443
L414	Houthalen	51.031, 5.372	1
Q100p	Houthem	50.872, 5.795	1964
K343p	Houtvenne	51.042, 4.809	1
P037p	Houwaart	50.933, 4.860	1
L103p	Huisseling	51.788, 5.641	1
Q109p	Hulsberg	50.890, 5.858	1761
K353c	Hulst Konijnsberg	51.075, 5.140	68
L320p	Hunsel	51.189, 5.813	2617
L289a	Hushoven	51.263, 5.697	3074
Q203b	Ingber	50.818, 5.858	1498
Q096b	Itteren	50.898, 5.703	1858
L321a	Ittervoort	51.177, 5.833	2651
Q028p	Jabeek	50.981, 5.940	1450
Q157p	Jesseren	50.806, 5.391	1315
P219p	Jeuk	50.734, 5.210	7493
Q121e	Kaalheide	50.863, 6.039	2299
Q188	Kanne	50.814, 5.669	1
Q188p	Kanne	50.814, 5.669	3481
L329a	Kapel in t Zand	51.182, 5.996	2265
L316p	Kaulille	51.190, 5.518	6126
L318a	Keent	51.243, 5.702	2038
Q191q	Keer	50.828, 5.768	12
Q255p	Kelmis	50.715, 6.013	1341
L320b	Kelpen	51.222, 5.825	2281
Q015b	Kerensheide	50.976, 5.781	542
K317a	Kerkhoven	51.160, 5.264	1803
P180p	Kerkom	50.773, 5.183	1140
Q121p	Kerkrade	50.866, 6.066	13360
P055p	Kermt	50.948, 5.253	3730
Q152p	Kerniel	50.817, 5.363	1007
L298p	Kessel	51.291, 6.057	3306
L298a	Kesseleik	51.285, 6.018	1648
Q094a	Kesselt	50.839, 5.628	8
L370p	Kessenich	51.151, 5.820	2353
Q176a	Ketsingen	50.800, 5.512	1093
Q283p	Kettenis	50.646, 6.046	109
Q002b	Kiewit	50.963, 5.355	621
L369p	Kinrooi	51.145, 5.741	4636
Q098a	Kl./Gr. Haasdal	50.898, 5.819	6
P194p	Klein-Gelmen	50.771, 5.276	26
L315	Kleine-Brogel	51.173, 5.452	1
L315p	Kleine-Brogel	51.173, 5.452	1629
Q092p	Kleine-Spouwen	50.838, 5.547	176
Q111p	Klimmen	50.876, 5.882	16825
K359p	Koersel	51.057, 5.269	4083
L281a	Kolonie	51.252, 5.385	44
L432a	Koningsbosch	51.052, 5.957	1359
L265e	Koningslust	51.357, 5.995	96
Q167p	Koninksem	50.767, 5.441	2094
K318a	Korspel	51.089, 5.248	5
P118b	Kortenbos	50.863, 5.242	73
Q074p	Kortessem	50.860, 5.387	4669
Q074	Kortessem	50.859, 5.387	1
P215p	Kortijs	50.707, 5.150	27
P031p	Kortrijk-Dutsel	50.926, 4.806	1
Q011a	Kotem	50.948, 5.740	65
P118p	Kozen	50.875, 5.239	1360
Q016q	Krawinkel	50.960, 5.812	58
L265b	Kronenberg	51.416, 5.999	414
Q112c	Kunrade	50.877, 5.936	763
P057p	Kuringen	50.944, 5.305	3823
P187a	Kuttekoven	50.810, 5.328	345
K314p	Kwaadmechelen	51.102, 5.148	5316
Q249a	La Clouse	50.692, 5.898	2
L379p	Laak	51.263, 5.914	1394
P167p	Laar	50.775, 5.036	127
L288b	Laar	51.301, 5.607	2150
K214p	Lage Mierde	51.407, 5.149	1
Q088p	Lanaken	50.900, 5.650	6167
Q189p	Lanaye	50.781, 5.694	2
P171	Landen	50.753, 5.081	1
P171p	Landen	50.753, 5.081	756
L422	Lanklaar	51.018, 5.724	3
L422p	Lanklaar	51.018, 5.724	6732
Q240p	Lauw	50.740, 5.415	2066
L262p	Leende	51.348, 5.552	1
L283a	Leenderstrijp	51.334, 5.543	9

Kloeke	Locality name	Coordinates (N, E)	Freq.
L332a	Leeuwen	51.212, 5.999	6
Q208a	Lemiers	50.785, 5.994	1
K317p	Leopoldsburg	51.122, 5.264	4578
L289b	Leuken	51.258, 5.733	3821
L211p	Leunen	51.515, 5.976	5712
Q006p	Leut	50.992, 5.736	1181
Q006	Leut	50.992, 5.736	2
L324a	Leveroij	51.253, 5.841	813
L242p	Lierop	51.419, 5.681	2
L203p	Lieshout	51.518, 5.595	2
Q281p	Limbourg	50.612, 5.940	2
L434p	Limbricht	51.012, 5.838	2582
Q104a	Limmel	50.866, 5.708	1498
L355a	Linde	51.114, 5.467	765
P046p	Linkhout	50.966, 5.137	3254
L376p	Linne	51.156, 5.942	1639
L376	Linne	51.156, 5.942	1
Q190p	Lixhe	50.755, 5.680	1
P047	Loksbergen	50.934, 5.070	1
P047p	Loksbergen	50.934, 5.070	5969
L250z	Lomm	51.449, 6.172	36
K278p	Lommel	51.231, 5.308	11437
Q259p	Lontzen	50.680, 6.007	1008
L248p	Lottum	51.461, 6.162	3021
L316a	Lozen	51.204, 5.560	1000
P094p	Lubbeek	50.882, 4.842	1
P051p	Lummen	50.986, 5.192	4591
Q016p	Lutterade	50.975, 5.827	5553
L284p	Maarheeze	51.312, 5.616	18
L377p	Maasbracht	51.155, 5.896	5231
L267p	Maasbree	51.359, 6.047	7622
L372p	Maaseik	51.095, 5.792	10548
L212p	Maashees	51.571, 6.0350	56
Q009p	Maasmechelen	50.964, 5.696	9333
Q009	Maasmechelen	50.964, 5.696	3
L332p	Maasniel	51.198, 6.012	5257
Q095p	Maastricht	50.849, 5.694	34811
Q180	Mal	50.775, 5.535	1
Q180p	Mal	50.775, 5.535	2224
L113p	Malden	51.780, 5.854	4
Q192p	Margraten	50.82, 5.828	2508
Q089p	Martenslinde	50.852, 5.535	3029
L292a	Maxet	51.251, 5.863	1014
Q204a	Mechelen	50.796, 5.926	6964
P220p	Mechelen-Bovelingen	50.743, 5.263	1047
K312p	Meerhout	51.132, 5.077	4
L217p	Meerlo	51.512, 6.087	6354
Q015a	Meers	50.963, 5.741	397
Q099p	Meerssen	50.879, 5.772	5374
L424	Meeswijk	51.001, 5.747	3
L424p	Meeswijk	51.001, 5.747	4294
L364p	Meeuwen	51.098, 5.519	8482
L265p	Meijel	51.344, 5.886	15538
L246b	Melderslo	51.462, 6.085	741
P045	Meldert	50.999, 5.142	1
P045p	Meldert	50.999, 5.142	2479
L383p	Melick	51.157, 6.014	3530
P157p	Melkwezer	50.823, 5.059	1
P176a	Melveren	50.831, 5.199	1448
Q282p	Membach	50.619, 5.995	117
Q169p	Membruggen	50.816, 5.535	919

