Master Thesis in Computer Science

Foundational Language Models for Ultra-low Resource Languages
The Case of Faroese

Author

Rói Olsen
rools20@student.sdu.dk

Supervisor:	**Peter Schneider-Kamp** Professor, Dept. of Mathematics and Computer Science University of Southern Denmark
External Supervisor:	**Henrik Hoffmann Nielsen** R&D Responsible, ODIN Ordbogen A/S

June 1, 2025

Department of Mathematics and Computer Science
Faculty of Science, University of Southern Denmark, Campus Odense

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

Abstract

some text
some more text
noget tekst på dansk

og noget mere, men med mellemrum først

endnu mere tekst der er meget langt og fylder meget mere end de andre tekster, og går over en linje eller to

Table of Contents

List of Figures

List of Tables

List of Snippets

Glossary

Bendingargrunnurin **– The Inflection Database** 6, 7

BERT **– Bidirectional Encoder Representations from Transformers** 2, 3

bitext **– billingual text** 7

Doc **– spaCy Doc** 10

DocBin **– spaCy DocBin** 10

Fróðskaparrit **– Faroese Scientific Journal** 5

Fróðskapur **– Faroe University Press** 6

GPT **– Generative Pre-trained Transformer** 2

LSTM **– Long Short-Term Memory** 2

Málráðið **– The Faroese Language Council** 7

MORPH **– Morphologizer** vii, 8, 10, 14

mT5 **– Multilingual Text-to-Text Transfer Transformer** 2, 3

MTD **– The Faroese Centre for Language Technology** iv, 5, 9

NLLB **– No Language Left Behind** iv, 7

NLP **– Natural Language Processing** 2, 5

POS **– Part-of-Speech** v, vii, 5, 6, 8, 10, 13, 14

RNN **– Recurrent Neural Network** 2

Rættstavarin **– The Spell Checker** 6

UD **– Universal Dependencies** iv, 6, 8

Chapter 1

Introduction

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat.

```
def hello_world():
    print("Hello, World!")


class MyClass:
    def __init__(self, name):
        self.name = name

    def greet(self):
        print(f"Hello, {self.name}!")
```

name	age	city
John	30	New York
Jane	25	Los Angeles
Doe	35	Chicago
Smith	40	Houston

Chapter 2

Scientific Background

This section provides an overview of the background information necessary to understand the context of the study. It covers the key concepts and relevant research that have shaped the current state of the field.

2.1 Transformers

Transformers are a class of deep learning models introduced by (Vaswani et al., 2017) that revolutionized Natural Language Processing (NLP) and other sequential data tasks. Unlike Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM), which process input sequentially, transformers leverage a self-attention mechanism that allows for parallelization and long-range dependency modeling. At the core of the transformer architecture is the self-attention mechanism, which enables the model to weigh the importance of different words in a sequence, regardless of their position. This is complemented by positional encodings, which provide information about word order, addressing the lack of inherent sequentiality in self-attention. The transformer is composed of stacked encoder and decoder layers, each containing multi-head self-attention and feedforward layers with residual connections and layer normalization. Transformers have been the foundation for state-of-the-art NLP models, including Bidirectional Encoder Representations from Transformers (BERT), Multilingual Text-to-Text Transfer Transformer (mT5) and perhaps the best known transformer to the general public, Generative Pre-trained Transformer (GPT). These architectures have been widely applied in text generation, translation, and classification tasks, as well as in domains such as computer vision, protein structure prediction, and reinforcement learning. The scalability and effectiveness of transformers have driven their adoption across various fields, making them a cornerstone of modern deep learning research.

2.1.1 Self-attention

(Vaswani et al., 2017)

2.1.2 mT5

mT5 is a pre-trained language model developed by Google. It is based on the Transformer architecture and is designed for text-to-text transfer learning.

2.1.3 BERT

BERT is a pre-trained language model developed by Google.

2.1.4 Grammar Correction

2.1.5 Spelling

There are 2 categories of spelling errors, the first is a typographical error, the second is a cognitive error. Typographical errors are made my mistyping a word, while cognitive errors are made by a lack of understanding. A spelling error in the context of this study is an error that results in a word that is not in the faroese dictionary.

