UNIVERSITY
OF TURKU

Distinguishing Noise and Main Text Content from Web-Sourced Plain Text Documents Using Sequential Neural Networks

Anna Salmela
Master's thesis
Language Specialist Degree Programme, Digital Language Studies
School of Languages and Translation Studies
Faculty of Humanities
University of Turku
April 2022

Master's Thesis

Language Specialist Degree Programme, Digital Language Studies:
Anna Salmela
Distinguishing Noise and Main Text Content from Web-Sourced Plain Text Documents Using Sequential Neural Networks
Number of pages: 60 pages, 9 appendices

Boilerplate removal and the identification of the actual textual content is a crucial step in web corpus creation. However, existing methods don’t always filter out the noise perfectly and are often not applicable for plain text corpora. In this thesis, I will develop machine learning methods to identify the main textual content in plain text documents. I will utilize transfer learning and pretrained language models as a base for training monolingual models with French and Swedish data as well as a multilingual model with French, Swedish, English, Finnish, German and Spanish data. I will compare two machine learning architectures based on the XLM-RoBERTa language model: first a classification model built on top of the pretrained XLM-RoBERTa model and a second model using an additional Long Short-Term Memory (LSTM) network layer. I will show that the LSTM layer improves the classification of the XLM-RoBERTa model and the built multilingual model performs well even with data in unseen languages. I will perform a further analysis on the results and show that the results of the boilerplate detection with the trained models differ with text varieties. Certain types of text documents, such as lyrical texts or discussion forum texts pose challenges in boilerplate detection, and it would be beneficial for future research to focus on gathering data that has been difficult to clean.

Key words: web corpus creation, boilerplate detection, artificial neural networks, sequence-to-sequence learning.

Table of contents

1 Introduction

Due to the rise of the use of the Internet over the last few decades, the importance of linguistic data collected from the Web has similarly increased. In response to the manually selected smaller collections of web-documents, automatic processes have been developed to obtain large amounts of textual data from the Internet, resulting in large-scale web corpora such as Common Crawl (Wenzek et al., 2019) and OSCAR (Abadji et al., 2022). Web-sourced corpora can be powerful tools in linguistic analysis and machine learning, but they often favour quantity over quality. Texts sourced from web crawls often contain all sorts of metadata also known as boilerplate that should be filtered out in order to improve the quality of the data (Laippala et al., 2020). Filtering out this kind of metadata is often overlooked and underreported in web corpus construction process even though it is a crucial step in order to obtain high quality data (Kilgarriff, 2007).

Even though there are great attempts in improving the quality of web-sourced text data, there is no one size fits all approach to cleaning the web-sourced texts, since they can vary greatly in their linguistic features. The existing methods don’t always generalize well and as such leave room for improvement (Laippala et al., 2020). Previous approaches have often used HTML documents as their base for boilerplate detection, and not much progress has been documented in boilerplate detection from plain text. There are however web-corpora such as OSCAR (Abadji et al., 2022) that don’t have their HTML documentation available. In addition to this, in some cases the boilerplate cleaning based on the HTML documents isn’t enough (Laippala et al., 2020) and a further cleaning of the textual data might be necessary. Most of the existing applications are based on heuristics such as Trafilatura (Barbaresi, 2021) and jusText (Pomikálek, 2011), and some on machine learning such as BoilerNet (Leonhardt et al., 2020) and Web2Text (Vogels et al., 2018).

Some groundwork has been made in defining what constitutes as boilerplate, and how it usually appears among main content text. The definition of boilerplate isn’t clear-cut (Pomikálek, 2011), but some annotation guidelines have been created in order to create datasets for boilerplate cleaning application development: in general, any text that doesn’t belong to the main content of a web page is counted as boilerplate. A common notion is that boilerplate and main content appear in blocks (Leonhardt et al., 2020; Pomikálek, 2011), and thus a sequential classifier seems like a logical solution to the problem.

The goal of this thesis is to develop a machine learning method for boilerplate detection in plain text documents and evaluate its performance. This will be done by first training an XLM-R model to label each line in a document and then use this classification as a sequence that will be fed to a separate model with an LSTM layer. I will train monolingual models in French and Swedish as well as a multilingual model with French, Swedish, English, Finnish, German and Spanish data. The research questions are based on the performance of the trained model and are thus the following:

1. Does the use of sequentiality improve the boilerplate detection done by the first model?
2. How well does the multilingual model perform with data in unseen languages?
3. How do the results of said sequential classification compare to the existing applications?
4. How well does the sequential model perform with different varieties of web-sourced data?

The language model used for the base of the training in this thesis is XLM-RoBERTa (Conneau et al., 2019), that has been previously used successfully in e.g., in text classification bot in single-language and multilingual settings (Repo et al., 2021; Rönnqvist et al., 2021). I will attempt to improve the results achieved with the XLM-R by training another model with an added Long Short-Term Memory (LSTM) layer in order to take the sequential form of the data into account: as noted in Leonhardt et al. (2020) and Pomikálek (2011), main content and boilerplate often appear in blocks. LSTM networks have been previously used in boilerplate detection (Leonhardt et al., 2020), as well as in text generation (Karpathy, 2015; Sutskever et al., 2011) and sequence tagging (Huang et al., 2015). The results of the classification are compared to Trafilatura (Barbaresi, 2021), jusText (Pomikálek, 2011), Web2Text (Vogels et al., 2018) and BoilerNet (Leonhardt et al., 2020) to estimate the models' performance compared to previous accomplishments in boilerplate detection.

The data used in this thesis consists mainly of FreCORE and SweCORE that have been previously used in (Repo et al., 2021) and that are annotated by register, "a [text] variety associated with a particular situation of use" (Biber & Conrad, 2019), such as News texts, Encyclopedia Articles or Discussion Forum texts. For the purposes of this thesis, the data is annotated per line in two classes: boilerplate and main content. The different text type varieties on the Internet might affect the results of the classification. I will examine the effect by calculating the models' performance separately for different Internet registers based on the register annotations available for FreCORE and SweCORE.

This thesis is done in collaboration with TurkuNLP[1] research group as a part of their project "Massively multilingual modeling of registers in web-scale corpora". This work contributes a pipeline for web corpus cleaning that can be used in future research. The data used in this thesis consists of 1982 French and 2 399 Swedish documents sourced from the free web. They contain 84 103 French and 99 406 Swedish lines that are manually annotated into two classes: boilerplate and main content. This data has been previously used in Repo et al. (2021) and contains annotated register information that will be used in the analysis. In addition to this, smaller datasets of English, Finnish, German and Spanish are included to conduct multilingual tests.

The structure of this thesis is the following: in chapters 2-4, I will present the theoretical framework necessary to the study. Chapter 2 will shed light on web corpus creation and its problematics, as well as the meaning behind the term "Web as Corpus". In Chapter 3, I will briefly explain the mechanics behind the artificial neural networks used in this thesis. Chapter 4 introduces the background of boilerplate removal and some previous approaches to it. In Chapter 5, I will focus on the data used in this thesis and explain the experimental setup in training the model. The results are analysed in Chapter 6 in both monolingual French and Swedish as well as multilingual settings, and Chapter 7 concludes the work.

[1] https://turkunlp.org/

2 Web as Corpus

The World Wide Web serves an enormous amount of textual data that can be harnessed for linguistic research: as of January 2021, there are 5.1 billion active Internet users[2] and 1.9 billion active websites[3]. The importance of digital media has steadily increased during the past decades, and the quick emergence of new social platforms forces the field in constant flux and development. Recent subjects of interest in the research of the Web have been e.g. Twitter discourse (Johansson et al., 2018), blogs (Lehti, 2013) and discussion forums and their discourse themes (Jantunen, 2018).

Corpus linguistics can be seen as a research field that utilizes digitalized corpora in linguistic analysis. For some, it represents a linguistic research method rather than a field: the results of a corpus linguistics analysis can be further used in other linguistic research, such as discourse analysis (Partington, 2013, p. 5-6). According to Biber et al. (1998), the goal of corpus linguistics is also to describe and analyse the usage of a language. Biber et al. emphasize that corpus linguistics enables the study of language usage or linguistic characteristics among a group of writers or speakers.

Traditionally, a linguistic corpus is a large collection of texts that are chosen to represent a certain area of a language, be it a certain genre, a temporal window or a method of usage (Baker, 2009, p. 34-35). These corpora are then used in quantitative analysis of the language, for example by searching for the most frequently used words in a text: can be presumed that if a word or an expression appears frequently in a corpus, it is probably typical for the corpus in question (Baker, 2006; Gatto, 2014). These kinds of word frequencies can be further analysed e.g., by comparing two different corpora to find words or expressions that appear more frequently in one corpus.

Even though the traditional corpora are usually very large, corpus linguistics is sometimes criticized of representing only parts of language without giving an exhaustive description of its use (Baker, 2006). Web as corpus approach differs from this since its goal is to represent the whole World Wide Web that contains lots of extremely varied data (Barbaresi, 2021). Web corpus construction includes "crawling, downloading, 'cleaning' and de-duplicating the data, then linguistically annotating it and loading it into a corpus query tool" (Kilgarriff,

---

[2] https://www.internetlivestats.com/internet-users/
[3] https://www.internetlivestats.com/total-number-of-websites/

2007). The crawling step has been made easier with datasets like Common Crawl (Wenzek et al., 2019) that essentially makes it possible to outsource the crawling step and use a massive selection of randomly sampled “snapshot” of the web. Many similar text extraction methods have been developed in order to gather as much data as possible (Barbaresi, 2016; Laippala et al., 2020), but it is essential to further filter out the contents of the crawled data.

The data from these kind of ready-made data sets is typically varied in its content and quality and not all of it is useful in linguistic analysis. Even the useful parts may contain duplicates, machine translation and useless metadata such as lists of links, headers, footers and navigation bars that need to be filtered in order to increase quality of the corpus (Barbaresi, 2021). This step is often overlooked even though it impact all further analysis and computational linguistics pipelines (Barbaresi, 2021; Kilgarriff, 2007). After cleaning and filtering a web corpus, there is often little information about the documents in it. Because of the random-sampled nature of a web corpus, its contents typically remain a mystery without further annotation and it is not until its completion that the contents of a web corpus can be listed (Barbaresi, 2019; Laippala et al., 2020).

3 Boilerplate Removal

In this chapter, I will define boilerplate as a concept, explain the task of boilerplate removal and its importance in web corpus creation process and introduce some different existing approaches in its implementation.

Boilerplate removal (also referred to as web scraping, web page segmentation, web page cleaning, template extraction, content extraction, etc.) is a crucial yet overlooked step in web corpus creation process (Barbaresi, 2021). According to Kilgarriff (2007), a well-done cleaning step results in a better outcome since all further analysis depends on the cleanliness of the data. The cleaning process is often done in seclusion with a particular data set in mind, rendering the developed tools to generalize to data in a different page style format or a different language poorly (Barbaresi & Lejeune, 2020; Kilgarriff, 2007). In addition to this, the documentation of the cleaning process is often very brief, if at all present, in many publications (Kilgarriff, 2007).

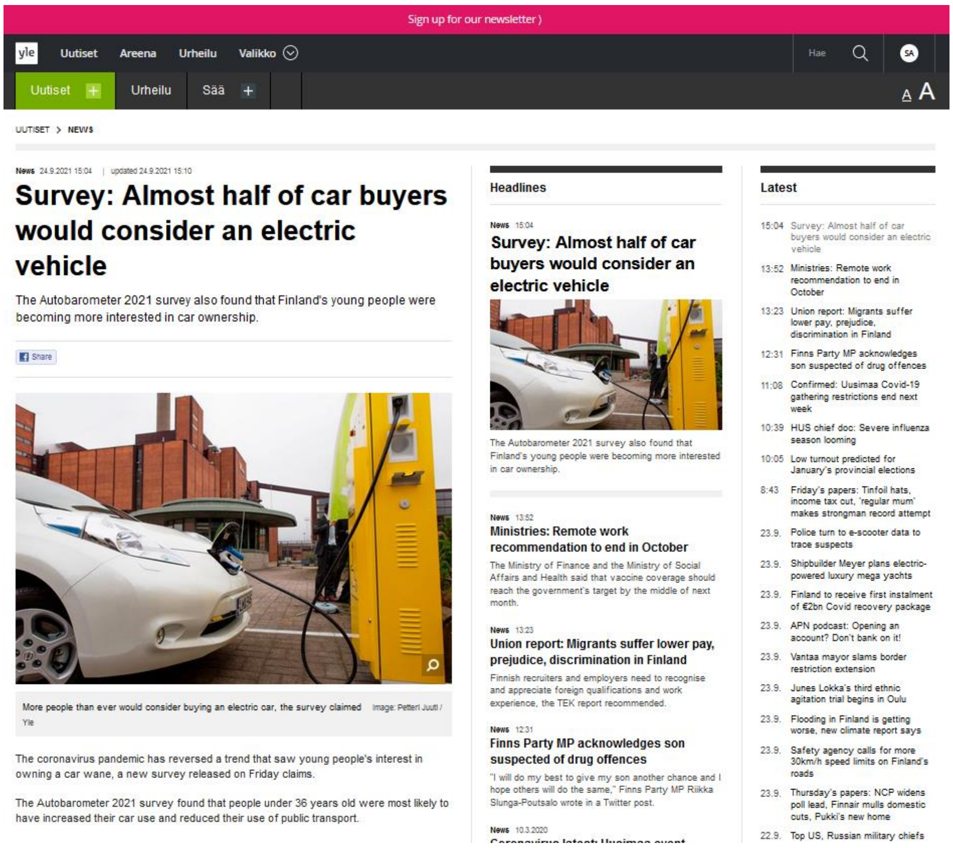
Definition

Web-extracted text content can usually be divided into main content and boilerplate (Pomikálek, 2011; Vogels et al., 2018). Pomikálek (2011) notes that boilerplate is often defined vaguely as "non-informative parts outside of the main content of a Web page" and it is often machine generated and repeated across the pages of one website. Typical examples of boilerplate include navigation menus, lists of links and ads. The distinction between boilerplate and main content isn't always straightforward. For example, news articles are often accompanied by an abstract of a related article and a link to the full text. Here the line between the main content and boilerplate gets blurry: does the abstract belong to the main content or is it boilerplate?