Kloeke	Locality name	Coordinates (N, E)	Freq.
Q034p	Merkelbeek	50.955, 5.934	2993
L209p	Merselo	51.528, 5.928	5835
Q198a	Mesch	50.764, 5.734	872
L245p	Meterik	51.456, 6.025	2025
P193p	Mettekoven	50.780, 5.290	979
Q196p	Mheer	50.781, 5.792	7225
L159a	Middelaar	51.724, 5.916	2897
P183p	Mielen-Boven-Aalst	50.755, 5.215	1773
Q177p	Millen	50.784, 5.560	2180
L163a	Milsbeek	51.725, 5.95	3532
Q199p	Moelingen	50.757, 5.716	1043
Q195z	Moerslag	50.787, 5.758	14
K276p	Mol	51.191, 5.115	2
L319p	Molenbeersel	51.169, 5.737	2967
L319	Molenbeersel	51.169, 5.737	1
K355p	Molenstede	51.004, 5.017	6
P214p	Montenaken	50.722, 5.128	1585
L382p	Montfort	51.129, 5.945	11363
L382	Montfort	51.129, 5.945	3
Q253p	Montzen	50.708, 5.962	4842
L115p	Mook	51.753, 5.884	903
Q018a	Moorveld	50.920, 5.765	411
Q090p	Mopertingen	50.862, 5.576	1758
Q252p	Moresnet	50.721, 5.988	803
Q252a	Moresnet-Eiksken	50.732, 5.994	3
P181	Muizen	50.759, 5.178	1
P181p	Muizen	50.759, 5.178	490
Q082p	Munsterbilzen	50.888, 5.527	1580
Q022p	Munstergeleen	50.977, 5.863	3193
L427z	Nattenhoven	51.010, 5.775	18
Q077b	Nederstraat	50.869, 5.483	63
L288p	Nederweert	51.292, 5.748	5980
L294p	Neer	51.262, 5.988	5091
L294	Neer	51.262, 5.988	1
Q019a	Neerbeek	50.950, 5.813	1392
L367p	Neerglabbeek	51.089, 5.616	3111
Q096c	Neerharen	50.908, 5.681	2791
P156p	Neerheilissem	50.760, 4.993	2
P164p	Neerhespen	50.794, 5.052	283
L321p	Neeritter	51.163, 5.803	6206
P165p	Neerlanden	50.778, 5.079	16
P110p	Neerlinter	50.840, 5.021	2
L368p	Neeroeteren	51.089, 5.705	8625
L312p	Neerpelt	51.229, 5.431	8134
Q155a	Neerrepen	50.812, 5.445	266
P138p	Neervelp	50.821, 4.810	1
P168p	Neerwinden	50.766, 5.037	14
Q182p	Nerem	50.766, 5.511	749
Q182	Nerem	50.766, 5.511	1
Q256p	Neu-Moresnet	50.720, 6.024	35
P213p	Niel-Bij-Sint-Truiden	50.741, 5.140	2111
L418p	Niel-bij-As	51.015, 5.601	2327
L418	Niel-bij-As	51.015, 5.601	1
Q117p	Nieuwenhagen	50.906, 6.034	8000
P117p	Nieuwerkerken	50.865, 5.194	2271
P036p	Nieuwrode	50.949, 4.833	1
L433p	Nieuwstadt	51.038, 5.861	3957
Q205q	Nijswiller	50.808, 5.957	64
Q197p	Noorbeek	50.769, 5.812	5165