Typographical Errors

Typographical errors can be split into 3 subcategories:
Transposition errors can be made by transposing two adjacent characters in a word.
Substitution errors can be made by substituting a character in a word.
Insertion errors can be made by inserting a character into a word.
Omission errors can be made by omitting a character in a word.

Omission errors are the simplest to make, since they only require removing a character from the original word. Transposition errors are also very simple to make, since they only require permutation of the original word, by swapping a pair of adjacent characters. Substitution and Insertion errors require a bit more thought. If characters are picked completely at random, most errors would not be realistic typos and it would result in most generated spelling errors being redundant. For a letter like **G** only 6/29 typos would be useful and a letter like **Q** would only have 3/29 useful errors. To make the errors more realistic, each character that is substituted or inserted is picked from adjacent characters on the keyboard. For example replacing a **G** with an **H** is a probable typo to make when writing on a keyboard, since the h key is right next to the **G** key. The same goes for inserting a character, if a character is inserted, it is picked from the adjacent characters on the keyboard. The keyboard layout used is the faroese keyboard layout, which is a modified version of the danish keyboard layout. The only relevant difference between the two layouts is that the faroese keyboard layout has the **Đ** character to the right of **Å**. The other difference is in punctuations and modifiers.

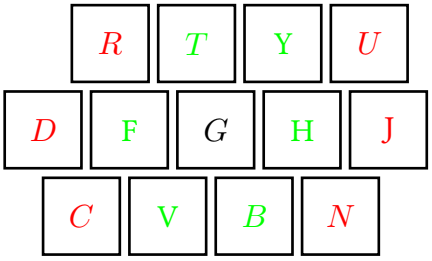

Figure 1: Possible typographical errors that can be made for G are marked with green and the red are too far away from the correct key to be considered a probable typo.

Cognitive Errors

Theses errors are a bit more difficult to categorize than typographical errors, as the reason for the error is not always clear. Most spelling errors are phonetic spelling errors, where a word is spelled as it sounds, this can have many different causes. The most common phonetic errors are errors with **ð**. In most cases it is a silent letter, which often leads to it being omitted. When **ð** is pronounced it is often pronounced as a **v** which leads to a spelling error where **v** is written in place of a **ð**. A type of spelling errors that is both phonetic and because of an exception to a rule, is a vowel followed by a double consonant is short, and a long vowel is followed by a single consonant. This rule has some exceptions like in the words **skula** and **fram**, the first **u** in skula and the **a** in fram are short, but the words are spelled with a single consonant, which causes people to spell them like **skulla** and **framm**. Dialects will also often lead to phonetic spelling errors. The faroese language has many dialects, and some of them have different pronunciations of the same word. The northern dialect, such as the one in Klaksvík, pronounces **á** as **a**, which leads to spelling errors where **á** is written as **a**. And someone with a southern dialect, might pronounce **p** more softly than someone with a northern dialect, which leads to spelling errors where **p** is written as **b**.

Chapter 3

Materials

This section describes the materials used in the study, including the datasets, tools, and resources that were essential for conducting the experiments and analyses.

3.1 spaCy

Spacy is an open-source software library for advanced NLP in Python. It is designed specifically for production use and is widely used in industry. Spacy provides a wide range of features for processing text data, including tokenization, part-of-speech tagging, named entity recognition, and dependency parsing. It also includes pre-trained models for a variety of languages and domains, making it easy to get started with NLP tasks. The relevant features of spaCy for this thesis are the POS tagger and morphologizer and to a lesser degree the lemmatizer and dependency parsing capabilities.

3.2 Language Resources From MTD

MTD Has released a number of resources for the Faroese language. A Github repository (*Faroese Language Resources by Fonlp*, n.d.) contains a collection of datasets and models (*Faroese Language Resources by the Centre for Language Technology*, n.d.).