As there is no clear-cut definition of boilerplate, several guidelines have been made for the purpose of boilerplate annotation. Two most widely used might be CleanEval[4] competition (Baroni et al., 2008) gold standard and KrdWrd[5] annotation tool guidelines. According CleanEval guidelines, boilerplate includes navigation information, internal and external link lists, copyright notices and other legal information, headers, footers and templates that are

[4] https://sigwac.org.uk/cleaneval/annotation_guidelines.html
[5] https://krdwrd.org/doc_manual/node6.html

repeated across the pages of one site, advertisements and web-spam. KrdWrd's definition is less strict: it outlines boilerplate as navigation information, copyright information, links that are not part of the text content, as well as headers and footers. In addition, KrdWrd annotation guideline however suggests that incomplete sentences, text containing non-language, advertisements, foreign language text and lists should not be included in the main content.



News 24.9.2021 15:04 | updated 24.9.2021 15:10

Survey: Almost half of car buyers would consider an electric vehicle

The Autobarometer 2021 survey also found that Finland's young people were becoming more interested in car ownership.

More people than ever would consider buying an electric car, the survey claimed Image: Petteri Juuti / Yle

The coronavirus pandemic has reversed a trend that saw young people's interest in owning a car wane, a new survey released on Friday claims.

The Autobarometer 2021 survey found that people under 36 years old were most likely to have increased their car use and reduced their use of public transport.

Headlines

News 15:04
Survey: Almost half of car buyers would consider an electric vehicle
The Autobarometer 2021 survey also found that Finland's young people were becoming more interested in car ownership.

News 13:52
Ministries: Remote work recommendation to end in October
The Ministry of Finance and the Ministry of Social Affairs and Health said that vaccine coverage should reach the government's target by the middle of next month.

News 13:23
Union report: Migrants suffer lower pay, prejudice, discrimination in Finland
Finnish recruiters and employers need to recognise and appreciate foreign qualifications and work experience, the TEK report recommended.

News 12:31
Finns Party MP acknowledges son suspected of drug offences
"I will do my best to give my son another chance and I hope others will do the same," Finns Party MP Riikka Slunga-Poutsalo wrote in a Twitter post.

News 10.3.2020

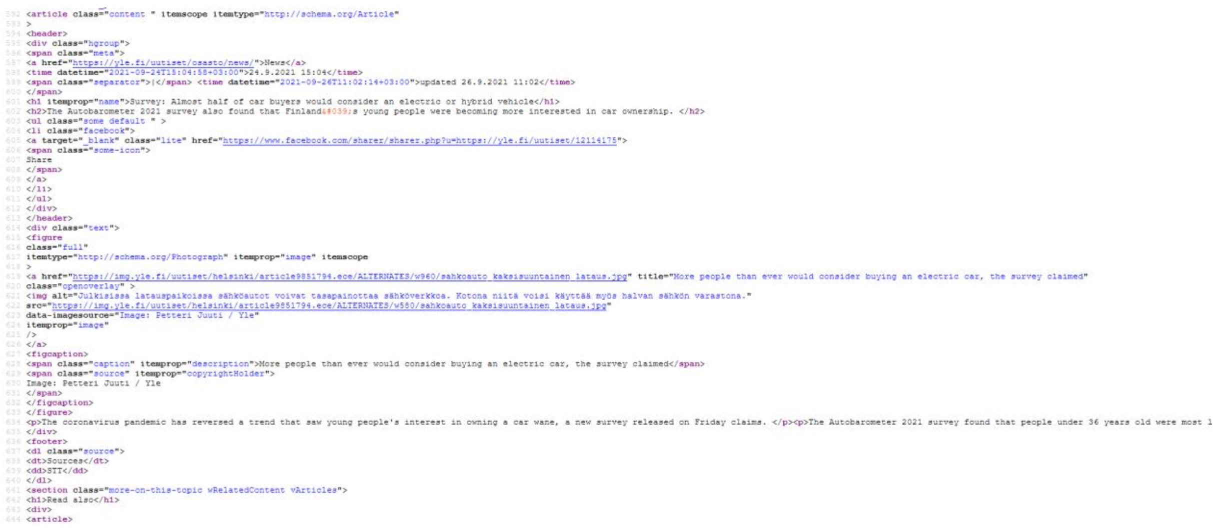
Latest

15:04	Survey: Almost half of car buyers would consider an electric vehicle
13:52	Ministries: Remote work recommendation to end in October
13:23	Union report: Migrants suffer lower pay, prejudice, discrimination in Finland
12:31	Finns Party MP acknowledges son suspected of drug offences
11:08	Confirmed: Uusimaa Covid-19 gathering restrictions end next week
10:39	HUS chief doc: Severe influenza season looming
10:05	Low turnout predicted for January's provincial elections
8:43	Friday's papers: Tinfoil hats, income tax cut, 'regular mum' makes strongman record attempt
23.9.	Police turn to e-scooter data to trace suspects
23.9.	Shipbuilder Meyer plans electric-powered luxury mega yachts
23.9.	Finland to receive first instalment of €2bn Covid recovery package
23.9.	APN podcast: Opening an account? Don't bank on it!
23.9.	Vantaa mayor slams border restriction extension
23.9.	Junes Lokka's third ethnic agitation trial begins in Oulu
23.9.	Flooding in Finland is getting worse, new climate report says
23.9.	Safety agency calls for more 30km/h speed limits on Finland's roads
23.9.	Thursday's papers: NCP widens poll lead, Finnair mulls domestic cuts, Pukki's new home
22.9.	Top US, Russian military chiefs

Figure 1: An example of a web page view on a news site

Figure 1 shows a typical news article page on a news site yle.fi[6]. The main body of text of the article is located on the left side of the page. On the right side, there are two columns of links to recommended and latest articles. On the top of the page, there is a navigation bar. However, finding the main content from the corresponding HTML document may prove difficult. Let's examine the HTML document that is behind this page, show in Figure 1.

6 https://yle.fi/uutiset/osasto/news/survey_almost_half_of_car_buyers_would_consider_an_electric_vehicle/12114175

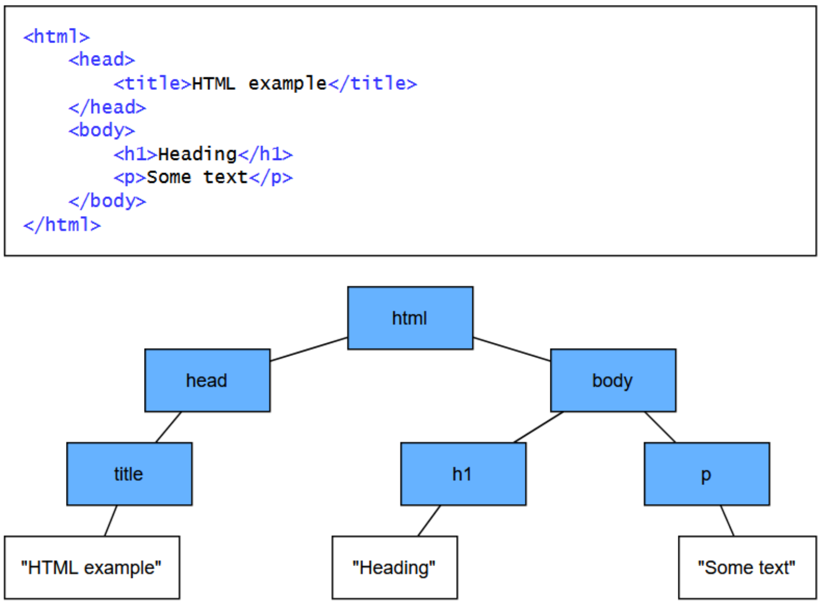


```
592 <article class="content " itemscope itemtype="http://schema.org/Article"
593 >
594 <header>
595 <div class="hgroup">
596 <span class="meta">
597 <a href="https://yle.fi/uutiset/osasto/news/">News</a>
598 <time datetime="2021-09-24T15:04:58+03:00">24.9.2021 15:04</time>
599 <span class="separator">|</span> <time datetime="2021-09-26T11:02:14+03:00">updated 26.9.2021 11:02</time>
600 </span>
601 <h1 itemprop="name">Survey: Almost half of car buyers would consider an electric or hybrid vehicle</h1>
602 <h2>The Autobarometer 2021 survey also found that Finland&#039;s young people were becoming more interested in car ownership. </h2>
603 <ul class="some default " >
604 <li class="facebook">
605 <a target="_blank" class="lite" href="https://www.facebook.com/sharer/sharer.php?u=https://yle.fi/uutiset/12114175">
606 <span class="some-icon">
607 Share
608 </span>
609 </a>
610 </li>
611 </ul>
612 </div>
613 </header>
614 <div class="text">
615 <figure
616 class="full"
617 itemtype="http://schema.org/Photograph" itemprop="image" itemscope
618 >
619 <a href="https://img.yle.fi/uutiset/helsinki/article9851794.ece/ALTERNATES/w960/sahkoauto_kaksisuuntainen_lataus.jpg" title="More people than ever would consider buying an electric car, the survey claimed"
620 class="openoverlay" >
621 <img alt="Julkisissa latauspaikoissa sähköautot voivat tasapainottaa sähköverkkoa. Kotona niitä voisi käyttää myös halvan sähkön varastona."
622 src="https://img.yle.fi/uutiset/helsinki/article9851794.ece/ALTERNATES/w580/sahkoauto_kaksisuuntainen_lataus.jpg"
623 data-imagesource="Image: Petteri Juuti / Yle"
624 itemprop="image"
625 />
626 </a>
627 <figcaption>
628 <span class="caption" itemprop="description">More people than ever would consider buying an electric car, the survey claimed</span>
629 <span class="source" itemprop="copyrightHolder">
630 Image: Petteri Juuti / Yle
631 </span>
632 </figcaption>
633 </figure>
634 <p>The coronavirus pandemic has reversed a trend that saw young people's interest in owning a car wane, a new survey released on Friday claims. </p><p>The Autobarometer 2021 survey found that people under 36 years old were most l
635 </div>
636 <footer>
637 <dl class="source">
638 <dt>Sources</dt>
639 <dd>STT</dd>
640 </dl>
641 <section class="more-on-this-topic wRelatedContent vArticles">
642 <h1>Read also</h1>
643 <div>
644 <article>
```

Figure 2: An excerpt of the HMTL code of the news site page shown in Figure 1

Figure 2 shows only a small excerpt of the HTML document of the news article page. The whole HTML document is 1604 lines total, but the desired text content can be found on lines 592-644, including the date of publication, the title, the subtitle, the image caption text, the main body of text and sources. These all are between <article> section of the document. Rest of the HTML document contains for example links to recommended articles and page formatting.

Most of the existing applications in boilerplate removal utilizes the HTML markup of the web pages. They make it possible to distinguish different parts of the page, even when the HTML-markup convention isn't consistent between different web sites (Barbaresi, 2021). The HTML nodes construct a Document Object Model (DOM) tree, which is used to further segment the web page. A very simple HTML document and its DOM tree is presented in Figure 2.



```
<html>
    <head>
        <title>HTML example</title>
    </head>
    <body>
        <h1>Heading</h1>
        <p>Some text</p>
    </body>
</html>
```

Figure 3: HTML example and corresponding Document Object Model tree

As shown in Figure 3, a HTML document can be divided node-wise into segments within the document. In this example, the “head” node contains the page title, which is also usually shown at the top of the web browser. Under the <body> node, there are two nodes: <h1> as in heading and <p> as in paragraph. These are typical markers for the main text content: in the example shown in Figure 2, <h1> corresponds to the main title of the news article (“Survey: Almost half of car buyers would consider an electric or hybrid vehicle”) and <h2> its subtitle (“The Autobarometer 2021 survey also found that Finland's young people were becoming more interested in car ownership.”). The main body of text is often divided into several <p> paragraphs. Even when utilizing the HTML markings, it can be difficult to develop a method that is generalizable due to the various HTML conventions across the Internet (Barbaresi, 2021).

In the following chapter, I will shortly present some of the boilerplate removal applications made for cleaning up web corpora. These methods are all based on cleaning html-marked text and as such they are not applicable to removing boilerplate or noise from plain text. It should also be emphasized that the existing boilerplate removal methods might not be enough to remove all the noise from web-sourced texts, as noted in Laippala et al. (2020). Even a successful HTML level filtering can still leave some noise in the text, and applications aimed for plain text are needed to resolve this.

Approaches to Boilerplate Removal

The approaches to boilerplate removal can in general be divided in two: rule-based and machine learning based approaches. Examples of rule-based boilerplate removal applications are jusText[7], Onion[8] (ONe Instance ONly) (Pomikálek, 2011) as well as Trafilatura[9] (Barbaresi, 2021) that classify text content on the web based on rule-based filters and heuristics.

jusText splits the HTML document into blocks based on the DOM tree nodes. Each block is first given a classification out of two possible classes: main content for long blocks with grammatical text, boilerplate for all the other long blocks, short blocks containing links and any blocks with many links. Grammatical text is found using a function word heuristic: according to Pomikálek (2011, p. 29), grammatical text usually contains a certain proportion of function words, whereas boilerplate doesn't. The blocks that don't fall clearly into either are further classified as short or near-good. In total the classification contains four classes: good, bad, short and near-good. These labels are given based on the length of the block, the number of links and the proportional amount of function words. (Pomikálek, 2011, p. 29-33)

[7] https://github.com/miso-belica/jusText
[8] https://corpus.tools/wiki/Onion
[9] https://github.com/adbar/trafilatura

Table 1: jusText context-wise block classification (Pomikálek, 2011)

BAD	BAD
SHORT	BAD
BAD	BAD
SHORT	BAD
NEAR-GOOD	BAD
BAD	BAD
SHORT	BAD
SHORT	BAD
GOOD	GOOD
SHORT	GOOD
SHORT	GOOD
GOOD	GOOD
SHORT	GOOD
NEAR-GOOD	GOOD
SHORT	BAD
BAD	BAD
NEAR-GOOD	BAD

After the labelling, the short and near-good blocks are further classified context-wise as in the example in Table 1. According to Pomikálek (2011, p. 31-32), "a boilerplate block is usually surrounded by other boilerplate blocks and vice versa". The short and near-good blocks are thus re-evaluated based on their context to form clusters of main content and boilerplate or groups of good and bad blocks. If a group of short and near-good blocks are surrounded by two good or bad blocks, they are classified according to the surrounding blocks. In the case where the group is surrounded with a "good" block on the other side and a "bad" block on the other, the near-good block nearest to the bad block is treated as a delimiter of the good and bad block area. If there is no near-good block, but the blocks are labelled as "short", the whole group is labelled as "bad".