Kloeke	Locality name	Coordinates (N, E)	Freq.
L322a	Nunhem	51.245, 5.963	4016
Q036p	Nuth	50.927, 5.887	4015
L427p	Obbicht	51.027, 5.782	3434
L380p	Ohé	51.113, 5.832	1166
L216p	Oirlo	51.511, 6.036	9274
Q033p	Oirsbeek	50.950, 5.908	8666
L326q	Oler	51.215, 5.829	79
K313p	Olmen	51.142, 5.149	10
L243a	Ommel	51.422, 5.748	1
Q198b	Oost-Maarland	50.796, 5.711	3691
L063p	Oosterhout	51.880, 5.827	1
K315p	Oostham	51.104, 5.177	3473
L216a	Oostrum	51.529, 6.018	3076
L416p	Opglabbeek	51.043, 5.582	11387
Q010p	Opgrimbie	50.944, 5.681	3264
Q010	Opgrimbie	50.944, 5.681	1
P222p	Opheers	50.737, 5.294	2520
L371p	Ophoven	51.127, 5.802	6970
L362p	Opitter	51.117, 5.646	2957
L186p	Oploo	51.607, 5.874	1
L415p	Opoeteren	51.069, 5.655	5445
P177a	Ordingen	50.812, 5.235	1071
L288a	Ospel	51.298, 5.784	5775
L163p	Ottersum	51.703, 5.982	7036
Q095a	Oud-Caberg	50.866, 5.664	2738
Q114p	Oud-Valkenburg	50.852, 5.859	9
Q003a	Oud-Waterschei	50.989, 5.534	534
Q001a	Oud-Winterslag	50.989, 5.493	118
P154p	Outgaarden	50.766, 4.919	1
P163p	Overhespen	50.795, 5.036	10
L209a	Overloon	51.571, 5.947	14
L314p	Overpelt	51.216, 5.424	5788
Q157a	Overrepen	50.807, 5.430	849
P170p	Overwinden	50.755, 5.047	1
K357	Paal	51.040, 5.173	1
K357p	Paal	51.040, 5.173	6443
L328a	Panheel	51.176, 5.872	4
L359a	Panhoven	51.134, 5.471	1
L077p	Pannerden	51.891, 6.037	1
L290p	Panningen	51.329, 5.982	6567
L355p	Peer	51.132, 5.455	6931
L381b	Peij	51.106, 5.902	3105
P091p	Pellenberg	50.871, 4.794	1
Q088a	Pietersem	50.896, 5.665	4
Q161p	Piringen	50.787, 5.420	1268
Q253a	Plombieres	50.738, 5.961	1
L387p	Posterholt	51.118, 6.039	7972
L381a	Putbroek	51.096, 5.989	696
Q032a	Puth	50.956, 5.870	4456
P210p	Raatshoven	50.739, 5.029	7
Q263p	Raeren	50.676, 6.111	418
P108a	Ransberg	50.875, 5.042	11
Q111*	Ransdaal	50.863, 5.892	3
Q111q	Ransdaal	50.863, 5.891	1182
Q072a	Rapertingen	50.906, 5.360	267
Q203a	Reijmerstok	50.800, 5.838	753
Q012p	Rekem	50.922, 5.689	7238
Q012	Rekem	50.922, 5.689	1
Q248p	Remersdaal	50.730, 5.879	720
L358p	Reppel	51.154, 5.563	2902
L299p	Reuver	51.283, 6.079	9183
L299	Reuver	51.283, 6.079	1
Q175p	Riemst	50.809, 5.599	1666
L258p	Riethoven	51.353, 5.386	1
Q194p	Rijckholt	50.800, 5.733	691
P189p	Rijkel	50.808, 5.260	338
Q168a	Rijkhoven	50.834, 5.516	2350
Q158p	Riksingen	50.807, 5.462	2542
P035p	Rillaar	50.974, 4.895	1
Q117b	Rimburg	50.916, 6.084	2031
L206p	Rixtel	51.505, 5.645	1
Q184p	Roclenge-Sur-Geer	50.757, 5.594	21
L329p	Roermond	51.193, 5.994	14398
L293p	Roggel	51.264, 5.925	2625
Q076p	Romershoven	50.858, 5.459	1544
L373p	Roosteren	51.083, 5.817	2628
Q093p	Rosmeer	50.846, 5.575	3269
L420p	Rotem	51.052, 5.737	6187
Q099q	Rothem	50.878, 5.740	2786
P223p	Rukkelingen	50.728, 5.254	938
P223	Rukkelingen	50.728, 5.254	2
Q120p	Rukker	50.864, 6.004	1466
P107a	Rummen	50.891, 5.165	1684
Q035a	Rumpen	50.939, 5.968	378
P114p	Runkelen	50.848, 5.152	154
Q241p	Rutten	50.747, 5.442	1398
Q200p	S-Gravenvoeren	50.759, 5.762	1706
L188p	Sambeek	51.636, 5.966	1
Q118p	Schaesberg	50.908, 6.026	4590
K356p	Schaffen	51.001, 5.083	21
L281q	Schaft	51.299, 5.461	3
P058a	Schakkebroek	50.906, 5.203	82
Q081p	Schalkhoven	50.842, 5.449	152
Q098p	Schimmert	50.907, 5.825	12990
Q115p	Schin op Geul	50.856, 5.871	269
Q032p	Schinnen	50.942, 5.886	8783
Q030p	Schinveld	50.970, 5.977	3801
K353a	Schoot	51.067, 5.037	12
P052p	Schulen	50.964, 5.186	2080
L266	Sevenum	51.412, 6.037	1
L266p	Sevenum	51.412, 6.037	10292
Q101a	Sibbe/$slash$IJzere	50.839, 5.832	555
L192a	Siebengewald	51.649, 6.107	2175
Q116p	Simpelveld	50.840, 6.001	4478
L113a	Sint Anna	51.822, 5.851	1
Q195p	Sint Geertruid	50.796, 5.765	743
L313p	Sint-Huibrechts-Lille	51.225, 5.484	4947
L377a	Sint Joost	51.117, 5.899	1
L385p	Sint Odilienberg	51.143, 6.001	2534
Q187p	Sint Pieter	50.832, 5.694	2747
Q154p	Sint-Huibrechts-Hern	50.828, 5.451	1589
P093p	Sint-Joris-Winge	50.908, 4.877	1
P119p	Sint-Lambrechts-Herk	50.902, 5.306	3650
Q247p	Sint-Martens-Voeren	50.749, 5.812	1822
Q247a	Sint-Pieters-Voeren	50.738, 5.823	1117
P176p	Sint-Truiden	50.816, 5.186	15955
Q210	Sippenaeken	50.751, 5.933	1
Q210p	Sippenaeken	50.751, 5.933	133
Q020p	Sittard	50.997, 5.867	14882
Q206p	Slenaken	50.771, 5.869	363

Kloeke	Locality name	Coordinates (N, E)	Freq.
Q181p	Sluizen	50.765, 5.531	1161
L212a	Smakt	51.566, 6.004	2467
Q096d	Smeermaas	50.886, 5.674	1524
L283p	Soerendonk	51.303, 5.573	9
L264p	Someren	51.386, 5.713	2
P054p	Spalbeek	50.949, 5.230	1895
Q031p	Spaubeek	50.943, 5.850	525
Q121b	Spekholzerheide	50.860, 6.025	3671
K359a	Stal	51.072, 5.251	356
Q015p	Stein	50.969, 5.767	8601
K278a	Stevensvennen	51.225, 5.232	253
L378p	Stevensweert	51.131, 5.846	4753
P058p	Stevoort	50.918, 5.251	2902
L296p	Steyl	51.332, 6.121	2302
L423p	Stokkem	51.021, 5.744	6117
Q284a	Stokkem	50.643, 5.997	9
P056p	Stokrooie	50.966, 5.281	2427
L318p	Stramproy	51.193, 5.724	3070
L432p	Susteren	51.065, 5.852	8024
L331p	Swalmen	51.235, 6.032	10420
L318c	Swartbroek	51.230, 5.772	108
Q032b	Sweikhuizen	50.953, 5.846	514
L246a	Swolgen	51.492, 6.117	3413
L270p	Tegelen	51.339, 6.143	11647
L270	Tegelen	51.339, 6.143	2
Q112z	Ten Esschen	50.898, 5.940	2003
Q197a	Terlinden	50.783, 5.835	3469
K358b	Tervant	51.061, 5.195	189
Q118a	Terwinselen	50.867, 6.022	294
K353p	Tessenderlo	51.068, 5.089	9582
K350p	Testelt	51.009, 4.952	2
Q209p	Teuven	50.751, 5.874	1632
L374	Thorn	51.161, 5.840	1
L374p	Thorn	51.161, 5.840	10260
L245b	Tienray	51.495, 6.093	5296
L238p	Tongelre	51.441, 5.501	1
Q162p	Tongeren	50.784, 5.474	12128
Q162	Tongeren	50.784, 5.474	1
L361p	Tongerlo	51.128, 5.660	1294
P022p	Tremelo	50.992, 4.704	2
L318b	Tungelroy	51.213, 5.731	12491
Q112b	Ubachsberg	50.854, 5.948	2215
L072p	Ubbergen	51.840, 5.909	1
Q013p	Uikhoven	50.925, 5.725	2506
P121p	Ulbeek	50.841, 5.309	3401
Q097	Ulestraten	50.906, 5.782	1
Q097p	Ulestraten	50.906, 5.782	5179
Q014	Urmond	50.991, 5.772	2
Q014p	Urmond	50.991, 5.772	6845
Q222p	Vaals	50.771, 6.019	3343
Q037p	Vaesrade	50.931, 5.907	87
Q178p	Val-Meer	50.788, 5.595	3757
Q101p	Valkenburg	50.869, 5.833	10142
L260p	Valkenswaard	51.351, 5.459	1
Q166p	Vechmaal	50.761, 5.374	1640
L268	Velden	51.416, 6.163	2
L268p	Velden	51.416, 6.163	6041
L255p	Veldhoven	51.419, 5.405	1
Q091p	Veldwezelt	50.866, 5.632	2337
P174p	Velm	50.779, 5.132	2358