A private corpus developed by Uni Johannesen was used as a testset to evaluate the performance of the grammar and spelling models. The corpus contains ~2700 sentences and consists of manually corrected and annotated essays from students. The essays were written in Faroese and contain a variety of errors, including spelling, grammar, and punctuation errors. Additionally the data used in the study by (Katrin Næs, 2005) on spelling errors in faroese students' essays, was also used in this study.

3.3 Faroe University Press Papers And Books

Faroese Scientific Journal (Fróðskaparrit) has published a number of papers and books in the various fields. The file format of the papers and books is pdf, so some preprocessing is needed to get the text out of them. All publications can be found for free on their website. The papers are from Fróðskaparrit. Fróðskaparrit is an annual journal with scientific articles from and about the Faroe Islands and Faroese issues. The journal spans all scientific fields with articles in Faroese (mostly Humanities and Social Sciences)

or English (Natural and Life Sciences). The books are from Faroe University Press (Fróðskapur). Fróðskapur was established in 2005. It publishes works from all scientific fields in the Faroe Islands. Publications are in Faroese, English and Danish. For this study only the faroese publications have been used.

3.4 Universal Dependencies

UD is a framework for cross-linguistic grammatical annotation designed to provide a consistent syntactic and morphological analysis across a wide variety of languages. It is based on a universal set of dependency relations and POS tags, enabling multilingual parsing and linguistic comparison. The UD framework represents syntactic structure using dependency trees, where words have labels that define their grammatical relationships. These annotations follow a principle of typological neutrality, ensuring that the framework remains applicable across languages with diverse syntactic structures. The two faroese UD datasets used in this study are the (*Faroese-OFT Treebank*, n.d.) and (*Faroese-Farpahc Treebank*, n.d.). Both datasets contain manually annotated sentences in Faroese.

(*Faroese-OFT Treebank*, n.d.) is based on sentences from the Faroese Wikipedia. The whole Wikipedia was analysed using (*Trond Trosterud's Tools for Faroese*, n.d.). Sentences that contained unknown words were removed. The remaining sentences were manually annotated for Universal Dependencies and the morphology and POS tags were converted deterministically using a lookup table. Errors in the original morphology and disambiguation were corrected where found. The treebank contains a lot of copula sentences and very little first or second person, as can be expected from Wikipedia texts.

(*Faroese-Farpahc Treebank*, n.d.) is a conversion of the (*Faroese Parsed Historical Corpus (Farpahc)*, n.d.) to the Universal Dependencies scheme using (*Udconverter*, n.d.). (*Faroese Parsed Historical Corpus (Farpahc)*, n.d.) is a 53,000 word corpus which includes three texts from the 19th and 20th centuries. These texts were originally manually parsed according to the (*Penn Parsed Corpora of Historical English*, n.d.) annotation scheme. Two of these parsed texts have been automatically converted to the Universal Dependencies scheme to create the 40,000 word (*Faroese-Farpahc Treebank*, n.d.).

3.5 Bendingar.fo

Bendingar.fo is a website, that hosts The Inflection Database (Bendingargrunnurin) (*Bendingarunnurin*, n.d.). Bendingargrunnurin contains inflections, lemmas and morphological data for faroese words and names. Most words are taken from wordlists that were made for The Spell Checker (Rættstavarin) (*Rættstavarin*, n.d.), that were taken from a faroese dictionary (Jóhan Hendrik W. Poulsen, 1998). The names are

from the name list from The Faroese Language Council (Málráðið) (*Bendingarunnurin*, n.d.). The project to make Bendingargrunnurin was started in 2021-22, and was called Insular Nordic morphological database project and got funds from Nordplus Nordic Language (NPLA-2021/10025).

3.6 NLLB

The faroese dataset consists of multiple types of data primary billingual text (bitext), mined bitext and monolingual Text. For faroese, the datasets consist of primary (*NLLB - Faroese - Primary*, n.d.) and mined bittext (*NLLB - Faroese - Mined*, n.d.). The primary bitext corpora are publicly available parallel corpora from a variety of sources. The mined bitext corpora are retrieved by large-scale bitext mining. The mined data includes all the English-centric directions and a subset of non-English-centric directions. Only the faroese data is used in this study.