Pomikálek (2011, p. 55, 66) also notes the importance of detecting duplicate content in web corpora. According to them, duplicate texts can cause duplicate concordance lines or bias in results of statistical analysis of the corpus since the frequencies of the words and expressions of the duplicate part is inflated. They define a duplicate being often generated, copy-pasted and non-independently created text, whereas naturally repeated text such as some expressions in interactive discussion are part of natural text and are thus not duplicates.

Trafilatura (Barbaresi, 2021) is another rule-based method for boilerplate removal that combines jusText and readability-l.xml[10] libraries and adds its own rule-based filters and content heuristics of top of them. In addition to the main text content, Trafilatura also extracts some metadata that can be useful in linguistic analysis. Extracted metadata includes the title of post, the title of blog, the date of publication, URL, the author as well as categories and tags if available (Barbaresi, 2016).

Even though the rule-based methods have been somewhat successful, some approaches make use of machine learning in the boilerplate removal. Some recent applications utilize deep neural networks. Web2text[11] (Vogels et al., 2018) is a machine learning based boilerplate removal method that uses convolutional neural networks and hand-picked features to classify the blocks of text nodes in the HTML document. Similarly to Pomikálek (2011), Vogels et al. (2018) treat the task as a sequence classification problem: the web page can be seen as a sequence of blocks of main content and boilerplate. BoilerNet (Leonhardt et al., 2020) is a neural sequence labelling model based on HTML tags and text on a web page. Leonhardt et al. (2020) also rely on the notion that boilerplate and main content appear in blocks: they use LSTM networks to recognize the main content on a web page. I will use this notion in building the model for the line-wise classification.

[10] https://github.com/buriy/python-readability
[11] https://github.com/dalab/web2text

4 Artificial Neural Networks in Digital Linguistics

In this chapter, I'll briefly introduce neural network systems and their applications in natural language processing and present the special case of sequential classification and artificial neural networks designed for this task.

Machine learning can be roughly divided into two categories: supervised and unsupervised learning. In supervised machine learning, the given input has a desired output and some kind of labelling. This kind of machine learning is used in text classification and question-answer tasks, for example. In unsupervised learning, there are no distinct output categories or desired results, but the model is used to learn patterns within the data. Unsupervised learning can be used e.g., in topic modelling. (Jurafsky & Martin, 2009) In this thesis, I will focus on supervised machine learning: the data used in this thesis is line-wise annotated into two classes, and the trained model aims to learn to classify the lines into these two classes.

Artificial neural networks have been widely applied in computational linguistics in tasks such as machine translation (Bahdanau et al., 2014; Bawden et al., 2020), text generation (Karpathy, 2015; Sutskever et al., 2011) and different kinds of classification tasks (Kim, 2014; Laippala et al., 2019; Repo et al., 2021; Rönnqvist et al., 2021; Salminen et al., 2020).

The development of artificial neural networks started already in the 1940s, when McCulloch and Pitts (McCulloch & Pitts, 1943) presented a simple computational model imitating the neural activity in the human brain. This idea was further developed with Rosenblatt's perceptron (1958), which is a simple artificial neural network that consists of one neuron. It tries to replicate the behaviour of a real neuron by taking in the inputs, calculating a weighted sum of said inputs and passing it through a threshold function before outputting the result.

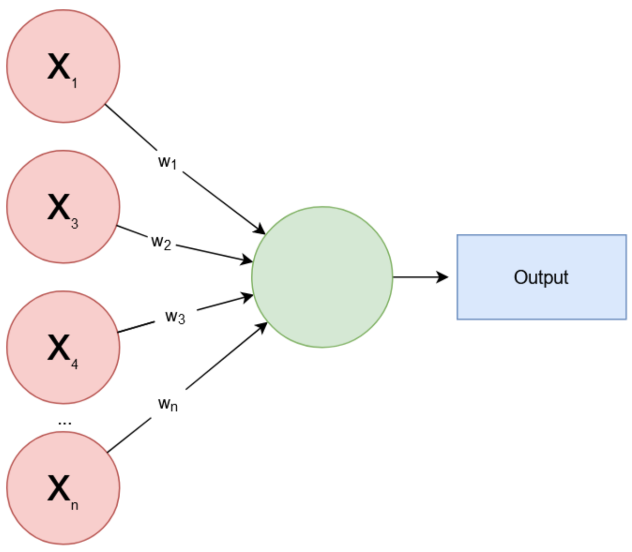


Figure 4: A visualization of a perceptron

Figure 4 represents a perceptron: the inputs x_1 ... x_n marked with red and their weights w_1 ... w_n are fed into the perceptron cell which calculates the weighted sum and the threshold function and puts out the prediction. Depending on the threshold function, the output can be e.g., a predicted label or probabilities of different class labels. These kinds of perceptrons can be stacked to form a multilayer perceptron, a feed-forward artificial neural network with one or more hidden layers in between the input and output layers. In general, if a model has more than one hidden layer, it is called a deep neural network. This structure is called feed-forward since it handles information in a chronological order.

Sequence Classification and Sequence-to-Sequence Learning

In a traditional classification task, it can be assumed that the data is independent and identically distributed: all the examples have the same probability of occurring in the data and are not dependent on any other data points. An example of a binary classification task could be predicting e-mails as “spam” or “not spam”. In this example, there are two possible classes and the classification doesn’t consider the surrounding e-mails since the classification doesn’t depend on the context: it doesn’t matter whether the previous or following e-mail is spam or not to predict the class of the current e-mail.

The independent and identically distributed data assumption doesn’t hold with sequential data, where the data order and context matter. This is the case with many natural language applications such as text generation (Karpathy, 2015; Sutskever et al., 2011), machine translation (Sutskever et al., 2014), named entity recognition (Luoma et al., 2020) and part-of-speech tagging (Kanerva et al., 2018). Language is naturally sequential: it consists of sequences of characters, words, sentences, chapters, etc. Two sequences containing the exact same characters can be semantically entirely different: “Sam eats cheese” and “cheese eats Sam” are two different phrases with two different meanings. In the context of this thesis, the classification of a line within a text depends on the surrounding lines: boilerplate and main text content often appear in blocks (Pomikálek, 2011; Vogels et al., 2018).

Sequence-to-sequence learning is an umbrella term that covers several types of mappings, commonly categorized by the length of the input and output as one to many, many to one and many to many. One to many maps a single variable into many, an example could be image captioning where the algorithm is given a single image and produces a sequence for the caption. Many to one produces a single output for a sequence of variables in cases such as text classification, where the algorithm is given a sequence of words and it outputs a target class. Many to many produces a sequence output out of a sequence input, as in machine translation where the input and output might be of different lengths, and sequence labelling such as named entity recognition or part-of-speech tagging where the input and output usually are the same length. (Karpathy, 2015) In the case of this thesis, the lines are first labelled as many-to-one: one label per a line of text (a sequence). Afterwards the whole document is given labels as a many-to-many task: the input sequence is the last hidden layer of the model resulted from the first step, and the output is a sequence of labels for the whole document.

A Recurrent Neural Network (RNN) is an adaptation of the feed-forward network that can model sequential data (Sutskever et al., 2011). It can be thought of as a neural network with loops: the information is passed along from a step in the network to the next through the hidden state of the RNN cell (Olah, 2015). A visualization of a standard RNN is shown in Figure 5. Here, A symbolizes a part of the neural network, x_t the input it gets and h_t the hidden state per time step. At each time step, the RNN cell receives an input, updates the hidden state and makes a prediction. Note that even though the input and hidden state change, the part of the network stays the same, in a loop. The hidden state stores information from the previous time steps and thus uses its “memory” to predict the values of the current input.

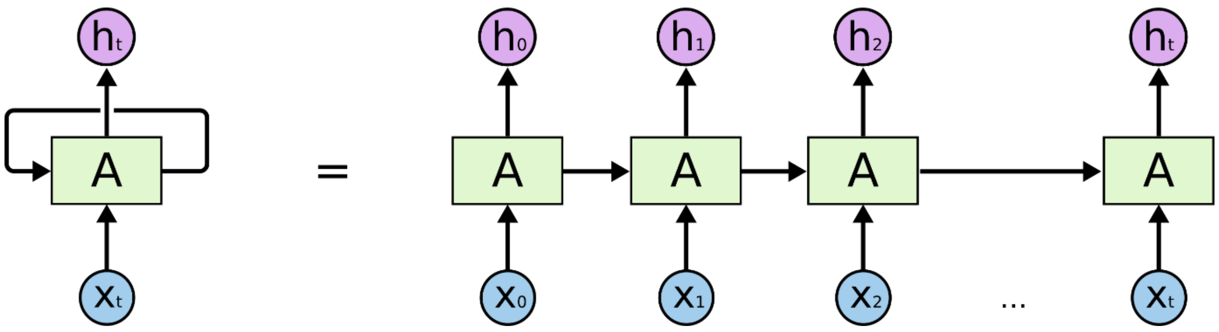


Figure 5: A visualization of a standard Recurrent Neural Network (Olah, 2015)

Notice that in Figure 5, information is processed in a chronological order. This makes this RNN example a feed-forward RNN. Sometimes it is, however, beneficial to utilize both “past” and “future” information. It is possible to attach two RNNs to form a bidirectional recurrent neural network (Schuster & Paliwal, 1997) in order to take the future information into account.

Despite its advantages, a standard RNN cell has trouble with its short-term memory and the hidden states lose information from the previous steps quite quickly. In addition to this, RNN networks are hard to train (Sutskever et al., 2011). The Long Short-Term Memory networks (LSTM) (Hochreiter & Schmidhuber, 1997) try to battle the problems with standard recurrent neural networks: they are a type of a RNN that are designed to learn long term dependencies by including a memory unit to store information from the previous steps. A visualization of an LSTM network can be seen in Figure 6.

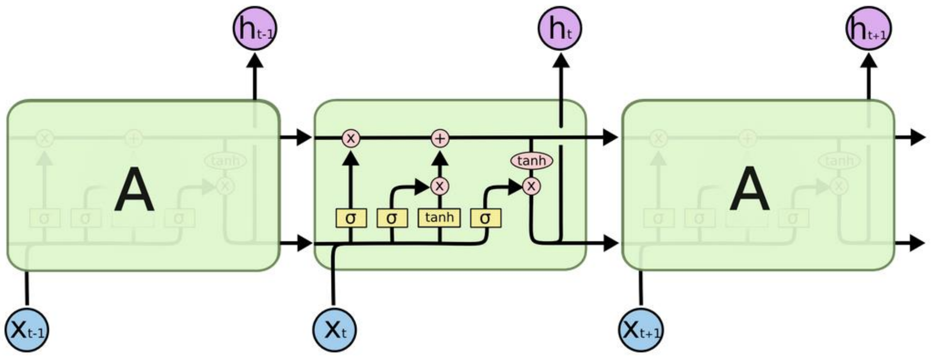


Figure 6: A visualization of an LSTM network (Olah, 2015)

In Figure 6, you can see an example of a three-cell excerpt of an LSTM network, where the middle cell is described in detail. Similar to the previous example, here x_t represents the input fed to the network and h_t its hidden state per time step. The upmost line of the cell represents the cell state that stores information for long periods of time. The cell state is controlled by three kinds of gates, here marked by “x”: a forget gate to decide what information is kept, an input gate to decide which values will be updated, and an output gate to decide the output. All of this is connected by four fully connected neural network layers that connect each of the neurons to the corresponding ones in the preceding layer, marked by the yellow boxes. LSTMs have been successfully used in applications such as text generation (Karpathy, 2015; Sutskever et al., 2011), sequence tagging (Huang et al., 2015) and boilerplate removal (Leonhardt et al., 2020).

Due to the sequential nature of natural language, recurrent neural networks have long been a standard in computational linguistic analysis (Wang et al., 2019). However, they are computationally expensive (Merity et al., 2017; Yang et al., 2017) and can be difficult to scale to larger projects. The Transformer model architecture (Vaswani et al., 2017) is based on self-attention and fully connected layers without recurrence and is thus parallelizable and much cheaper to compute.

In this thesis, I have chosen to combine the Transformer architecture with Long Short-Term Memory networks and conduct the labelling process in two steps.

Language Models and Transfer Learning

Language models are large-scale probabilistic distributions over words in a language (Jurafsky & Martin, 2009) that are widely used in natural language processing tasks. They eliminate the need to train huge models every time a new task is needed: you can build upon the existing models. In transfer learning, the information learned from a previously trained large-scale language model is utilized in training a model aimed for a different task. The new model thus builds upon the pre-trained one and naturally, the selection of the pre-trained language model affects the results of the task. The representations learned from large-scale language models can be effectively fine-tuned for more specific tasks such as register classification tasks (Repo et al., 2021; Rönnqvist et al., 2021) or named entity recognition (Luoma et al., 2020).

Recently, the Transformer architecture (Vaswani et al., 2017) is utilized by many of the state-of-the-art language models such as BERT, Bidirectional Encoder Representations from Transformers (Devlin et al., 2018), which is a deep bidirectional language model that has been pre-trained with unlabelled data. Monolingual BERT models exist for example in English (Devlin et al., 2018), Finnish (Virtanen et al., 2019) and French (Le et al., 2019). Multilingual M-BERT hasn’t reached as good results as monolingual models especially with lowerresourced languages (Virtanen et al., 2019), but XLM-RoBERTa that follows Cross-lingual Language Modelling (XLM) approach by Lample and Conneau (2019) gains good results even compared to a monolingual model (Repo et al., 2021; Rönnqvist et al., 2021).

Repo et al. (2021) compared the results of a register classification task with multilingual mBert (Devlin et al., 2018) and XLM-RoBERTa large (Conneau et al., 2019) and different language-specific BERT models (Finnish FinBERT (Virtanen et al., 2019), French FlauBERT large (Le et al., 2019), Swedish KB-BERT (Malmsten et al., 2020) and English BERT large (Devlin et al., 2018)). Both in the monolingual and cross-lingual setups, the XLM-RoBERTa large performed the best when evaluating the performance with a test set. Based on its good performance in multilingual settings, I will be using XLM-RoBERTa in this thesis as the base for training the line-wise classification models, but to avoid an unnecessarily large model, I will use the base version.