Kloeke	Locality name	Coordinates (N, E)	Freq.
L163b	Ven-Zelderheide	51.712, 6.023	1433
L271	Venlo	51.370, 6.169	2
L271p	Venlo	51.370, 6.169	19554
L210p	Venray	51.529, 5.976	12393
Q280p	Verviers	50.593, 5.868	1
L244a	Veulen	51.482, 5.955	2518
P196p	Veulen	50.763, 5.306	638
L190p	Vierlingsbeek	51.596, 6.009	1
Q208p	Vijlen	50.788, 5.964	4166
Q107p	Vilt	50.858, 5.810	2
Q246p	Vise	50.735, 5.694	1
Q080	Vliermaal	50.841, 5.429	2
Q080p	Vliermaal	50.841, 5.429	2872
Q075p	Vliermaalroot	50.864, 5.424	1823
Q171p	Vlijtingen	50.833, 5.590	2507
L386	Vlodrop	51.132, 6.078	1
L386p	Vlodrop	51.132, 6.078	4825
Q112p	Voerendaal	50.887, 5.929	2699
L368a	Voorshoven	51.108, 5.696	1
P192p	Voort	50.793, 5.317	2133
P227p	Vorsen	50.704, 5.172	1401
K352p	Vorst	51.080, 5.020	7
L289q	Vrakker	51.263, 5.681	30
Q183p	Vreren	50.752, 5.496	451
Q172p	Vroenhoven	50.828, 5.637	2572
Q113c	Vrusschemig	50.878, 5.997	442
Q008p	Vucht	50.977, 5.713	1604
P043p	Waanrode	50.917, 5.002	2
P211p	Waasmont	50.729, 5.065	728
Q205p	Wahlwiller	50.810, 5.937	305
Q260p	Walhorn	50.675, 6.047	275
P210a	Walsbets	50.738, 5.087	114
P212p	Walshoutem	50.721, 5.088	255
Q084p	Waltwilder	50.864, 5.546	1131
P162a	Wange	50.787, 5.029	12
L184p	Wanroij	51.657, 5.820	33
L214p	Wanssum	51.536, 6.079	4204
L368b	Waterloos	51.092, 5.679	840
Q117a	Waubach	50.915, 6.053	9151
L289p	Weert	51.254, 5.707	17503
Q278p	Welkenraedt	50.661, 5.973	1760
L213p	Well	51.555, 6.093	2649
Q078p	Wellen	50.843, 5.342	7710
L215a	Wellerlooi	51.535, 6.139	2425
Q113a	Welten	50.873, 5.966	1046
Q155p	Werm	50.834, 5.480	977
L375p	Wessem	51.166, 5.882	996
L280p	Westerhoven	51.332, 5.399	3
P211a	Wezeren	50.730, 5.105	27
Q164a	Widooie	50.772, 5.410	374
L106p	Wijchen	51.812, 5.731	1
L354	Wijchmaal	51.135, 5.418	1
L354p	Wijchmaal	51.135, 5.418	1786
P118a	Wijer	50.897, 5.223	1098
Q201p	Wijlre	50.833, 5.898	3740
Q108p	Wijnandsrade	50.906, 5.884	1275
L365p	Wijshagen	51.107, 5.558	975
P172p	Wilderen	50.820, 5.143	1667
Q073p	Wimmertingen	50.883, 5.352	887
Q079a	Wintershoven	50.851, 5.419	1718

Kloeke	Locality name	Coordinates (N, E)	Freq.
Q204p	Wittem	50.813, 5.913	601
L226p	Woensel	51.461, 5.476	1
Q094b	Wolder/Oud-Vroen	50.844, 5.669	2872
Q222q	Wolfhaag/Raren	50.760, 5.998	12
P157a	Wommersom	50.813, 5.017	1
Q186p	Wonck	50.768, 5.633	3
Q104p	Wyck	50.847, 5.703	1490
L244d	Ysselsteyn	51.490, 5.896	331
Q153a	Zammelen	50.814, 5.407	18
P044p	Zelem	50.979, 5.103	2996
P048a	Zelk	50.960, 5.090	5
P177p	Zepperen	50.823, 5.248	3345
K351p	Zichem	51.006, 4.988	1
Q179p	Zichen-Zussen-Bolder	50.795, 5.620	4712
K361p	Zolder	51.024, 5.314	5662
Q001p	Zonhoven	50.991, 5.368	11657
P112p	Zoutleeuw	50.833, 5.103	2
Q179a	Zussen	50.798, 5.633	125
Q005p	Zutendaal	50.949, 5.591	2702

The following kloeke codes and their entries were omitted from the WLD because they were either more than 20 km removed from the border of Belgian Limburg, Dutch Limburg and Liège or they are incomplete and cannot be reliably linked to an existing kloeke code or are nonsensical (corrupted entries such as "(mv)", "HaspId"):