Chapter 4

Data Processing

This chapter describes the data processing steps involved in preparing the data for training the models. The data processing pipeline includes data collection, cleaning, preprocessing, and augmentation. Each step is essential for ensuring the quality and integrity of the data used for training the models. The following sections provide a detailed overview of the data processing pipeline used in this study.

4.1 Data Collection

All the training data was from various online repositories and websites. The training data is comprised of Faroese text from various sources, including articles from wikipedia, news websites, and research papers, as well as social media posts, legal documents, posts from government institutions and books. The data is available online and can be accessed through the respective websites. The formats of the data vary depending on

the source, and the data was collected in the form of text files, CSV, json, jsonl, html, xml and pdf, so some preprocessing is needed to get it in a uniform format.

4.2 Data Cleaning & Preprocessing

The unlabeled data was cleaned using a mix of heuristics. To remove a lot of the foreign sentences, a blacklist of foreign characters was used to filter out sentences that contained these characters. Additionally a list of common danish and english words were used to remove foreign sentences. To get rid of some metadata, words and abbreviations like **img src**, **aspx**, **pid**, **newsid**, **html**, **date** were used to remove the sentences they occurr in. Due to encoding errors in the data, a lot of the data was wrongly converted to ascii, but a lot of it could be reverse engineered by manually inspecting the data, and from context, get a mapping from the wrong encoding to the correct faroese character. For example the character **ð** was written as **Ã°** and the character **á** was written as **Ã¡** and so on. The **ð** character is also sometimes written as **ƺ** or **đ**, so to make the dataset more uniform, they are converted to **ð**, which is the only one of them that can be written with a faroese keyboard without modifiers. The data was also cleaned by removing any html tags, and any other non-faroese characters. Some of the data has really long sentences, some of them up to 800.000 characters long, so they were split on the period character, excluding periods from abbreviations. All duplicate sentences were removed from the dataset. The dataset was shuffled to make sure the model doesn't overfit on the order of the data. The unlabeled dataset was saved as jsonl for pretraining of the mt5 model and as a txt file for further processing. The txt file was tokenized and saved as a doc in the spaCy format, by tokenizing the text, before corrupting it, a lot of time is saved in the corruption process. The corruption process will be covered in the data augmentation section.

4.2.1 UD Data

The labeled data from the faroese UD repositories required minimal processing, it was saved in a single json file for easier inspection. Then all duplicate sentences were removed from the dataset. The json file was the converted to the spaCy format, which is a binary format used to train spacy models. It consists of a list of docs, where each doc contains sentences that are labeled, depending on what model you train. In this case, there were a few different files, the majority of them only had POS and morphologizer labels, but some of them also had dependency labels, and some of them had lemmatizer labels. The files with POS and MORPH labels, had 6652 labeled faroese sentences, which add up to 9.3 MiB of data. The files with lemma labels had 1428 sentences, which added up to 756 KiB of data. And lastly the files with dependency parser labels had 3049 sentences which added up to 2.8 MiB of data. Each of the files was split into a training and a validation set, where 95% of the data was used for training and 5% was used for validation.

4.2.2 Private Corpus from MTD

The original corpus is in xml format where each sentence is a separate xml tag and each word and punctuation is a tag in the sentence. If a word or punctuation is corrected, a revision tag replaces the error and contains the original and corrected text.

Data Snippet 1: An example of how a sentence in the private corpus is formatted (this is not a real sentence from the corpus)

```
<s n="1">
    <w>Vit</w>
    <revision id="15">
        <original>
            <w>fóru</w>
        </original>
        <corrected>
            <w>fara</w>
        </corrected>
        <errors>
            <error xtype="past4pres" eid="0" />
        </errors>
    <revision id="16">
        <original>
            <w>niðaná</w>
        </original>
        <corrected>
            <w>niðan</w>
            <w>á</w>
        </corrected>
        <errors>
            <error xtype="adv4adv-prep" eid="0" />
        </errors>
    </revision>
    <w>fjallið</w>
    <c>.</c>
</s>
```

For each error in a sentence a pair of incorrect and correct sentences are generated. The incorrect sentence contains the error and the correct sentence is the same as the incorrect sentence, but with the error corrected. Figure 2 shows the sentences that are generated from the sentence in Data Snippet 1.

