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 Algorithms

Glossary

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

***DAG* – Directed Acyclic Graph** 6

***MORPH* – Morphologizer** vii, 4, 8

***NLP* – Natural Language Processing** 2

***POS* – Part-of-Speech** iv, vii, 2, 4, 7, 8

***T5* – Text-to-Text Transfer Transformer** 2

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

Background

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

2.1 Transformers

2.1.1 T5

Text-to-Text Transfer Transformer (T5) 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.

2.1.2 BERT

Bidirectional Encoder Representations from Transformers (BERT) is a pre-trained language model developed by Google.

2.2 spaCy

Spacy is an open-source software library for advanced natural language processing Natural Language Processing (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, morphologizer, lemmatizer and dependency parsing capabilities.

Chapter 3

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.

3.1 Data Collection

All the training data was from various online repositories and websites [1], [2], [3], [4], [5]. 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. Sprotin.fo [6], a website that hosts dictionaries, was scraped for faroese word and name inflexions. 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.

3.2 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 **ð** 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 errorifying it, a lot of time is saved in the errorification process. The errorification process will be covered in the data augmentation section.

The labeled data from the faroese Universal Dependencies[4], [5] repos 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.

3.3 Data Augmentation

The data is augmented by taking correct text and errorifying it. The errorification 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 errorification 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. The error type hierarchy for the grammar model is shown in Figure 1.

3.3.1 Errorification

The error types are classified as follows:

- **Inflexions** A general error type for inflexion errors where it is unsure what type of inflexion it is.
- **Adjectives** inflexion errors for adjectives.
- **Nouns** inflexion errors for nouns.
- **Verbs** inflexion errors for verbs.
- **Missing_Ð** The character **ð** is missing. Ð is a silent letter in faroese, so it is common to forget it.
- **Added_Ð** The character **ð** is added. Ð is a silent letter in faroese, so it is common to add it in places it's not supposed to be.

- **Pronouns**
- **Proper_Nouns**
- **Missing_Comma**
- **Added_Comma**
- **Period**
- **Missing_Space**
- **Ordinal_Number**
- **Apostrophe**
- **Hyphen**
- **Quotation_Mark**
- **Colon**
- **Semicolon**
- **Question_Mark**
- **Exclamation_Mark**
- **Capitalize**
- **Lowercase**
- **Split_Thousands**
- **Split_Over_100**
- **Under_100_Dont_Split**
- **Word_Confusion**

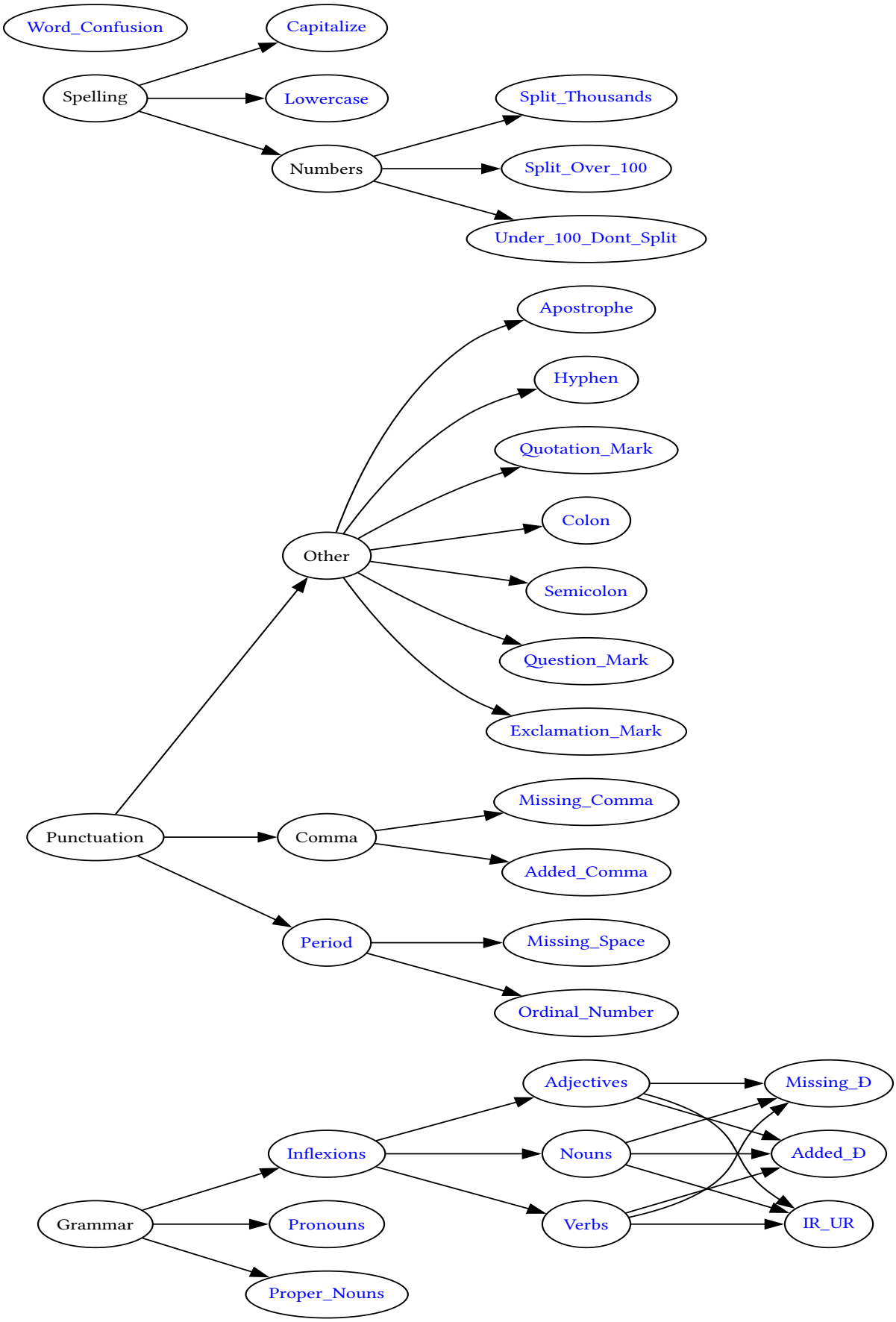

Figure 1: A Directed Acyclic Graph (DAG) showing the error type hierarchy, where the blue nodes are the error types that are used in the errorification process. Each edge indicates that a child node has higher priority than its parent node.

Chapter 4

Design

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 Faroese MT5

5.2 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.

5.2.1 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

5.2.2 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

5.3 Evaluation Metrics

Chapter 6

Discussion

Chapter 7

Conclusion

Chapter 8

Outlook

References

[1] *Faroese Language Resources by FoNLP*. [Online]. Available: https://github.com/FoNLP/faroese_language_resources

[2] *Faroese Language Resources by The Centre for Faroese Language Technology*. [Online]. Available: https://mtd.setur.fo/en/tilfeingi/swoof/product_cat-text/

[3] *Faroe University Press*. [Online]. Available: https://ojs.setur.fo/

[4] *Universal Dependensies Faroese-FarPaHC Treebank*. [Online]. Available: https://github.com/UniversalDependencies/UD_Faroese-FarPaHC

[5] *Universal Dependensies Faroese-OFT Treebank*. [Online]. Available: https://github.com/UniversalDependencies/UD_Faroese-OFT

[6] "Sprotin." [Online]. Available: https://www.sprotin.fo/

APPENDICES