Q168p	Q020	L003p	Q000	K252p	Q111*	Q019q	O112p	K314	I030p	P198p	L368*	K036p
L200p	K286p	K282p	L318	L202p	K246p	L030p	P078p	P074p	L387	Q011	K257p	L005p
P004p	P001p	P075p	P073p	K289p	P005p	L036p	L084p	L331	L298b	Q066p	P067p	K326p
L024p	P129p	L020p	P064p	L037p	P199p	L033p	L038p	L0426	P002p	K259p	P092p	K037p
L377	Q016	L371	WestLb	L423	P008p	P015p	Q002	P060p	P060a	P020p	P080p	O178p
O118p	O113p	O119p	O116p	O165p	O179p	HaspId	L046p	K321p	K330p	K258p	P077p	L001p
L015p	L090p	P125p	P076p	L011p	P021p	Q160	O056p	L050p	L265	Q191	L310	K353
L035p	L434	L247	L293	Q156	Q205	Q205x	Q208	Q253	Q278	Q158	P048	L426
Q088	L321	L432	Q039	Q101	Q095	Q111	L269	L316	L332	K361	Q015	Q099
Q112	P071p	P086p	L013p	L349	L009p	L174	L014p	L008p	L048p	L016q	L042p	P176
P058	P219	Q078	L031p	L016p	(mv)	K163p	K163a	K164p	K165p	K211p	L043p	L023p
P013p	Q203e	K268p	L007p	L021p	P059p	P090p	K002p	L002p	P126p	P081p	O095p	O004p
L419	Q179	P214a	K120p	K176p	K177p	P083p	L022p	P128p	P019p	K329p	L360	K179p
K061p	P200p	L095p	K318	K279p								

Appendix C: WLD phonetic notation charts

Below we provide the original vowel and consonant charts (digitally edited and translated) as used in the WLD (Weijnen et al., 1983) for reference:

Consonant chart

	bilabial	labio-dental	dental, apical alveolar	palatal, prepalatal	velar, dorsal	glottal, laryngeal
occlusive explosive **voiceless**	*p* put gapen wippen stop web		*t* tam meute zetten buit rood		*k* kar beker rukken stok	*ʔ*
voiced	*b* beer Kobus krabben E. snob		*d* duur leden ladder E. mad		*q* D. geben sagen E. good smog F. garçon	
fricative approximant **voiceless**		*f* fiets rafel buffel roof	*s* som brasem kussen pees	*š* chocola D. schön Fisch E. shop wish F. cheval cache	*x* chroom kuchen lach leeg recht deugd	*h* hoe hebben ophouden
voiced		*v* vader zeven E. love	*z* zeer blozen mazzel E. prize	*ž* journaal ravage E. measure F. rouge	*g* geven wagen rogge vreugde	
nasal	*m* mee dame dammen stom		*n* niet leunen kunnen laan		*ŋ* lang longen D. Menge E. singing	
liquid			*r* rook leren sarren deur		*r* rook leren sarren deur	
lateral			*l* laag veulen vallen pijl			
semivowel	*w* wol dwang kwik twee zwaar	*w* wol dwang kwik twee zwaar		*j* jong jicht D. Jahr E. yes you		

Vowel chart

	back										medialized		front									
	rounded										unrounded		rounded								"spread"	
	monophthong				diphthong						diphthong	mono-phthong	monophthong		diphthong						diphthong	
	short		long			ending on a							kort	lang	vocalic paragoge as 2nd element	*w*-sound	*j*-sound		kort	lang	vocalic paragoge as 2nd element	ending on a
	palat.		velar			*w*-sound	*j*-sound															*w*-sound	*j*-sound
closed	*ú*		*ū*		*ʊə ʊᵊ*	*ʊɥ*	*ʊi*				*ʊ̈ə ʊ̈ɥ ʊ̈i*	*ʊ̈*	*ɥ*	*ʉ*	*ɥə ɥᵊ*	*ɥɥ*	*ɥi*		*i*	*ī*	*iə iᵊ*	*iɥ*	*ii*
	boek goed F. cou		boer F. jour D. gut E. flute D. Kuh				boci loeien						fuut nu nuzie F. lune D. Fülle	D. Bühne D. kühl	F. dur	duw duwen			diep kiem ieder lier knie F. qui D. siebzig	D. ihm D. wieder F. crise D. Knie E. key D. sieben		nieuw kieuwen	
half closed	*ó*		*ō*		*oə oᵊ* *oɥ* *ōə ōᵊ*	*oɥ* *ōɥ*	*oi* *ōi*				*ö̈ə ö̈ɥ ö̈i*	*ö̈*	*ø*	*ø̄*	*øə øᵊ* *øɥ* *øə øᵊ*	*øɥ* *øɥ*	*øi* *øi*		*e*	*ē*	*eə eᵊ* *eɥ* *eə eᵊ*	*eɥ* *ēɥ*	*ei* *ēi*
	kokom romein hotel konijn koraal		kool F. rôle room D. Dom D. rot zoon koor				*mooi* dooien												pin kil lip witten kirren mist	peen keel leep wetten keren meest zee D. See		leeuw eeuwen	
half open	*ǫ*		*ǭ*		*ǫə ǫᵊ* *ǫɑ* *ǭə ǭᵊ*	*ǫɥ*	*ǫi*		*ɔ*	*ɔ̄*	*ɔ̈ə ɔ̈ɥ ɔ̈i*	*ɔ̈*	*œ*	*œ̄*	*œə œᵊ* *œɛ* *ōə ōᵊ*	*œɥ* *ōɥ*	*œi* *ōi*		*ɛ*	*ɛ̄*	*ɛə ɛᵊ* *ɛɛ* *ɛə ɛᵊ*	*ɛɥ* *ɛ̄ɥ*	*ɛi* *ɛ̄i*
	ros zon vol dor hok F. coq		rose zone F. Sadne F. mort				hoi! D. neu E. voice						mud turf dun lus vullen	D. Köln F. neuf F. oeuf									
open	*á*		*ā*		*aə aᵊ*	*aɥ*	*ái*						*ø*	*ø̄*	*øə øᵊ* *øɛ* *øə øᵊ*	*øɥ* *øɥ*	*øi* *øi*		*ɛ*	*ɛ̄*	*ɛə ɛᵊ* *ɛɛ* *ɛə ɛᵊ*	*ɛɥ* *ɛ̄ɥ*	*ɛi* *ɛ̄i*
	klasje D. dass oranje F. cabane maije D. Ball		Klaas kaam baan kaar maat baal				ai! D. Zeit F. travail haai zaaien																
	a		*ā*		*aə aᵊ*	*aɥ*	*ái*						*ə*	*ə̄*	*əə əᵊ* *əɛ* *əə əᵊ*	*əɥ* *əɥ*	*əi* *əi*		*ɛ*	*ɛ̄*	*ɛə ɛᵊ* *ɛɛ* *ɛə ɛᵊ*	*ɛɥ* *ɛ̄ɥ*	*ɛi* *ɛ̄i*
	klas kam ban kar mat bal		klasje D. dass oranje F. cabane maije D. Ball																				