5 Data and Methodology

In this chapter, I will present the data used in this thesis as well as the experimental setup behind the classification task. In addition to this, I will explain the metrics to evaluate the performance of the classification models.

Line-wise Annotated FreCORE and SweCORE

The majority of the data in this thesis consists of FreCORE and SweCORE (Repo et al., 2021), French and Swedish Corpora of Online REgisters that follow in the footsteps of CORE (Egbert et al., 2015) and FinCORE (Laippala et al., 2019). The corpora are register (genre) annotated random samples of Common Crawl (Wenzek et al., 2019) that have been run through Trafilatura version 0.3 (Barbaresi, 2020) for boilerplate removal and Onion (Pomikálek, 2011) for deduplication. The corpora have been created for automatic classification of Internet registers as they are good indicators of linguistic variation and can affect the way a text is interpreted.

As noted in Laippala et al. (2020), the cleanliness level after deduplication and cleaning with Trafilatura wasn’t sufficient and thus, the data was further annotated manually into main content and boilerplate. The annotations were chosen to be done line-wise in order to optimize the level of cleaning: since document level filtering had already been done, a lower-level boilerplate annotation was needed, but sentence or token-wise annotation task would have been much more time consuming. In this case, a line refers to a segment of text that is bordered by line breaks and can thus also be understood in a colloquial sense as a paragraph.

The main portion of the data used in this study consists of 1 982 French and 2 399 Swedish documents of 84 103 (French) / 99 406 (Swedish) manually annotated lines in total. The data has been published without the manually annotated boilerplate lines in Repo et al. (2021). In addition to the French and Swedish data, small data sets of line-wise annotations in Finnish, English, Spanish and German were added in order to train a multilingual model.

The documents were chopped into 50-line segments in order to speed up the model and improve the performance with longer documents with the goal of not letting the model see too much of the document while making predictions for each block, in case the situation of a line in the document would affect the classification. The average length per document is 43 lines in the French data, while the majority of the documents are 12 lines or less. Since the data pre-processing step for the LSTM model chops up the documents into blocks of maximum 50 lines, this would mean that in general, the model sees the whole document when making predictions. There are however a few outliers in the data that are very lengthy: while 95 % of the French documents have 140 lines or less, there are some documents that have over 4 000 lines. The average length of a document in SweCORE is 41 lines, while the majority of the documents are 18 lines or less. This means that most of the Swedish texts are treated as a whole. Compared to FreCORE, the Swedish data has fewer very long documents, but a single outlier that has 7 411 lines.

The line-wise annotations were done by several annotators with a background in linguistics following a common set of guidelines (see Appendix 1 Line-wise Annotation Guidelines) with the goal of accepting the lines that belong to the body of text extracted from a web page and rejecting all the lines that don't add to the text such as generated ads or links and meta data. Difficult cases, such as news article titles and headnotes, advertisements within the text and text in foreign language were discussed and solved within the group of annotators. The decisions might differ from earlier guidelines: for example, references and tables of content were treated as main content, as they usually provide information about the text and are considered to be typical content in texts that belong to e.g., informative register categories.

Inter-annotator-agreement is a measure of annotation quality that can be calculated by comparing the annotations made independently by different annotators (Artstein, 2017). At its simplest form, inter-annotator-agreement can be measured as an accuracy between the labels given by two annotators by counting the number of agreements and dividing the number by the number of annotations in total. This essentially returns a value between 0 and 1, where 1 signifies complete agreement. According to Artstein (2017), this might not be the most reliable measure to assess the reliability of the annotations: the accuracy doesn't consider any probability of accidental agreement. Another, generally more recommended way of calculating inter-annotator agreement is called Krippendorff's alpha, which takes into account the distance between the annotators disagreements and returns a value between -1 and 1, where 1 signifies perfect agreement and 0 an agreement level that could be achieved by coin toss. In general, values above 0.8 are accepted as reliable, but in lower level agreement is acceptable in some cases (Artstein, 2017).

As the line-wise annotation was divided between two people for the French data and seven people for the Swedish data, an inter-annotator-agreement was calculated: a sample of 127 texts (7 241 lines) in French resulted in line-wise accuracy of 0.86 and a sample of 240 texts (7 763 lines) in Swedish resulted in 0.80 accuracy between two annotators. Respectively, Krippendorff's alpha is 0.65 for French and 0.55 for Swedish data. This indicates that the annotation task wasn't trivial for human annotators. An example of a text from a genealogy portal with multiple disagreements can be seen in Table 2: here, one annotator annotated almost all the lines, including the ones that seem somewhat generated or that are duplicates of other lines, as main content, whereas the second annotator has annotated these lines as boilerplate. The one agreement on boilerplate content is the final line "Kommentar" (comments). Table 3 demonstrates the distribution of the lines annotated as main content and boilerplate.

Table 2: Example of a line-wise annotation disagreement

Annotator 1	Annotator 2	Text
Main content	Boilerplate	Rötter - din källa för släktforskning driven av Sveriges Släktforskarförbund
Main content	Boilerplate	Vet du något om fotografen? Skicka in informationen här
Main content	Boilerplate	Tillhör samma album som #199144. Inga anteckningar på fotot.
Main content	Main content	Tillhör samma album som #199144. Inga anteckningar på fotot.
Main content	Main content	Okänd man. Ur samma album som #37875 från Rossön (Y).
Main content	Main content	Bilden föreställer Emilia Lindborg född Westman. Gift med #6252 och dotter till #5906. Från album tillhörande Anna Lindborg, Gävle (=dotter).
Main content	Main content	Bilden föreställer sjökapten Brynolf Lindborg, 1847-1896. Död i Torneå. Styrmansexamen i Gävle 1870. Son till #6004. Bild från album tillhörande Anna Lindborg, Gävle.
Boilerplate	Boilerplate	Kommentar

Table 3: Annotated lines in the data

	Texts	Main content lines	Boilerplate lines	Lines total
French	1 982	57 789 (68.7 %)	26 314 (31.3 %)	84 103 (100 %)
Swedish	2 399	47 960 (48.2 %)	51 446 (51.8 %)	99 406 (100 %)
Finnish	137	2 480 (58.0 %)	1 797 (42.0 %)	4 277 (100 %)
English	191	3 360 (54.4 %)	2 812 (45.6 %)	6 172 (100 %)
Spanish	105	1 536 (50.9 %)	1 482 (49.1 %)	3 018 (100 %)
German	140	2 529 (73.2 %)	925 (26.8 %)	3 454 (100 %)
Total	4 954	115 654 (57.7 %)	84 776 (42.3 %)	200 430 (100 %)

As we can see from the Table 3, in total 31.2 % of the French and 51.8 % of the Swedish lines were boilerplate. Overall, the Swedish dataset contains the largest proportional amount of boilerplate (51.8 %), whereas the German dataset contains the least (26.8 %). The boilerplate annotation was made slightly easier by using a BERT model[12] created with the data released in Laippala et al. (2020) to make baseline predictions for the lines. An example of a text with the preliminary predictions for the line-wise labels and the corrected line-wise annotations can be seen in Tables 4 and 5.

[12] https://github.com/sronnqvist/web-text-cleaner

Table 4: Baseline predictions for line-wise classification

Predicted class	Confidence	Text
Boilerplate	0.05146	Klassisk smörgåstårta med lax & räkor | Biggans
Boilerplate	0.02162	Klassisk smörgåstårta med lax & räkor
Boilerplate	0.02399	Smörgåstårta med matjessill
Boilerplate	0.01795	Böcklingtårta med sting
Boilerplate	0.01128	Biggans klämma
Boilerplate	0.01730	Biggans fyllda krustader
Boilerplate	0.02162	Klassisk smörgåstårta med lax & räkor
Boilerplate	0.05197	Botten: 9 skivor landgångsbröd, 3 skivor per lager
Boilerplate	0.01696	2st hackade ägg
Boilerplate	0.01449	2st Böcklingpastejer
Boilerplate	0.01477	1dl hackad saltgurka
Boilerplate	0.02917	Garneringskräm:
Boilerplate	0.01378	1dl finhackad dill
Boilerplate	0.01515	100g skalade räkor
Boilerplate	0.02515	2st hackade hårdkokta ägg
Boilerplate	0.01440	1dl smörgåskrasse
Boilerplate	0.02181	1 huvud grillsallad
Main content	0.71449	Koka, skala och hacka äggen. Blanda Biggans Böcklingpastej med créme fraiche
Main content	0.97894	och gurka. Vänd sedan försiktigt ned de hackade äggen.
Main content	0.97492	Dela laxen i mindre bitar. Blanda majonnäs med senap och den förhackade dillen.
Boilerplate	0.26372	Vänd ned laxen.
Boilerplate	0.23866	Fördela fyllningen mellan de olika bottnarna. Lägg det andra lagret bröd åt motsatt
Main content	0.86247	håll från det första så tårtan inte delar sig. Skär kanterna jämna. Stryk färskost på
Main content	0.69858	sidorna och ovanpå tårtan. Fäst finhackad dill på kanterna. Garnera med laxrosor,
Main content	0.85052	löjrom, hackade ägg, smörgåskrasse och salladsblad.
Boilerplate	0.30846	Adress: Musseronvägen 3
Boilerplate	0.03289	141 23 Huddinge • Box 1097
Boilerplate	0.01864	Facebook: Besök oss nu »

Table 5: Manual line-wise annotation

Predicted class	Confidence	Text
Boilerplate	0.05146	Klassisk smörgåstårta med lax & räkor | Biggans
Boilerplate	0.02162	Klassisk smörgåstårta med lax & räkor
Boilerplate	0.02399	Smörgåstårta med matjessill
Boilerplate	0.01795	Böcklingtårta med sting
Boilerplate	0.01128	Biggans klämma
Boilerplate	0.01730	Biggans fyllda krustader
Main content	0.02162	Klassisk smörgåstårta med lax & räkor
Main content	0.05197	Botten: 9 skivor landgångsbröd, 3 skivor per lager
Main content	0.01696	2st hackade ägg
Main content	0.01449	2st Böcklingpastejer
Main content	0.01477	1dl hackad saltgurka
Main content	0.02917	Garneringskräm:
Main content	0.01378	1dl finhackad dill
Main content	0.01515	100g skalade räkor
Main content	0.02515	2st hackade hårdkokta ägg
Main content	0.01440	1dl smörgåskrasse
Main content	0.02181	1 huvud grillsallad
Main content	0.71449	Koka, skala och hacka äggen. Blanda Biggans Böcklingpastej med créme fraiche
Main content	0.97894	och gurka. Vänd sedan försiktigt ned de hackade äggen.
Main content	0.97492	Dela laxen i mindre bitar. Blanda majonnäs med senap och den förhackade dillen.
Main content	0.26372	Vänd ned laxen.
Main content	0.23866	Fördela fyllningen mellan de olika bottnarna. Lägg det andra lagret bröd åt motsatt
Main content	0.86247	håll från det första så tårtan inte delar sig. Skär kanterna jämna. Stryk färskost på
Main content	0.69858	sidorna och ovanpå tårtan. Fäst finhackad dill på kanterna. Garnera med laxrosor,
Main content	0.85052	löjrom, hackade ägg, smörgåskrasse och salladsblad.
Boilerplate	0.30846	Adress: Musseronvägen 3
Boilerplate	0.03289	141 23 Huddinge • Box 1097
Boilerplate	0.01864	Facebook: Besök oss nu »

In Tables 4 and 5, we can see an example of a recipe text pre and post annotation. The first column represents the predicted or annotated class, 1 being main content and 0 boilerplate. The second column includes the predicted probability of the line belonging into class 1. The line of text in question is in third column. According to this example, the BERT model failed to recognize 13 out of 19 rows that have been manually annotated as main content. This is quite natural since the beginning of the main content in this document perhaps isn't obvious for a non-human annotator. The main content block begins with the title "Klassisk smörgåstårta med lax & räkor" and continues with the number of servings and a list of ingredients. These kinds of short lines are very similar to many boilerplate examples, but in this case, they are a necessary part of the text: we're dealing with a recipe after all.

After the ingredients, there is the instructive part with its own problems. As previously noted in chapter 3 Boilerplate removal, the HTML marking conventions can vary significantly between different websites. Here, for some reason, it seems that the instructions are divided across several paragraphs with no regard to the sentence boundaries. This has made the rows of text to "stop" unexpectedly and caused some unconventional line breaks.

Register annotations

Different registers vary in their linguistic features and how well they can be automatically recognized. According to Repo et al. (2021), Informative and Narrative texts are more stable in their linguistic features and thus easier to categorize in general, while some categories such as Advice texts in Opinion register are much more varied and harder to identify. E-forum texts may contain non-standard linguistic structures (Biber & Conrad, 2019) and they are often tagged as containing generated text along with Encyclopedia texts and Informational Persuasion texts (Laippala et al., 2020). This variation might affect the line-wise classification as well: if the texts do not follow a certain structure, it is possible that the line-wise classifier won't perform as well as with texts that are more uniform.

The register annotation for FreCORE and SweCORE was done by two annotators with a linguistic background and following the taxonomy presented in CORE (Egbert et al., 2015), where there are eight upper register categories: Narrative, Informative, Opinion, How-to, Spoken, Informational Persuasion, Lyrical and Interactive Discussion. These upper registers contain several sub-registers, such as News article / blog in the Narrative category. In addition to this, the annotation was made richer by allowing the annotators to annotate a text with two separate register labels. The data doesn't include documents that contained only machine translated or generated text or "non-text", as in documents that consist solely of boilerplate. An inter-annotator-agreement was calculated for the Swedish (0.84 F1-score) and French (0.78 F1-score) annotations. The upper-level register dispersion can be examined in Table 6.

Please note that since one text can have more than one label, the number of labels given doesn't correspond to the number of texts.