Vit $\left(\begin{smallmatrix}\text{fóru}\\ \text{fara}\end{smallmatrix}\right)$ $\left(\begin{smallmatrix}\text{niðaná}\\ \text{niðan á}\end{smallmatrix}\right)$ fjallið í morgin.

⬇

Vit fóru niðan á fjallið í morgin. Vit fara niðan á, fjallið í morgin.
Vit fara niðaná fjallið í morgin. Vit fara niðan á fjallið í morgin.

4.3 Data Augmentation

The data is augmented by taking correct text and corrupting it. The corruption process is done by using a list of rules, that are applied to the correct sentence. The types of errors are ordered in a hierarchy, where a category can directly have errortypes or have subcategories. A subcategory has errors. The hierarchy is a way to organize the errors, so that the corruption process can be more precise where possible and in the cases where an error could belong to multiple error types, it is defined which error type has higher priority.

4.3.1 Corruption Process

Before the corruption process can begin, the text is tokenized and annotated with POS and MORPH tags and saved in a spaCy Doc (Doc). The Doc is then put in a spaCy DocBin (DocBin) which is serialized and written to a file. The first step in the corruption is to generate a json file that contains every corruption that can be made. Before distributing the corruptions into a training set, a distribution file, which is a json file that specifies the proportion of each errortype to introduce into the dataset, is used. The json file is then loaded and the corruptions are applied, with the proportions specified in the distribution file, to the text. The corrupted dataset is then shuffled to make sure the model doesn't overfit on the order of the data.