"colorless" vocal
in unstressed words or syllables

ə

de, **je**, **me**, **te**, **ze**
gemoet, **rafelen**
F. de, **je**, **me**, **le**, **que**

Appendix D: Manual preprocessing rules

The following rules are executed sequentially during the cleaning part of the preprocessing pipeline:

Condition	Action	Reason
last character = !	remove last character !	superfluous (used as exclamation)
contains "	" replace by ø	transcription error
contains #	&#x0304 replace by ¯	transcription error (unicode)
contains &	remove	superfluous (used as conjunction)
contains (sic.), (mv.), (m.), (v.), (.)	remove	superfluous (marks gender or plural)
contains (gew.uitspr.)	remove	superfluous
contains (,)	remove	superfluous (optional parts of word)
last character = *	remove	superfluous
contains *	remove entry	too noisy
contains ,	remove	superfluous (alternative or in sentences)
entire entry = /	remove	nonsensical
contains /	remove	superfluous (used for alternatives)
contains digit	remove digit	not phonetically relevant
contains ;	remove entry	too difficult to resolve, only 112 entries
contains ?	remove	superfluous (used for uncertainty or question)
contains […]	remove	superfluous (used for semantic meaning)
contains [,]	remove	superfluous
contains ∧	remove	too noisy
contains `t	`t replace by ət	transcribe phonetically
contains `n	`n replace by ən	transcribe phonetically
first character = `	remove	transcription error
last character = `	remove	transcription error
contains {, }	remove	superfluous
contains ~	remove entry	too noisy, only 196 entries
contains ->	remove	too noisy, superfluous
contains <, >	remove	superfluous
contains 'n	'n replace by ən	transcribe phonetically
contains 't	't replace by ət	transcribe phonetically
contains d'r	d'r replace by dər	transcribe phonetically
contains '	remove	superfluous
contains /t	/t replace by ət	transcribe phonetically
contains /n	/n replace by ən	transcribe phonetically
contains /m	/m replace by əm	transcribe phonetically
contains /	remove entry	too noisy, 2759 entries
contains ∂	remove entry	too noisy, 189 entries
contains g not used as gk, kg, gg, ng	g replace by x	transcribe phonetically
word is empty or only space	remove entry	nonsensical

We note that the fourth last step loses particularly many entries (2759/8982) but was not further improved upon due to time constraints.

Bibliography

T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Apple Machine Learning Research: Natural Language Processing Team. Language Identification from Very Short Strings. `https://machinelearning.apple.com/research/language-identification-from-very-short-strings`, 2023.

N. Arivazhagan and et al. Massively Multilingual Neural Machine Translation in the Wild: Findings and Challenges, 2019.

B. Assendelft. De codificatie van het Limburgs. *Taal en Tongval*, 71(1):1–30, 2019. ISSN 2215-1214.

F. Bakker. Wat is Limburgs? *Onze Taal*, 66, 1997.

F. Bakker. *Waar scheiden de dialecten in Noord-Limburg?* LOT, 2016. ISBN 978-94-6093-217-5.

P. Bakkes, R. Belemans, G. Cornelissen, R. Keulen, T. Van de Wijngaard, H. Crompvoets, and F. Walraven. Riek van klank: inleiding in de Limburgse dialecten, 2007. ISSN 978-90-8596-034-8.

P. Bakkes, H. Crompvoets, J. Notten, and F. Walraven. Spelling 2003 voor de Limburgse dialecten. `https://www.veldeke.net/wp-content/uploads/2020/12/Spelling-2003.pdf`, 2023.

I. Balazevic, M. Braun, and K. Müller. Language Detection For Short Text Messages In Social Media, 2016.

R. Belemans and R. Keulen. *Belgisch-Limburgs.* Taal in stad en land. Lannoo, 2004. ISBN 9789020958553.

V. Blaschke, H. Schütze, and B. Plank. A survey of corpora for Germanic low-resource languages and dialects. In *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 392–414, Tórshavn, Faroe Islands, 2023. University of Tartu Library.

R. Botarleanu and et al. Age of Exposure 2.0: Estimating word complexity using iterative models of word embeddings. *Behavior Research Methods*, 54, 02 2022. doi: 10.3758/s13428-022-01797-5.

A. Conneau and et al. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451. Association for Computational Linguistics, 2020.

L. Cornips, V. de Rooij, I.L Stengs, and L. Thissen. *Dialect and local media: Reproducing the multi-dialectal hierarchical space in Limburg (the Netherlands)*, pages 189–216. Novus Press, 2016.

L. Cornips, J. Klatter-Folmer, T. Schils, and R. Roumans. *Cornips, L., Klatter-Folmer, J., Schils, T., Roumans, R. (2022). A Longitudinal Comparison of Spelling and Reading Comprehension of Bidialectal and Monolingual Dutch Speaking Children in Primary School. In: Saiegh-Haddad, E., Laks, L., McBride, C. (eds) Handbook of Literacy in Diglossia and in Dialectal Contexts. Literacy Studies, vol 22. Springer, p. 219–245. ISBN978-3-030-80071-0*, pages 219–245. 2022. doi: 10.1007/978-3-030-80072-7_11.

H. Crompvoets. *Klank- en woordgeografie rond Venlo.* 1998.

H. Crompvoets and H. van de Wijngaard. *Woordenboek van de Limburgse dialecten, Deel 2. Niet-agrarische vakterminologieën: Aflevering 4. Turfsteker en ertsontginner.* Van Gorcum & Comp., 1987.

H. Crompvoets and H. van de Wijngaard. *Woordenboek van de Limburgse dialecten, Deel 2. Niet-agrarische vakterminologieën: Aflevering 5. Mijnwerker.* Van Gorcum & Comp., 1989.

R. Dabre, C. Chu, and A. Kunchukuttan. A Brief Survey of Multilingual Neural Machine Translation, 2020.

J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019.

W. Dols. *Iets over Limburgsche dialecten*, pages 2–4. Vereniging voor Limburgse Dialect- en Naamkunde, Veldeke Limburg, 1946.

M. Elaraby and M. Abdul-Mageed. Deep Models for Arabic Dialect Identification on Benchmarked Data. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pages 263–274, Santa Fe, New Mexico, USA, 2018. Association for Computational Linguistics.

Hans Erren. Der Rheinische Fächer. `https://commons.wikimedia.org/wiki/File:Rheinischer_faecher.png`, 2010.

Ethnologue. Languages of the World. `https://www.ethnologue.com`. Accessed: 04/07/2023.

Eurostat. NUTS. `https://ec.europa.eu/eurostat/web/gisco/geodata/reference-data/administrative-units-statistical-units/nuts`, 2023. Accessed: 08/06/2023.

Hugging Face. XLM-RoBERTa. `https://huggingface.co/xlm-roberta-base`, a. Accessed: 12/07/2023.