Table 6: Upper-level register dispersion by language in FreCORE and SweCORE

	Narrative	Informative	Opinion	How-to	Informational Persuasion	Interactive Discussion	Spoken	Lyrical	Texts total
FR	581	547	180	135	596	159	26	9	1 982
SV	828	778	207	130	583	94	7	8	2 399
Total	1409	1325	387	265	1179	253	33	17	4 381

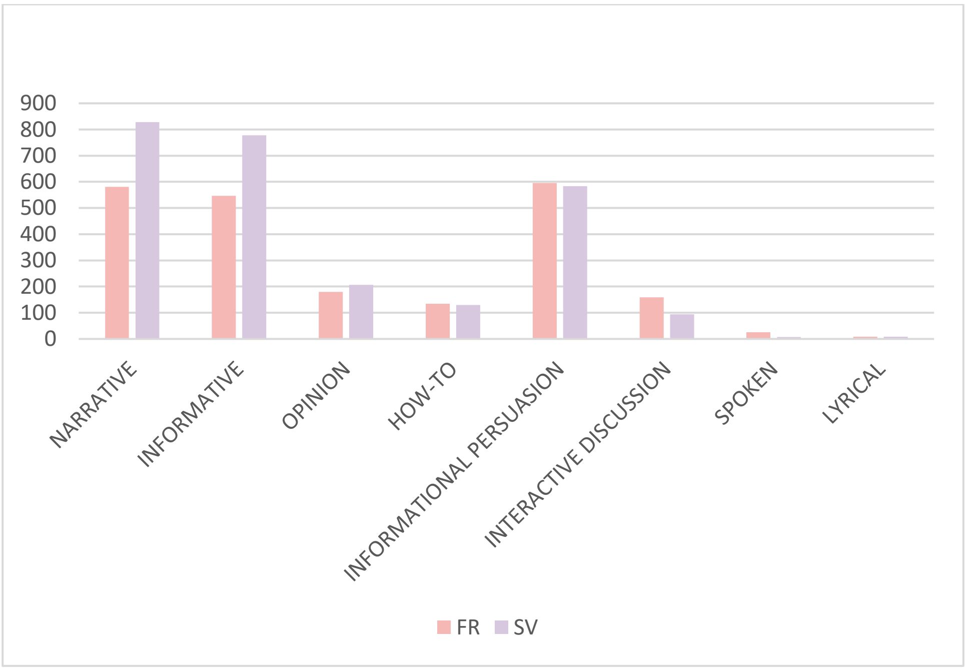


Figure 7: Register distribution in FreCORE and SweCORE

As Table 6 and Figure 7 demonstrate, the largest upper register categories in FreCORE and SweCORE combined are Narrative, Informative and Informational Persuasion. In FreCORE, the three largest categories are quite balanced between Informational Persuasion, Narrative and Informative register categories, whereas in SweCORE, the Narrative and Informative register categories are much larger than the third largest Informational Persuasion category. It should also be noted that the French data contains relatively more texts in Interactive Discussion and Spoken categories than the Swedish data.

On lower level, the five largest sub-registers in FreCORE are Description with intent to sell (n = 367) from Informational Persuasion, News article / blog (n = 232) from Narrative, Description of a thing (n = 130) and Encyclopedia article (n = 119) from Informative and Discussion forum (n = 110) from Interactive Discussion register categories. In SweCORE, the largest sub-registers are Encyclopedia article (n = 507) and Description of a thing (n = 94) from Informative, Description with intent to sell (n = 396) from Informational persuasion as well as News article / blog (n = 218) and Personal blog (n = 364) from Narrative categories. It should be noted that some of these sub-registers make up a great proportion of the upper-level category: for example, 65 % of the Informative category in SweCORE consists of Encyclopedia articles. Respectively, 65 % of the Informational Persuasion category is made up from Description with intent to sell. There are also some language specific differences: the Swedish data contains more Personal blog texts whereas the French data contains more Discussion forum texts.

Experimental Setup

The task at hand is two-fold: the first step is a many-to-one sequence classification task to determine a preliminary label for a line, the second is a many-to-many sequence labelling task where both the input, a sequence of line embeddings, and output, the final line labels, are of the same length. The data consists of sequences of sequences, as in lines of text. Following the previous boilerplate removal approaches (Leonhardt et al., 2020; Pomikálek, 2011; Vogels et al., 2018), the hypothesis is that a text contains blocks of main content and blocks of boilerplate. When each line is given a label, it is presumed that boilerplate lines appear among other boilerplate lines and main content lines appears among other main content lines.

In this study, transfer learning will be utilised by training a multilingual XLM-RoBERTa-base (Conneau et al., 2019; Lample & Conneau, 2019) model that has been pre-trained on 2.5TB of CommonCrawl data in 100 languages. The model will be first trained with the line-wise annotated data to retrieve a preliminary line-wise classification for each line independent from the surrounding lines. The data is run through an XLM-RoBERTa classifier with a fully connected decision layer on top of the pooled pretrained weights

In this case, the hypothesis is that by taking into account the surrounding lines, the line-wise classification task performance can be improved. To consider both the previous and following lines, a bidirectional LSTM layer and a fully connected decision layer are applied to the word embeddings retrieved from the last hidden layer of the Transformer model trained in the first step. In this step, a dropout of 0.15 is used to prevent overfitting the model. I will refer to this second model as LSTM model, although it is also built on top of the XLM-RoBERTa-base. A visualization of the full process can be viewed in Figure 8.

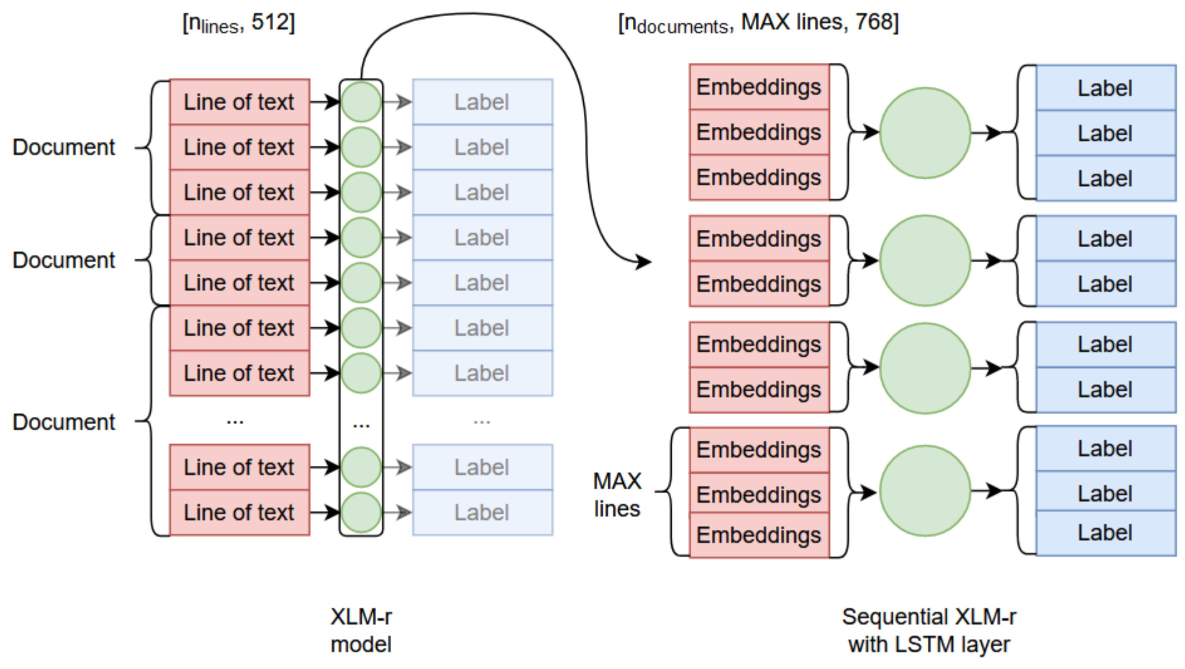


Figure 8: Line-wise classification model training process

As you can see in Figure 8, an input of *n* lines times 512 tokens is fed to the first model. In this first step, document boundaries are ignored, and lines are given label predictions independently. Instead of these label predictions, the second model takes the word embeddings from the first models last hidden layer as its input. These embeddings are split according to the document boundaries unless the document is longer than the separately defined maximum number of lines (MAX lines in the figure), in which case the document is split into segments that hold the maximum number of lines at most. The input's sequence length is 768 that corresponds to the vocabulary size of XLM-RoBERTa-base. Finally, these segments of embeddings are fed into the second model and labelled.

The XLM-R model is trained with learning rate of 2e-6 that was chosen after training the Swedish and French models with a set of learning rates[13] and maximum number of lines per document of 250. As mentioned in the previous chapter, Line-wise Annotated FreCORE and SweCORE, for the LSTM model a maximum number of lines was set to 50 in order to make the training lighter computation-wise. This also could prevent the model from learning e.g., that boilerplate exists only in the beginning or at the end of a document, especially with longer documents. For the LSTM models, the learning rates were chosen individually among

[13] {[1..9]e-5, [1..9]e-6, [1..9]e-7}

the same values as with the XLM-R model. Both the XLM-R and LSTM models used accuracy as a metric and cross entropy as a loss function.

The data is split into train, validation and test sets with splits of 0.7, 0.15 and 0.15 respectively. This split was chosen because it had been used with previous experiments with the multilingual data. Note that the data is split by documents, not by line numbers, and it is not stratified by registers. This means that the number of lines in a set might not correspond to the proportional number of lines in the data and that the register distribution within a set might not represent the register distribution in the whole data.

The code used in this study can be found on Github[14] and it is based on transformers classifiers by Samuel Rönnqvist[15] and Sampo Pyysalo[16]. The classifier is built with Tensorflow 2.4 Keras in Python 3, with libraries Transformers, Sklearn and Numpy among others. The computation was made possible by Finnish IT center for science[17] and their supercomputer Puhti.

The main metric used in the evaluation is accuracy, on top of which F1-score is used in order to be able to compare the results with previous studies. Because the classification only has two possible classes, it is easy to calculate the accuracy based on the number of true positives (in this case main content lines predicted as main content), true negatives (boilerplate lines predicted as boilerplate), false positives (boilerplate lines predicted as main content) and false negatives (main content lines predicted as boilerplate). These values can be used to compose confusion matrices and different evaluation metrics, described below.

Accuracy is the proportional amount of correctly predicted labels in the whole data. Precision (or sensitivity in binary classification tasks) signifies the proportional amount of correctly labelled predictions, in other words the percentage of labels that the model predicted positive that actually are positive. Recall tells the proportional amount of positive labels that were found in the prediction process. (Jurafsky & Martin, 2009) These metrics are summarized in Table 7.

---

[14] https://github.com/annsaln/transformers-classifier
[15] https://github.com/sronnqvist/transformers-classifier
[16] https://github.com/spyysalo/transformer-text-classifier
[17] https://www.csc.fi/en/home

Table 7: Confusion matrix visualisation of different evaluation metrics for binary classification

	Positive	Negative	
Predicted positive	True positive	False positive	precision = $\frac{tp}{tp+fp}$
Predicted negative	False Negative	True negative	
	recall = $\frac{tp}{tp+fn}$		accuracy = $\frac{tp+tn}{tp+fp+tn+fn}$

In addition to accuracy, precision and recall, F-score is used to evaluate performance especially in cases when the label distribution is unbalanced. It is a weighted harmonic mean of precision and recall that can be scaled to favour either precision of recall, but the most often used F-score is F1-score that is balanced. (Jurafsky & Martin, 2009) F1-score can be calculated with the following formula:

$$F1 = \frac{2 * precision * recall}{precision + recall}$$

This calculates an F1-score for a label. Because the label distribution in my data is relatively balanced, I will use accuracy as my main evaluation metric as it is easy to interpret. However, since most of the previous applications in boilerplate detection have used F1-score in their evaluation, I will also calculate this metric to be able to compare my result with the previous achievements.

I will use statistical tests to compare the performance of different models in two ways: one-sample t-test for comparing the different builds of the models (XLM-R vs. LSTM) to test whether the difference in the means of accuracy in said models is statistically significant and one-proportion z-test for comparing different models with the same build.

6 Analysis

This chapter will comprise of

a. presenting the results of the classification

b. comparison of the results of the different XLM-R and LSTM models and inspection of register-wise performance

c. error analysis (What does the classifier learn and where it goes wrong?)

d. discussion of the results.

Results of the classification

As mentioned in chapter 5.2 Experimental setup, the first step in the process was to train an XLM-R classifier to predict the classes of the lines. This process was repeated to create three models: monolingual Swedish and French models and a multilingual model based on all the datasets. The multilingual model was trained with all the languages and tested language specifically in addition to the overall evaluation. As an experiment, zero-shot models were also trained, where the model was tested on a language that wasn't included in the training and validation data. The setup is similar to (Rönnqvist et al., 2021), where the model is trained with data that includes all languages but the test language. The classification results of said models can be viewed in Table 8, where the top row indicates the tested language and the left-most column the tested model. The results present the mean accuracy (M) of three repetitive training instances of the model as well as standard deviation (SD). The cells representing the LSTM models include the t-value of the one-sample t-test comparing the LSTM model with the equivalent XLM-R model: the larger the value, the larger the difference between the compared models.