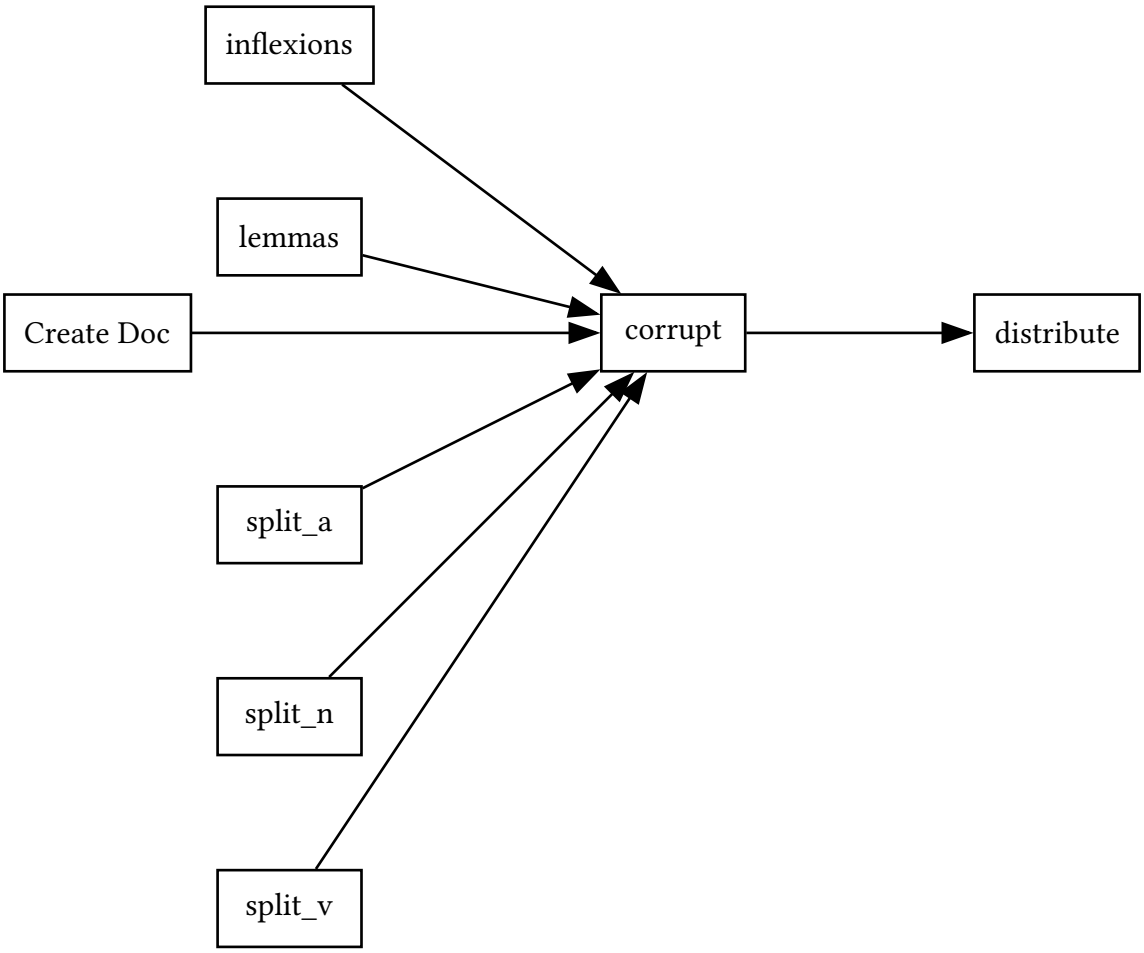

Figure 3: An overview of the corruption process used for data augmentation.

Grammar

A grammar error is an error that violates the rules of grammar. This means all the words in the sentence are spelled correctly, but the sentence is still grammatically incorrect. Generating grammar errors varies a lot in complexity, some are as simple as a missing comma, which just needs simple string matching and removing the comma. Others require multiple components, working together, to make.

Data Snippet 2: Inflexion data for the Faroese language.


```
{
  "noun": {
      "10-mannafar": [
          {
              "gender": "Neut",
              "case": "Nom",
              "number": "Sing",
              "definiteness": "Ind",
              "word": "10-mannafar"
          },
          {
              "gender": "Neut",
              "case": "Nom",
              "number": "Sing",
              "definiteness": "Def",
              "word": "10-mannafari\u00f0"
          },
...
```


Data Snippet 3: Lemmas for Faroese


```
{
  "noun": {
      "10-mannafar": "10-mannafar",
      "10-mannafari\u00f0": "10-mannafar",
      "10-mannafari": "10-mannafar",
...
```

Spelling

Chapter 5

Results

This chapter presents the results of the experiments conducted in this study. The results are organized into sections: Each section provides a detailed analysis of the performance of the models and the evaluation metrics used to assess their effectiveness.

5.1 mT5 Grammar Model

5.2 mT5 Spelling Model

5.3 spaCy Pipeline

A spaCy pipeline was trained on the Faroese dataset using the following components: a POS tagger, morphologizer, lemmatizer, and dependency parser. The performance of each component was evaluated using either accuracy or precision, recall, and F1 score, depending on what metric is relevant for the given component. The POS tagger and morphologizer were trained on the same dataset, while the lemmatizer and dependency parser were each trained on separate datasets. Due to limited data, the same training set could not be used for all components, the amounts of data available for each component is mentioned in their section. The results of the experiments are presented in the following subsections.

5.3.1 POS Tagger & Morphologizer

In the table below, due to confusuing naming conventions, the POS tagger is referred to as “Tag” and the POS refers to the morphologizers coarse grained POS.

Model	Tag	POS	MORPH
spaCy	**93.39**	**97.80**	**94.03**
Stanza	91.56	96.51	92.92

Table 1: POS Tagger and MORPH performance metrics comparison between the spaCy and Stanza models

Feat	Precision	Recall	F1
Case	96.06	95.54	95.80
Gender	95.32	95.28	95.30
NameType	98.48	97.01	97.74
Number	96.98	96.64	96.81
Definite	97.90	98.00	97.95
Mood	98.31	97.85	98.08
Person	96.19	95.77	95.98
Tense	97.60	96.70	97.15
VerbForm	96.75	95.14	95.94
PronType	95.67	94.91	95.29
Degree	93.25	93.84	93.54
Voice	97.22	92.11	94.59
NumType	98.06	94.39	96.19
Abbr	92.86	96.30	94.55
Foreign	96.23	92.73	94.44