Hugging Face. BERT multilingual base model (cased). `https://huggingface.co/bert-base-multilingual-cased`, b. Accessed: 12/07/2023.

Rachel Fournier. Perception of the tone contrast in East Limburgian dialects. 2008.

K. Franco, D. Geeraerts, D. Speelman, and R. van Hout. Concept characteristics and variation in lexical diversity in two Dutch dialect areas. *Cognitive Linguistics*, 30(1):205–242, 2019a.

K. Franco, D. Geeraerts, D. Speelman, and R. van Hout. Maps, meanings and loanwords: The interaction of geography and semantics in lexical borrowing. *Journal of Linguistic Geography*, 7(1):14–32, 2019b.

Y. Goldberg. A Primer on Neural Network Models for Natural Language Processing, 2015.

J. Goossens. *Die Gliederung des Südniederfränkischen*, pages 79–94. 1965.

J. Goossens. *Woeringen en de oriëntatie van het Maasland*. 1988. ISBN 9789090025469.

J. Goossens. *Bijdrage tot de woordstratigrafie van de Limburgia Romana*, pages 352–362. 1996.

K. Goswami, R. Sarkar, B. R. Chakravarthi, T. Fransen, and J. P. McCrae. Unsupervised Deep Language and Dialect Identification for Short Texts. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1606–1617, Barcelona, Spain, 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.141.

C. Gussenhoven and F. Aarts. The dialect of Maastricht. *Journal of the International Phonetic Association*, 29(2):155–166, 1999. ISSN 00251003, 14753502.

T. L. Ha, J. Niehues, and A. Waibel. Toward Multilingual Neural Machine Translation with Universal Encoder and Decoder, 2016.

M. P. Harper and M. Maxwell. *Springer Handbook of Speech Processing*, pages 797–809. Springer Handbooks. Springer, Berlin, 2008. ISBN 978-3-540-49125-5. doi: 10.1007/978-3-540-49127-9.

J. Heaton. *Introduction to Neural Networks for Java, 2nd Edition*. Heaton Research, Inc., 2nd edition, 2008. ISBN 1604390085.

L. Heijmans and C. Gussenhoven. The Dutch dialect of Weert. *Journal of the International Phonetic Association*, 28(1/2):107–112, 1998. ISSN 00251003, 14753502.

B. J. H. Hermans. *Phonological features of Limburgian dialects.*, pages 336–356. Walter De Gruyter, Germany, 2013.

D. Hitch. Distinctive vowel heights in Limburgish and Bavarian. *Taal en Tongval*, 11 2020. doi: 10.5117/TET2020.2.HITC.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

P. Honnet, A. Popescu-Belis, C. Musat, and M. Baeriswyl. Machine Translation of Low-Resource Spoken Dialects: Strategies for Normalizing Swiss German. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, 2018. European Language Resources Association (ELRA).

J. I. Hualde. *Los sonidos del español: Spanish Language edition*. Cambridge University Press, 2013. doi: 10.1017/CBO9780511719943.

Hugging Face. The AI community building the future. `https://huggingface.co`. Accessed: 12/07/2023.

Meertens Instituut. Kloekecodes opzoeken. `https://www.meertens.knaw.nl/kloeke`. Accessed: 08/06/2023.

T. Ismail. A Survey of Language and Dialect Identification Systems. 9, 2020.

T. Jauhiainen, M. Lui, M. Zampieri, T. Baldwin, and K. Lindén. Automatic Language Identification in Texts: A Survey, 2018.

T. Jauhiainen, H. Jauhiainen, T. Alstola, and K. Lindén. Language and Dialect Identification of Cuneiform Texts. In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 89–98, Ann Arbor, Michigan, 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-1409.

M. Johnson and et al. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation, 2017.

L. Jongbloed-Faber, J. Loo, and L. Cornips. Regional languages on Twitter: A comparative study between Frisian and Limburgish. *Dutch Journal of Applied Linguistics*, 6:174–196, 2017. doi: 10.1075/dujal.16017.jon.

E. Keuleers, M. Brysbaert, and B. New. SUBTLEX-NL: A new measure for Dutch word frequency based on film subtitles. *Behavior research methods*, 42:643–50, 08 2010. doi: 10.3758/BRM.42.3.643.

P. Klein. *De aanvoegende wijs (conjunctief) in de dialecten van Zuidoost-Limburg*, pages 81–98. 2020.

T.J.W.M. Kruijsen. *Een eeuw lang Limburgs. SGV-enquête 1914 - Veldeke 2006*. Uitgeverij TIC & Veldeke Limburg, 2006. ISBN 90-78407-06-9.

T.J.W.M. Kruijsen, R. van Hout, and Swanenberg J. *Nieuwerwets omgaan met oude gegevens. De SGV-enquête op internet*, pages 877–896. Akademia Press, 2004.

P. Ladefoged and S.F. Disner. *Vowels and Consonants*. Wiley, 2012. ISBN 9781444334296.

G. Lample and A. Conneau. Cross-lingual Language Model Pretraining, 2019.

J. Leenen. *Franse taaluitzetting over Limburg*, pages 149–166. 1938.

J. Leenen, S. van der Meer, and W. Roukens. *Limburgse klankgrenzen*. Amsterdam, 1947.

Limburgish Academy. Limburgish Language. `https://limburgs.org/en/limburgish`, a. Accessed: 15/07/2023.

Limburgish Academy. Limburgish keyboard. `https://limburgs.org/en/keyboard`, b. Accessed: 15/07/2023.

N. F. Liu, M. Gardner, Y. Belinkov, M. E. Peters, and N. A. Smith. Linguistic Knowledge and Transferability of Contextual Representations. 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)*, pages 1073–1094, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1112.

Y. Liu and et al. Summary of ChatGPT/GPT-4 Research and Perspective Towards the Future of Large Language Models, 2023.

M. Lui and P. Cook. Classifying English Documents by National Dialect. In *Australasian Language Technology Association Workshop*, 2013.

J. Luo, V. G. Li, and P. P. K. Mok. The Perception of Cantonese Vowel Length Contrast by Mandarin Speakers. *Language and Speech*, 63:635–659, 2020.

LVR-Institut für Landeskunde und Regionalgeschichte. Sprechende Sprachkarte. `https://rheinische-landeskunde.lvr.de/de/sprache/sprechende_sprachkarte/konzept_1/detailseite_303.html`. Accessed: 12/07/2023.

Meertens Instituut. Cartography: Kloeke. `https://projecten.meertens.knaw.nl/mand/ECARTkartografiefie.html`. Accessed: 08/06/2023.