Table 8: The classification results of the models (N=3) in monolingual and multilingual settings

	Swedish	French	English	Finnish	German	Spanish	Multilingual
Monolingual XLM-R	M = 0.84 SD = .034	M = 0.82 SD = .004					
Multilingual XLM-R	M = 0.84 SD = .002	M = 0.85 SD = .020	M = 0.82 SD = .044	M = 0.80 SD = .020	M = .70 SD = .017	M = 0.69 SD = .037	M = 0.84 SD .011
Zero-shot multilingual XLM-R	M = 0.73 SD = .008	M = 0.81 SD = .015	M = 0.76 SD = .018	M = 0.64 SD = .267	M = 0.66 SD = .513	M = 0.73 SD = .099	
Monolingual LSTM	M = 0.92 SD = .001 t(2) = 26.4	M = 0.84 SD = .005 t(2) = 7.46					
Multilingual LSTM	M = 0.88 SD = .007 t(2) = 9.98	M = 0.89 SD = .004 t(2) = 32.3	M = 0.89 SD = .01 t(2) = 12.4	M = 0.86 SD = .004 t(2) = 24.3	M = 0.73 SD = .016 t(2) = 2.72	M = 0.70 SD = .029 t(2) = 0.42	M = 0.88 SD = .002 t(2) = 45
Zero-shot multilingual LSTM	M = 0.80 SD = .004 t(2) = 25.9	M = 0.84 SD = .002 t(2) = 21.8	M = 0.71 SD = .005 t(2) = -14.8	M = 0.85 SD = .002 t(2) = 228	M = 0.77 SD = .02 t(2) = 9.71	M = 0.88 SD = .018 t(2) = 14.6	

As seen in Table 8, The mean accuracies are 0.84 for the monolingual Swedish XLM-R model and 0.82 for the monolingual French XLM-R model. The overall mean accuracy for the multilingual XLM-R model is 0.84, with accuracies of 0.84, 0.85, 0.82, 0.80, 0.70 and 0.69 for Swedish, French, English, Finnish, German and Spanish. The mean zero-shot accuracies for the multilingual model are 0.73, 0.81, 0.76, 0.64, 0.66 and 0.73 for Swedish, French, English, Finnish, German and Spanish respectively. A surprising result was that the French data tested better with the multilingual XLM-R model than the monolingual French one (0.85 accuracy vs 0.82 accuracy) and didn't suffer much in performance when tested with the multilingual zero-shot model. Another observation is that the Spanish test evaluation on average improved from multilingual model to the zero-shot setup, where the model hadn't seen said language at all. This could imply that there's something weird going on with the Spanish data.

Regarding the LSTM model training results found in Table 8, the monolingual Swedish LSTM model has a mean test accuracy of 0.92 whereas the monolingual French model has a mean test accuracy of 0.84. In multilingual tests, the model was trained and tested with these languages together with sample sets of German, Spanish, English and Finnish and the performance is similar with total accuracy of 0.88. Zero-shot experiments where the model is tested on a language that is not included in the training data have been slightly less successful:

the mean accuracies are 0.80, 0.84, 0.71, 0.85, 0.77 and 0.88 for Swedish, French, English, Finnish, German and Spanish. However, it should be noted that the variance in the zero-shot tests with the multilingual LSTM model are smaller than with the XLM-R model, indicating a more stable performance.

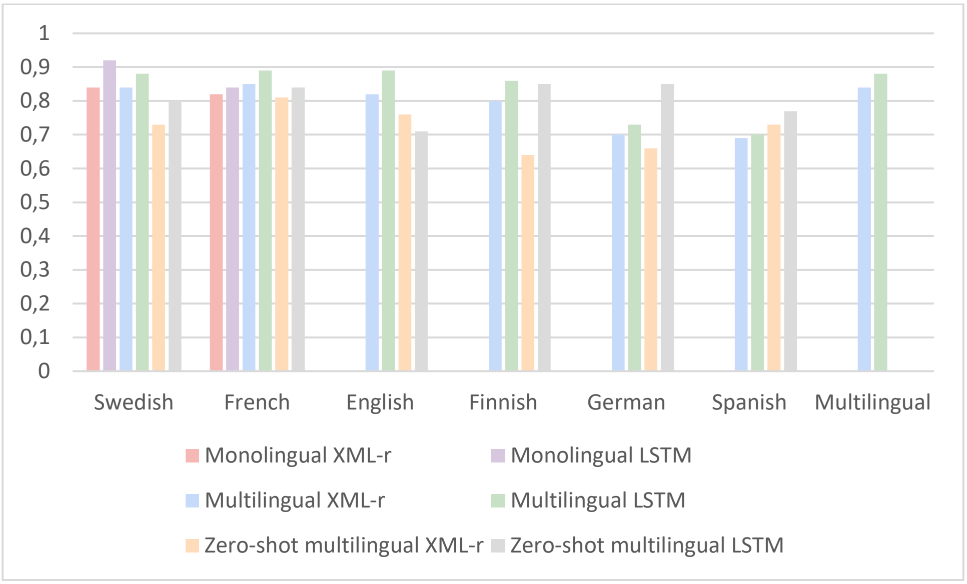


Figure 9: Visualization of the performance of the different models

From Table 8 and its corresponding visualization in Figure 9, it is clear that the LSTM model improves the results from the XLM-R model. The improvement is statistically significant in all the models (p < 0.05 with two-tailed one-proportion t-test) except in the multilingual model when tested with German and Spanish and with the English test set evaluation in zero-shot setup, where the XLM-R model performed better than the LSTM model (0.76 mean accuracy vs 0.71 mean accuracy, p < 0.05). The greatest jump in performance is with the Finnish data in zero-shot setting, where originally a mean accuracy of only 0.64 was reached and improved to 0.85 with the LSTM model. Overall, the multilingual model accuracy went up from 0.84 to 0.88.

Similar to the XLM-R model, Spanish data doesn't suffer in performance when tested in multilingual zero-shot setup. In addition to this, the German data tests better with the zero-shot model than with the multilingual model. This further indicates that there might be some problems with the Spanish and German datasets. Like with the XLM-R model, the French data tests worse with the monolingual French model than with the multilingual one, this time with no drop between monolingual and zero-shot multilingual. Please note that from now on, I will refer to the LSTM model performance when discussing these models. A further summary of the results can be found in Table 9, where the shown evaluation results are means of the three training instances.

Table 9: Summary of the test evaluation results of the LSTM models

	Accuracy	Main content F1-score	Boilerplate F1-score	F1-score
French monolingual	0.84	0.88	0.72	0.84
Swedish monolingual	0.92	0.91	0.93	0.92
Multilingual	0.88	0.90	0.86	0.88

As seen in Table 9, both the French and multilingual model perform better with the main content lines, whereas the Swedish monolingual model performs slightly better with the boilerplate lines. The greatest difference between the identifying the two labels is with the French model, where the F1-score for main content lines is 0.88 and 0.72 for the boilerplate lines.

Comparing the results to previous achievements is somewhat tricky due to the different format of the data: most of the existing boilerplate removal application work with HTML tags and thus cannot be directly applied into this dataset that is in plain text format, where each line is labelled either as main content or boilerplate, and that has already been cleaned based on the HTML tags. In addition to this, there is no one common evaluation dataset that would have been tested with all the other applications. The most common one for evaluation is Cleaneval (Baroni et al., 2008) that has been used in evaluation of jusText (Pomikálek, 2011), Web2Text (Vogels et al., 2018) and BoilerNet (Leonhardt et al., 2020) resulting in F1-scores 0.9421, 0.88 and 0.87 respectively. Trafilatura (Barbaresi, 2021) has been evaluated with a dataset of web-sourced documents mainly in German reaching 0.914 accuracy and 0.912 F-score. Barbaresi has used this dataset to evaluate other applications as well: of the ones mentioned in this thesis, jusText reaches accuracy of 0.749 and f-score 0.699, suggesting that there is some performance loss with this dataset compared to the Cleaneval one. When comparing these numbers to the performance of the LSTM models of this thesis, the monolingual Swedish model outperforms these numbers and the multilingual model isn't far behind.

Register-wise Performance

As the different text types can vary greatly in their features, I will further examine the linguistic variation's effect on the line-wise classification performance with FreCORE and SweCORE. This part of the analysis is restricted only to the French and Swedish parts of the data, since the texts in English, Finnish, Spanish and German do not contain any register labels. As noted in Chapter 5, FreCORE and SweCORE differ in their register dispersion. Despite the smaller size of the dataset, FreCORE holds more texts in How-to, Informational Persuasion, Interactive Discussion and Spoken categories, whereas the larger SweCORE contains more Narrative, Informative and Opinion texts. I will examine only the eight upper registers: Narrative (NA), Informative (IN), Opinion (OP), How-to (HI), Informational Persuasion (IP), Interactive Discussion (ID), Spoken (SP) and Lyrical (LY).

The different register distribution can influence the performance of the models: the linguistic features vary between registers, and for some registers, non-standard language or generated text are more typical than for others. For example in Laippala et al. (2020), Discussion forum texts, Encyclopedia texts and Informational Persuasion texts were often tagged as containing generated text and Biber and Conrad (2019) note that e-forums contain some non-standard linguistic structures. In addition to this, some register categories such as Lyrical contain very few texts (9 in French and 8 in Swedish), and it can be presumed that the line-wise classifier will not perform well with these kinds of texts. Table 10 demonstrates the averaged document level test accuracies per upper register category in the monolingual French and Swedish LSTM models with standard deviation in parenthesis.

Table 10: Mean LSTM classification accuracies per upper register category with monolingual models

	NA	IN	OP	HI	IP	ID	SP	LY
French	M = .92 (.005)	M = .89 (.025)	M = .86 (.012)	M = .87 (.026)	M = .82 (.019)	M = .88 (.022)	M = .87 (.084)	M = .75 (.059)
Swedish	M = .93 (.002)	M = .93 (.006)	M = .95 (.009)	M = .90 (.004)	M = .88 (.007)	M = .97 (.008)	M = .95 (.002)	M = .45 (.042)

As shown in Table 10, the monolingual Swedish model performs better ($p < 0.05$ with two-tailed one-proportion Z-test) than the French model in all the other upper register categories than Lyrical, where the averaged accuracy is considerably higher with the French model. Despite FreCORE having more texts in Informational Persuasion, Interactive Discussion and Spoken categories, the Swedish model still outperforms the French one. The Narrative register category however is high in both cases despite the difference in the size of the data. This could mean that Narrative texts are in general uniform in their formatting and contain little noise.

Table 11: Mean LSTM classification accuracies per upper register category with the multilingual model

	NA	IN	OP	HI	IP	ID	SP	LY
French	M = .91 (.007)	M = .89 (.005)	M = .87 (.003)	M = .90 (.005)	M = .87 (.003)	M = .90 (.004)	M = .94 (.006)	M = .58 (.032)
Swedish	M = .91 (.003)	M = .90 (.011)	M = .94 (.001)	M = .88 (.002)	M = .86 (.003)	M = .93 (0005)	M = .93 (.006)	M = .64 (.023)
Total	M = .91 (.002)	M = .90 (.008)	M = .91 (.006)	M = .89 (.004)	M = .86 (.002)	M = .91 (.005)	M = .93 (.005)	M = .60 (.019)

Table 11 shows that the register-wise accuracies don't differ that much between the French and Swedish texts with the multilingual model. Compared to the results in Table 10, it can be seen that with the multilingual model, there is an overall improvement in the French document level test accuracies, while a slight decrease in performance with the Swedish texts. The Lyrical register category is a clear outlier with few examples in the data and poor classification accuracy, but the differences between the French and Swedish monolingual models are levelled out in the multilingual model: when the monolingual French model had a mean accuracy of 0.75 for Lyrical texts and the Swedish model's mean accuracy for them was 0.45, the overall mean average test accuracy in the multilingual is 0.61.

Error Analysis

In this sub-chapter, I will further examine the three models and their performance and provide a brief analysis of the models' errors. I will use the best performing LSTM models of the three training instances as the base for the examination.

Monolingual French Model

As stated in the previous chapter, the monolingual French LSTM model has an overall accuracy of 0.84. It performs a little better with main content lines than with boilerplate lines: a confusion matrix describing the performance can be viewed in Table 12, where you can find the number of predictions that correspond to "true positive" (main content predicted as main content), "false positive" (boilerplate predicted as main content), "false negative" (main content predicted as boilerplate) and "true negative" (boilerplate predicted as boilerplate) as well as the corresponding percentages of overall number of predictions and label-wise precision and recall values.

Table 12: Confusion matrix of the monolingual French model test set performance

	Main content	Boilerplate	
Predicted main content	12 837 (58.31 %)	1 726 (7.84 %)	Precision = 0.88
Predicted boilerplate	2 335 (10.61 %)	5 116 (23.24 %)	Precision = 0.69
	Recall = 0.85	Recall = 0.75	

As seen in Table 12, precision and recall are 0.88 and 0.85 for main content and 0.69 and 0.75 for boilerplate with the test dataset. This amounts to F1-scores 0.86 and 0.72 for main content and boilerplate, respectively. The test dataset is a bit unbalanced: of 22 014 lines, only 6 842 lines (31 %) are annotated as boilerplate, whereas 15 172 (69 %) are annotated as main content. This is in line with the overall label distribution in the French data: in the whole data, 31.3 % of the lines are annotated as boilerplate and 68.7 % as main content. This could explain the better performance with main content lines with the monolingual model as well as the performance improvement when testing the multilingual model with the French data.

As mentioned in Chapter 5, the documents are chopped into 50-line segments for the LSTM model, most of the texts having fewer than 50 lines. In order to further examine the effect of the length of the document and the significance of the position of a line, I calculated average line-wise accuracies per line number: the times the model predicted a label correctly divided by the number of occurrences of said line number in the data. A visualisation of these average line-wise accuracies of the predictions made by the monolingual French LSTM model can be viewed in Figure 10.

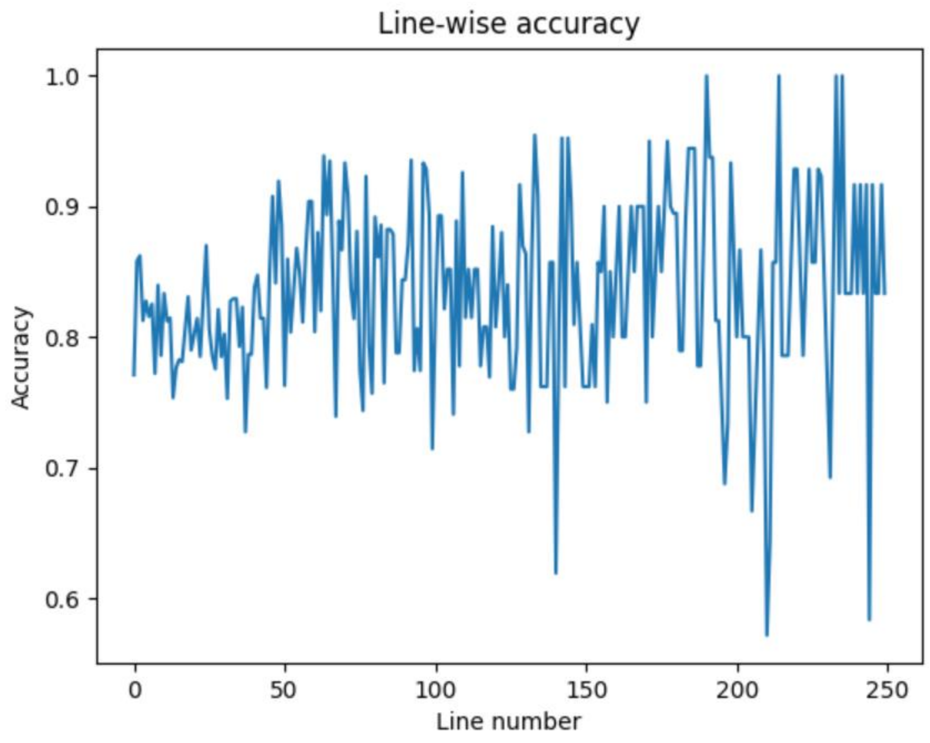


Figure 10: A graph of average line-wise test accuracy with the monolingual French model

In Figure 10, it is visible that the average accuracy per line is quite stable with reasonable variation up until the 50th line or so. After this, the variation in accuracies grows. This could be explained by the small number of documents that contain over 50 lines. Another hypothesis is that longer documents differ in quality from the shorter documents: they might have more lines of generated text of non-standard language, for example.