Table 2: Morphologizer performance metrics per feature for the spaCy model

5.3.2 Lemmatizer & Dependency Parser

Model	Lemma	UAS	LAS
spaCy	81.22	82.39	63.23
Stanza	**99.64**	**84.39**	**80.07**

Table 3: Dependency parser performance metrics for the spaCy model

Feat	p	r	f
sents	87.22	90.59	88.87
cc	82.35	72.73	77.24
advmod	86.46	82.18	84.26
cop	73.17	28.04	40.54
nsubj	88.33	86.89	87.60
case	96.15	88.24	92.02
root	14.01	94.57	24.40
obl	68.09	66.32	67.19
mark	89.29	89.29	89.29
ccomp	50.00	60.00	54.55
obj	84.52	87.65	86.06
det	89.19	86.84	88.00
acl:relcl	75.68	70.00	72.73
conj	58.93	58.93	58.93
dep	50.00	10.04	16.72
nummod	100.00	88.89	94.12
advcl	68.18	71.43	69.77
nmod:poss	100.00	83.33	90.91
amod	90.00	81.82	85.71
vocative	100.00	33.33	50.00
appos	27.27	30.00	28.57
nmod	100.00	4.82	9.20
acl	62.50	62.50	62.50
aux	90.74	89.09	89.91
iobj	84.62	84.62	84.62
xcomp	66.67	40.00	50.00
compound:prt	77.27	77.27	77.27
nsubj:cop	0.00	0.00	0.00
flat:name	80.00	80.00	80.00
fixed	0.00	0.00	0.00
discourse	0.00	0.00	0.00

Table 4: Dependency parser performance metrics per type for the spaCy model

5.4 Evaluation Metrics

Chapter 6

Discussion

Chapter 7

Conclusion

Chapter 8

Outlook

References

Bendingarunnurin. Retrieved February 19, 2025, from https://bendingar.fo/um/

Faroese Language Resources by FoNLP. Retrieved December 3, 2024, from https://github.com/FoNLP/faroese_language_resources

Faroese Language Resources by The Centre for Language Technology. Retrieved December 3, 2024, from https://mtd.setur.fo/en/tilfeingi/swoof/product_cat-text/

Faroese Parsed Historical Corpus (FarPaHC). Retrieved February 19, 2025, from https://github.com/einarfs/farpahc

Faroese-FarPaHC Treebank. Retrieved November 29, 2024, from https://github.com/UniversalDependencies/UD_Faroese-FarPaHC

Faroese-OFT Treebank. Retrieved November 29, 2024, from https://github.com/UniversalDependencies/UD_Faroese-OFT

Jóhan Hendrik W. Poulsen. (1998). *Føroysk Orðabók*. Føroya Fróðskaparfelag.

Katrin Næs. (2005). *Færøsk retskrivning – resultater fra en undersøgelse af færøske stile*. http://ojs.statsbiblioteket.dk/index.php/sin/issue/archive

NLLB - Faroese - Mined. Retrieved March 4, 2025, from https://github.com/facebookresearch/fairseq/blob/nllb/examples/nllb/modeling/scripts/flores200/lang_pairs_mine.txt

NLLB - Faroese - Primary. Retrieved March 4, 2025, from https://github.com/facebookresearch/fairseq/blob/nllb/examples/nllb/modeling/scripts/flores200/lang_pairs_primary.txt

Penn Parsed Corpora of Historical English. Retrieved February 19, 2025, from https://www.ling.upenn.edu/hist-corpora/

Rættstavarin. Retrieved February 19, 2025, from https://divvun.org/

Trond Trosterud's tools for Faroese. Retrieved February 19, 2025, from https://gtweb.uit.no/cgi-bin/smi/smi.cgi?text=%C3%81+tunguni+eru+sm%C3%A1ar+tenn.&action=analyze&lang=fao&plang=eng

UDConverter. Retrieved February 19, 2025, from https://github.com/thorunna/UDConverter

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention Is All You Need*. https://doi.org/10.48550/ARXIV.1706.03762

APPENDICES