Y. Michielsen Tallman, L. Lugli, and M. Schuler. A Limburgish Corpus Dictionary: Digital Solutions for the Lexicography of a Non-standardized Regional Language. 2017.

A. Millour and K. Fort. Unsupervised Data Augmentation for Less-Resourced Languages with no Standardized Spelling. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 776–784, Varna, Bulgaria, 2019. INCOMA Ltd. doi: 10.26615/978-954-452-056-4_090.

A. Moors and et al. Norms of valence, arousal, dominance, and age of acquisition for 4,300 Dutch words. *Behavior research methods*, 09 2012. doi: 10.3758/s13428-012-0243-8.

D. R. Mortensen, P. Littell, A. Bharadwaj, K. Goyal, C. Dyer, and L. S. Levin. PanPhon: A Resource for Mapping IPA Segments to Articulatory Feature Vectors. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3475–3484. ACL, 2016.

C. Moseley. *Atlas of the World's Languages in Danger*. Memory of peoples Series. UNESCO Publishing, 2010. ISBN 9789231040962.

B. Muller, B. Sagot, and D. Seddah. Can Multilingual Language Models Transfer to an Unseen Dialect? A Case Study on North African Arabizi, 2020.

Nomatim. Nomatim. `https://nominatim.openstreetmap.org/ui/search.html`. Accessed: 08/06/2023.

OpenAI. ChatGPT by OpenAI. `https://openai.com`, 2021. Accessed: 12/07/2023.

S. Parida, E. Villatoro-Tello, S. Kumar, P. Motlicek, and Q. Zhan. Idiap Submission to Swiss-German Language Detection Shared Task. In *Proceedings of the 5th Swiss Text Analytics Conference (SwissText) 16th Conference on Natural Language Processing (KONVENS)*. CEUR Workshop Proceedings, 2020.

J. L. Pauwels and L. Morren. De Grens Tussen het Brabants en het Limburgs in België. *Zeitschrift für Mundartforschung*, 27(2):88–96, 1960.

M. Popović. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, 2015. Association for Computational Linguistics. doi: 10.18653/v1/W15-3049.

Marine Regions. Marine gazetteer placedetails. `https://www.marineregions.org`. Accessed: 18/07/2023.

W. Roukens. *Wort- und Sachgeographie in Niederländisch-Limburg und den benachbarten Gebieten mit bes. Berücks. d. Volkskundlichen: Atlas*. de Gelderlander, 1937.

W. Roukens. Het Limburgs Woordenboek. *Veldeke*, 35:28–29, 1960.

Y. Scherrer and N. Ljubeić. Automatic normalisation of the Swiss German ArchiMob corpus using character-level machine translation. In *Conference on Natural Language Processing*, 2016.

Y. Scherrer and O. Rambow. Word-Based Dialect Identification with Georeferenced Rules. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1151–1161, Cambridge, MA, 2010. Association for Computational Linguistics.

J. Schrijnen. *Taalgrenzen in Zuidnederland - het mich-kwartier*, pages 2–3. Vereniging voor Limburgse Dialect- en Naamkunde, Veldeke Limburg, 1907.

J. Schrijnen, J. van Ginneken, and J.J. Verbeeten. Enquête Schrijnen-Van Ginneken-Verbeeten (SGV), 1914.

I. Semenov. Wikipedia word frequency generator. `https://github.com/IlyaSemenov/wikipedia-word-frequency`. Accessed: 15/07/2023.

A. Shoufan and S. Alameri. Natural Language Processing for Dialectical Arabic: A Survey. In *Proceedings of the Second Workshop on Arabic Natural Language Processing*, pages 36–48, Beijing, China, 2015. Association for Computational Linguistics. doi: 10.18653/v1/W15-3205.

A. Shrestha and A. Mahmood. Review of Deep Learning Algorithms and Architectures. *IEEE Access*, 7: 53040–53065, 2019. doi: 10.1109/ACCESS.2019.2912200.

V. Simaki, P. Simakis, C. Paradis, and A. Kerren. Identifying the Authors' National Variety of English in Social Media Texts. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 671–678, Varna, Bulgaria, 2017. INCOMA Ltd. doi: 10.26615/978-954-452-049-6_086.

B. Spaan. `https://github.com/erfgoed-en-locatie/data/blob/master/kloeke/hg_kloeke.csv`, 2015.

Y. Tay, M. Dehghani, D. Bahri, and D. Metzler. Efficient Transformers: A Survey, 2022.

M. Toftrup, S. A. Sørensen, M. R. Ciosici, and I. Assent. A reproduction of Apple's bi-directional LSTM models for language identification in short strings. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 36–42, Online, 2021. Association for Computational Linguistics.

A. L. Tonja, M. G. Yigezu, O. Kolesnikova, M. S. Tash, G. Sidorov, and A. Gelbuk. Transformer-based Model for Word Level Language Identification in Code-mixed Kannada-English Texts, 2022.

Unidecode. Unidecode. `https://pypi.org/project/Unidecode`. Accessed: 15/07/2023.

S. Van Hoof and J. Jaspers. Hyperstandaardisering. *Tijdschrift voor Nederlandse Taal- en Letterkunde*, 128 (1):97–125, 2012.

R. van Hout, N. van der Sijs, E. Komen, H. van den Heuvel, and et al. Elektronisch Woordenboek van de Limburgse Dialecten (e-WLD). `https://www.e-wld.nl`. Accessed: 08/06/2023.

R. van Hout, N van der Sijs, E. Komen, and H. van den Heuvel. A Fast and Flexible Webinterface for Dialect Research in the Low Countries. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA), May 2018.

V. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need, 2017.

Veldeke. De stand van het Limburgs. `https://www.veldeke.net/de-stand-van-het-limburgs`, 2021. Accessed: 12/07/2023.

R. Weijenberg. Mestreechter Taol. `https://www.mestreechtertaol.nl`. Accessed: 12/07/2023.

A. Weijnen and F. Van Coetsem. *De rijksgrens tussen België en Nederland als taalgrens*. Noord-Hollandsche uitgeversmaatschappij, 1957.

A. Weijnen, J. Goossens, and P. Goossens. *Woordenboek van de Limburgse dialecten, Deel 1. Agrarische terminologie: Inleiding en Aflevering 1. Bemesten en ploegen: mijnwerker*. Van Gorcum & Comp., 1983.

L. Wintgens. Plurilinguisme et idéologies linguistiques au «Pays sans frontières». 66:85–96, 1994.

T. Wolf and et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6.

M. Zampieri, L. Tan, N. Ljubešić, and J. Tiedemann, editors. *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, Dublin, Ireland, 2014. Association for Computational Linguistics and Dublin City University. doi: 10.3115/v1/W14-53.