In general, the model performs well enough, but it has trouble with some cases. For example, documents that contain lots of generated text often contain more errors in the line-wise predictions. Table 13 demonstrates an example of an Informational Persuasion text that contains generated text that has been predicted as main content.

Table 13: An example of an Informational Persuasion text prediction

Predicted class	Confidence	Annotated class	Text
Main content	0.81988	Boilerplate	{"newsletter-popup-email":{"valueMissing":"L'adresse e-mail est obligatoire.","typeMismatch":"Entrez une adresse e-mail valide.","patternMismatch":"Entrez une adresse e-mail valide."},"newsletter-footer-email":{"valueMissing":"L'adresse e-mail est obligatoire.","typeMismatch":"Entrez une adresse e-mail valide.","patternMismatch":"Entrez une adresse e-mail valide."},"newsletter-email":{"valueMissing":"L'adresse e-mail est obligatoire.","typeMismatch":"Entrez une adresse e-mail valide.","patternMismatch":"Entrez une adresse e-mail valide."}}
Main content	0.74452	Boilerplate	SPRING SUMMER 2020
Main content	0.77610	Boilerplate	HOMME FEMME
Main content	0.79154	Boilerplate	WORKWEAR COLLECTION
Main content	0.70021	Boilerplate	SPRING SUMMER 2020
Main content	0.74098	Boilerplate	HOMME FEMME
Main content	0.75266	Boilerplate	WORKWEAR COLLECTION
Boilerplate	0.47736	Main content	Livraisons standard gratuite
Main content	0.67874	Main content	BE BRAVE #stayhome
Main content	0.82889	Main content	Chères toutes et tous, Pour faire face à cette situation difficile, l’ensemble de nos boutiques en France ont été fermées. Le site internet reste évidemment ouvert et pour toute commande la livraison en France est gratuite.
Main content	0.59804	Main content	DIESEL ♡ YOU
Boilerplate	0.37388	Main content	Nous avons prolongé notre période de retour: 30 jours

In this example, the annotator has annotated the main content being the last five lines of the document informing the customer about the changed delivery and return protocols of a web store. Before these lines, the document contains a few lines of generated text, a list of links and a generated error message, that have all been predicted as main content. Even though the model’s confidence on a line being main content is the highest on the line that actually is main content (and in fact the majority of the whole text), it is almost as sure of a line that contains the json-format error message being main content.

Another common case of misclassified lines can be seen in Table 14, where there is an example of a Discussion Forum text.

Table 14: An example of a Discussion Forum text prediction

Predicted class	Confidence	Annotated class	Text
Boilerplate	0.18945	Boilerplate	Suivez-nous : ForumEspace achatAvisPetites annoncesTestsActualitéGuidesInterviewsChroniquesCult ure Guitare
Boilerplate	0.32788	Main content	Qui a d�j� travaill� chez europ assistance??? Auteur
Boilerplate	0.16132	Main content	Inscrit le: 04 Oct 03
Main content	0.75944	Main content	Je ne sais pas vraiment en quoi cela consiste pour les boulots �tudiants.
Boilerplate	0.47713	Main content	quels sont les diff�rents crit�res � avoir?
Boilerplate	0.13512	Boilerplate	Haut Evaluer ce post : 0 cryfingers
Boilerplate	0.07891	Main content	Inscrit le: 04 Oct 03
Boilerplate	0.40243	Main content	pk vous regardez mais vous r�pondez pas?????
Boilerplate	0.10623	Boilerplate	Haut Evaluer ce post : 0 tombordo
Boilerplate	0.09376	Main content	Inscrit le: 27 Nov 04
Main content	0.52989	Main content	Nous sommes une famille d'h�telier, et l'�t� on a de temps en temps affaire � europe assistance...
Main content	0.73537	Main content	Une des branches possibles est sans doute le standard t�l�phonique, avec la prise en charge du sinistr� (recherche de logement, envoie de taxi etc)
Boilerplate	0.13822	Boilerplate	Haut Evaluer ce post : 0 cryfingers
Boilerplate	0.08643	Boilerplate	Inscrit le: 04 Oct 03

In this example, there are several problems regarding the quality of text: there are some encoding errors as well as some non-standard language. The annotators decided to include publishing dates of comments on a Discussion Forum in main content to better distinguish the comments between different authors. The model, however, doesn’t usually predict them as main content. In addition to this, some of the comments are also predicted as boilerplate.

Monolingual Swedish Model

As mentioned in Chapter 6.1., the monolingual Swedish LSTM model has a 0.92 overall accuracy. Further inspection of the model's performance can be viewed in Table 15 where you can find the number of predictions that correspond to "true positive" (main content predicted as main content), "false positive" (boilerplate predicted as main content), "false negative" (main content predicted as boilerplate) and "true negative" (boilerplate predicted as boilerplate) and the corresponding percentages of overall number of predictions and label-wise precision and recall values.

Table 15: Confusion matrix of the monolingual Swedish model test set performance

	Main content	Boilerplate	
Predicted main content	4 928 (41.37 %)	482 (4.05 %)	Precision = 0.91
Predicted boilerplate	461 (3.87 %)	6 040 (50.71 %)	Precision = 0.93
	Recall = 0.91	Recall = 0.93	

The Swedish monolingual model performs better with boilerplate lines with precision and recall 0.93, but the main content performance isn't far behind with precision and recall of 0.91. The model's F1-scores for boilerplate and main content are 0.93 and 0.91 respectively. The Swedish test dataset has 11 911 lines total, of which 6 522 (55 %) are boilerplate and 5 389 (45 %) main content. This is in line with the label distribution in the Swedish data as a whole, even though the test set leans a bit more towards boilerplate. A graph visualizing the model's average accuracies per line can be viewed in Figure 11.

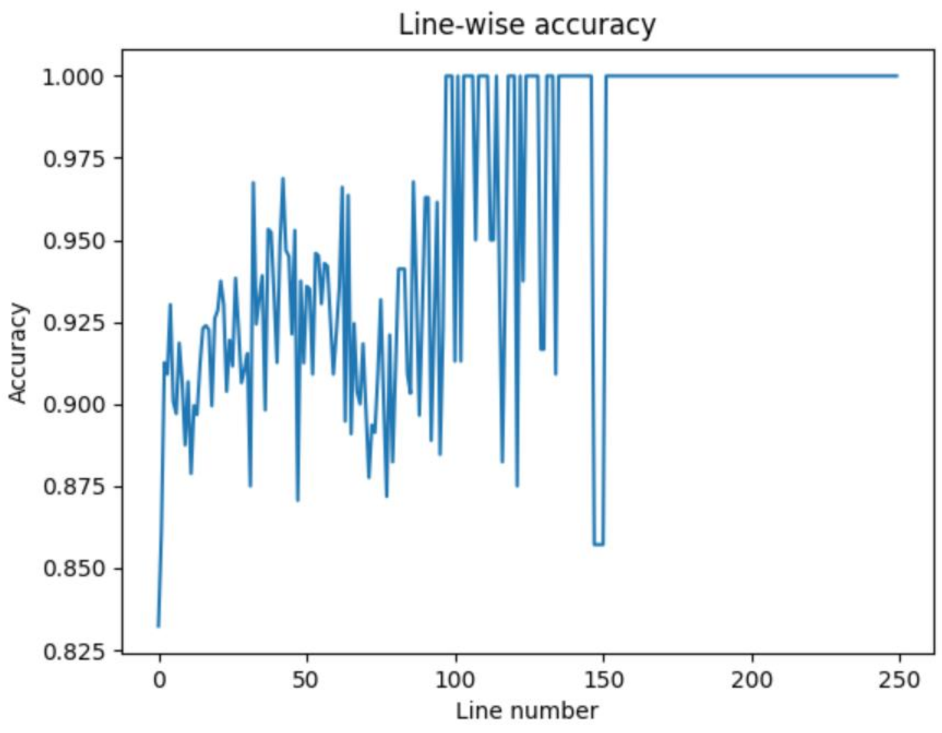


Figure 11: A graph of average line-wise test accuracy with the monolingual Swedish model

As seen in Figure 11, the Swedish monolingual model's average accuracies per line are quite high overall, averaging between 0.825 and 1. There is a drop in the average accuracy of the first lines of the documents in the test set, otherwise the line-wise accuracies hover over 0.85 and with documents containing over 150 lines averaging to 1. Compared to the French model, the monolingual Swedish model's performance with longer documents seems much more stable.

In general, the model performs very well. As with the French, the most notable performance fail is with texts that contain non-standard language, the most notable category being Lyrical texts: they are few in the whole data. Table 16 shows an example of a Narrative and Lyrical hybrid text, where the narrative parts are mostly correctly predicted, but the classification fails with the Lyrical part.

Table 16: An example of a Narrative and Lyrical hybrid text classification

Predicted class	Confidence	Annotated class	Text
Main content	0.68182	Main content	Den jag kunde va
Main content	0.66489	Boilerplate	Den jag kunde va (Till Björn Afzelius)
Main content	0.97430	Main content	Jag träffade Björn Afzelius första gången sommaren 1970. Vi spelade i Hoola Bandoola Band tillsammans till 1976 när bandet upplöstes. Vi bodde i samma hus först på Föreningsgatan och senare på Rönneholmsvägen i Malmö tills Björn flyttade till Göteborg 1977. Från 1982 när vi för första gången spelade ihop igen i TV-programmet "Måndagsbörsen" till 1996 när vi återförenade Hoola gjorde vi hundratals spelningar tillsammans. Vi uppträdde i Sverige, Danmark, Norge, Finland och på Färöarna men också i Italien, Nicaragua och Kuba.
Main content	0.97268	Main content	Vi semestrade tillsammans med våra familjer. Vi åt och drack. Vi pratade politik, affärer och kärlek. Vi skrattade. Vi grälade. Vi stod varandra kort sagt mycket nära.
Main content	0.95364	Main content	1997 i september kom Björn ner till Malmö från Göteborg och berättade att han hade fått lungcancer. När han hade åkt tillbaka skrev jag "Den, jag kunde va". Jag spelade den för honom nästa gång vi sågs. Han bad mig spela den på hans begravning. Det gjorde jag. (Kommentar från "Sånger i tiden", 2001)
Boilerplate	0.48057	Main content	Den jag kunde va på persiska Över vida oceaner
Boilerplate	0.49381	Main content	emot fjärran horisonter
Boilerplate	0.18734	Main content	över hav och kontinenter
Boilerplate	0.33198	Main content	genom skymningar och dagrar
Boilerplate	0.48876	Main content	har vi färdats med varandra
Boilerplate	0.08250	Main content	Vi har vandrat samma vägar
Boilerplate	0.07195	Main content	Vi har burit samma bördor
Boilerplate	0.09718	Main content	Vi har sett mot samma stjärnor
Boilerplate	0.07243	Main content	Vi har sjungit samma sånger
Boilerplate	0.11121	Main content	Vi har delat samma drömmar
Boilerplate	0.06379	Main content	Genom månader och år

Predicted class	Confidence	Annotated class	Text
Boilerplate	0.24108	Main content	Du är med mej alla dar
Boilerplate	0.26582	Main content	Du är den, jag kunde va
Boilerplate	0.15710	Main content	Som broar över djupen
Boilerplate	0.21554	Main content	som skuggor under träden
Boilerplate	0.22675	Main content	har vi varit för varandra
Boilerplate	0.14627	Main content	Vi har delat samma minnen
Boilerplate	0.13315	Main content	Vi har burit samma längtan
Boilerplate	0.20377	Main content	Vi har sett med samma ögon
Boilerplate	0.25278	Main content	Vi har trott på samma löften
Boilerplate	0.30443	Main content	Och ingenting kan splittra oss
Boilerplate	0.43768	Main content	och ingenting kan söndra oss
Boilerplate	0.48884	Main content	och ingenting kan slita oss isär
Boilerplate	0.35388	Main content	Skuggor kanske slukar oss
Boilerplate	0.46116	Main content	Sorger kanske tvingar oss på knä
Boilerplate	0.45233	Main content	Men ingenting i världen
Main content	0.68628	Main content	kan lösa våra band
Main content	0.68112	Main content	Du är med mej där jag är
Main content	0.75827	Main content	Du är med mej vart jag ser Du är med mej alla dar
Main content	0.79766	Main content	Du är den, jag kunde va

The model consistently misclassifies the lyrical lines that have some non-standard language in addition to the lyrical form. Another non-standard language example where the model fails is comments in a blog post, where the blog post part consists solely of a playlist (a list of songs and their performers). This example can be seen in Table 17. Due to the length of the blog post on the first line, it has been abbreviated.

Table 17: An example of the line-wise classification of a blog text with comments

Predicted class	Confidence	Annotated class	Text
Boilerplate	0.39953	Main content	Beat Happening - Revolution Come and Gone The Velvet Underground - After Hours Melody Dog - Don't Worry Baby Slow Down Tallahassee - U R Grace U R Knight School - Pregnant Again Stereolab - The Light That Will Cease To Fail The Horrors - Sea With A Sea Phil Wilson - It's A Rainy Day – –
Boilerplate	0.46052	Main content	3 comments:
Boilerplate	0.15503	Main content	Vad fan, dansar ni tryckare på era fester. Vad är Malmö för ett ställe egentligen? Stadshotellet i Tranås, änna...
Boilerplate	0.18776	Main content	Dom hånglar juh!
Boilerplate	0.16956	Main content	På dansgolvet! Vad är det, gymnasiedisco!?
Boilerplate	0.05223	Boilerplate	Post a Comment

As we can see from Table 17, all the annotated main content lines in this example have been predicted as boilerplate. This is understandable for the first line that contains the blog post, because it consists of song listings only. For some reason, the model's confidence on this line is higher than for the comments following it, even though these lines contain natural language. The language is non-formal, however, which could mean that the model has problems with non-standard Swedish in general.

Multilingual Model

As already noted in Chapter 6.1, the multilingual model has an overall accuracy of 0.88. Like with the analysis of the monolingual models, a further inspection of evaluation can be viewed in Table 18 that includes the confusion matrix of the model performance, that is the number of true positives (main content predicted as main content), false positives (boilerplate predicted as boilerplate), false negatives (main content predicted as boilerplate) and true negatives (boilerplate predicted as boilerplate), and their corresponding percentages of the all the predicted labels. In addition to this, the table includes calculated precision and recall for both main content and boilerplate.

Table 18: Confusion matrix of the multilingual model test set performance

	Main content	Boilerplate	
Predicted main content	19 571 (52.51 %)	1 840 (4.94 %)	Precision = 0.91
Predicted boilerplate	2 627 (7.05 %)	13 230 (35.50 %)	Precision = 0.83
	Recall = 0.88	Recall = 0.88	

The multilingual test set has 37 268 lines in total, of which 15 070 (40 %) are boilerplate and 22 198 (60 %) main content. Approximately the same distribution as in the whole data, with a slight bias towards main content lines. The model performs better with main content with precision of 0.91 and recall of 0.88. The recall for boilerplate lines is also 0.88, and precision 0.83. The model's F1-scores are thus 0.91 for main content and 0.86 for boilerplate. Overall, looking at the numbers, the model performs better than the monolingual French one, but slightly worse than the monolingual Swedish one. A visualisation of the line-wise performance can be viewed in Figure 12.

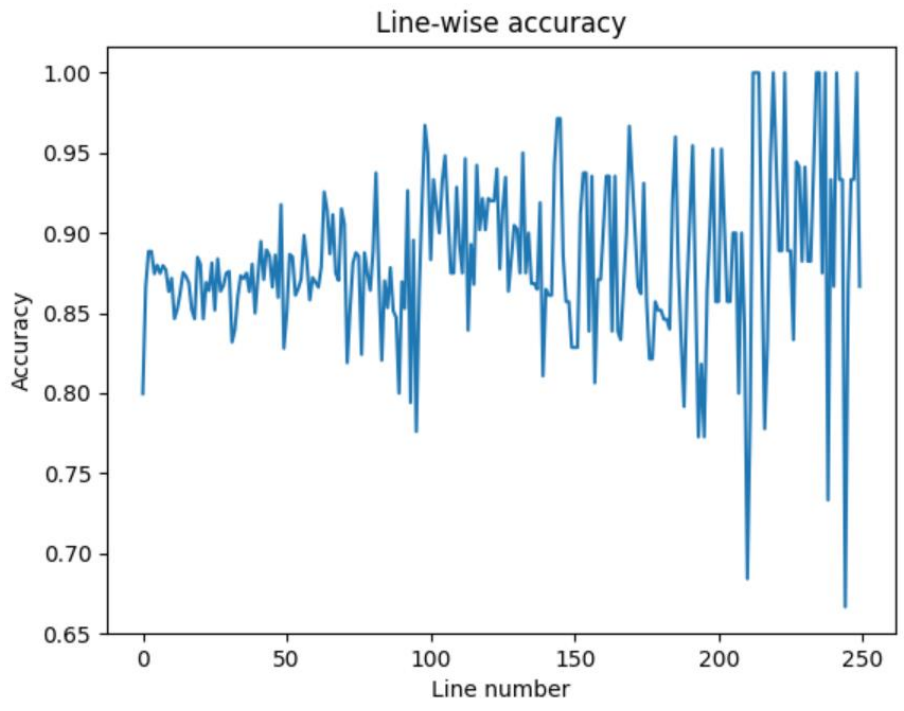


Figure 12: A graph of average line-wise test accuracy with the multilingual LSTM model

Similar to the corresponding graph of the French monolingual model, the variance in the average accuracies grows with the line numbers, and there is also a small drop with the first lines of the documents.

Classification-wise, there is no significant improvement from the earlier examples where the monolingual models fail the worst. In addition to this, there is poor performance with the Spanish and German test sets. This can partly be explained by the poor quality of the texts included: there are some texts in the Spanish and German test sets that are composed of generated text and / or non-text, as in text that is not natural language. The model naturally doesn't perform well with these kinds of texts. The model also doesn't recognize duplicate generated lines within a document as boilerplate, which makes it perform poorly with documents that contain lots of duplicate content such as the example in Table 19. In this example, every main content line is followed by an exact duplicate line that has been annotated as boilerplate, but the model has predicted all the lines as main content.

Table 19: An example of a text with duplicate lines

Predicted class	Confidence	Annotated class	Text
Main content	0.98526	Main content	Hola MarisolPink! Te escribo des USA para felicitarte por tus recetas pradisimas y faciles y a la vez preguntarte si tienes alguna receta para hacer tostadas rojas, tipo las de la marca la mision o algo por el estilo. Te agradezco de antemano por tu tiempo y tu ayuda. Atte. : Jesus R Mendoza
Main content	0.99219	Boilerplate	Hola MarisolPink! Te escribo des USA para felicitarte por tus recetas pradisimas y faciles y a la vez preguntarte si tienes alguna receta para hacer tostadas rojas, tipo las de la marca la mision o algo por el estilo. Te agradezco de antemano por tu tiempo y tu ayuda. Atte. : Jesus R Mendoza
Main content	0.99737	Main content	lo felicito son recetas muy fáciles y además baratas, ya que para estos tiempos la mayor parte de la gente no contamos con muchos recursos
Main content	0.99769	Boilerplate	lo felicito son recetas muy fáciles y además baratas, ya que para estos tiempos la mayor parte de la gente no contamos con muchos recursos
Main content	0.99507	Main content	HOLA SUSY: ME FASCINA TODAS LAS COSAS QUE HACES, SON MUY SENCILLAS Y FÁCILES DE HACER, ME GUSTA MUCHO TU CREATIVIDAD Y TU SENCILLES, DIOS TE BENDIGA, DESDE CÚCUTA COLOMBIA, ABRAZOS DESDE LA DISTANCIA
Main content	0.99453	Boilerplate	HOLA SUSY: ME FASCINA TODAS LAS COSAS QUE HACES, SON MUY SENCILLAS Y FÁCILES DE HACER, ME GUSTA MUCHO TU CREATIVIDAD Y TU SENCILLES, DIOS TE BENDIGA, DESDE CÚCUTA COLOMBIA, ABRAZOS DESDE LA DISTANCIA
Main content	0.99540	Main content	Hola soy nueva en tu pagina, y te quiero comentar que estan padrisimas las recetas que hasta ahorita eh visto, muy faciles y riquisimas.... felicidades. Sra. Olivia Rosas

Predicted class	Confidence	Annotated class	Text
Main content	0.99390	Boilerplate	Hola soy nueva en tu pagina, y te quiero comentar que estan padrisimas las recetas que hasta ahorita eh visto, muy faciles y riquisimas.... felicidades. Sra. Olivia Rosas
Main content	0.98825	Main content	CHEF VITORIA BRANDA Hace ya tiempo dió en el programa de Televisión la receta de unas galletas de clara de huevo con nuez, muy faciles, podría ser tan amable y mandarmela receta, me robaron mi recetario. Gracias de antemano. Elsa Ruy. Mexico, D. F.
Main content	0.96968	Boilerplate	CHEF VITORIA BRANDA Hace ya tiempo dió en el programa de Televisión la receta de unas galletas de clara de huevo con nuez, muy faciles, podría ser tan amable y mandarmela receta, me robaron mi recetario. Gracias de antemano. Elsa Ruy. Mexico, D. F.

Discussion

Based on the previous hypothesis of boilerplate and main content occurring in blocks, sequential labelling seemed like a good solution to boilerplate detection. The hypothesis holds true: the addition of sequential LSTM model improves the line-wise classification results reached by the XLM-R model. In nearly all cases the LSTM performs better than the XLM-R model, the only exception being zero-shot tested English, where the performance dropped when tested on the multilingual model that wasn't trained with English data.

Comparing the results to the previous research is somewhat complicated due to the different format of the data. Note that this data has already been run through Trafilatura Version 0.3 (Barbaresi, 2020) for boilerplate removal, and the remaining data still contains a number of boilerplate lines. When looking at the numbers, monolingual Swedish LSTM model outperforms the previous research and the monolingual French model and multilingual model aren't far behind. A more accurate comparison could be achieved by e.g. converting the CleanEval (Baroni et al., 2008) dataset into plain text form. In this case, it would be assumed that the blocks in the dataset annotated as main content don't contain any boilerplate and vice versa.

All the models fail with generated text and duplicate lines within a text. These are general tendencies that need improvement. The data set is quite small and all the variations of language probably won't be present. This is evident already in the register dispersion in the French and Swedish datasets: the whole data contains only 17 texts annotated as lyrical. It should also be noted that the evaluation is based on test sets that haven't been stratified by register: they may contain a different distribution of internet registers than the training and validation sets. In addition to this, there is no indication on how the models perform with unseen languages other than the zero-shot tests. These however don’t include any languages written with other than the Latin alphabet and the only language outside the Indo-European language family is Finnish, which does perform relatively well.

As with deep learning models in general, this model is not an exception in that it is not the lightest in computational efficiency. When choosing the method for boilerplate removal, one should consider if the possible performance gain is worth the additional computational cost (in comparison with e.g., Trafilatura (Barbaresi, 2021)), especially if the corresponding HTML documents are available.

7 Conclusion

The goal of this thesis was to find solutions to improving the quality of Internet-sourced data that is not in HTML format, as is the case with many existing Web corpora. Web-sourced material may contain substantial amounts of boilerplate: generated text, lists of links etc. that are not useful for further linguistic analysis, and it is a necessary step to clean the data of this. In addition, it is necessary to find methods to improve the job done by the existing boilerplate cleaning methods, that might leave some noise behind.

The data used in the training is a manually annotated sample of FreCORE and SweCORE sampled from Common Crawl and annotated for their register as well as smaller sets of random samples of English, Finnish, German and Spanish Common Crawls. The annotations were done per line: each line of text was annotated as either boilerplate or main content. In total, the French dataset consists of 84 103 lines, the Swedish dataset of 99 406 lines and the multilingual dataset of 200 430 lines. FreCORE and SweCORE also include annotations for register classes per text. It should be noted that the training data doesn't contain texts that are fully generated or machine translated.

For the boilerplate detection, I trained a sequential classifier to label main content and boilerplate lines in a plain text document. The training process was done in two steps: first, train a XLMR-RoBERTa model to label lines as main content and boilerplate and second, use the last hidden layer from this model to train a sequential labelling model with a Long-Short-Term-Memory (LSTM) network layer in order to take the surrounding lines into consideration. This process was done to train three different kinds of models: monolingual models for Swedish and French data, a multilingual model trained with Swedish, French, English, Finnish, German and Spanish data as well as experimental zero-shot multilingual models that were tested with an unseen language. These LSTM models are treated as the final product of the training.

Out of the two architectures examined in this thesis, the LSTM models outperform the models trained on top of the XLM-RoBERTa. The results of the evaluation of the LSTM models are close to the previous boilerplate detection applications. The monolingual Swedish model outperforms most of the existing models with mean accuracy of 0.92 (SD = 0.001), the monolingual French and multilingual models reach mean accuracies of 0.84 (SD = 0.005) and 0.88 (SD = 0.002). The highest evaluation score any previous application has got is

Trafilatura’s 0.914 accuracy (Barbaresi, 2021), but straight comparison is impossible due to the different nature of the data format. Trafilatura, as well as the other boilerplate removal applications discussed in this thesis, is trained and evaluated on DOM elements in HTML documents, while the data used in this thesis is line-wise annotated plain text.

The approach provided in this thesis offers an alternative or addition to HTML-based boilerplate removal: these models are able to differentiate between boilerplate and main content based on textual context only. This means that it is possible to process previously retrieved web corpora that do not have their HTML source documents or that have been treated with boilerplate removal applications that have for some reason or another not been successful enough. The multilingual tests provide hope that the multilingual model is somewhat generalizable, and that multilingual training for boilerplate removal is possible and perhaps even beneficial.

The results are however not perfect: the models have problems especially with texts with lots of generated text and registers that are not common, such as lyrical, and they don’t generally recognize duplicate lines within text. Increasing the amount of data could be a way to reach better results as this is quite a little dataset for deep learning. In addition to this, some smaller subsets (Spanish, German) were clearly somewhat faulty to begin with and didn’t yield good results in evaluation.

These models are evaluated only on French, Swedish, English, Finnish, Spanish and German, so there is no indication on how they will perform with languages outside the ones used in this study. This should be further examined with e.g., annotated datasets in other languages to evaluate the multilingual model’s generalizability. It would also be essential to further evaluate the model’s performance compared to others with datasets such as CleanEval (Baroni et al., 2008), this would however require some additional annotation. Another point to be examined could be the variance within the line-wise accuracies in the models’ predictions: it is not clear, why the average accuracy differs so much after a certain point.

References

Abadji, J., Suarez, P. O., Romary, L., & Sagot, B. (2022). *Towards a Cleaner Document-Oriented Multilingual Crawled Corpus*. http://arxiv.org/abs/2201.06642

Artstein, R. (2017). Inter-annotator Agreement. In N. Ide & J. Pustejovsky (Eds.), *Handbook of Linguistic Annotation* (pp. 297–313). Springer Netherlands. https://doi.org/10.1007/978-94-024-0881-2

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–15. http://arxiv.org/abs/1409.0473

Baker, P. (2006). Using corpora in discourse analysis. In *Continuum discourse series*. Continuum.

Baker, P. (2009). *Contemporary corpus linguistics*. Continuum. https://login.ezproxy.utu.fi/login?url=http://site.ebrary.com/lib/uniturku/Doc?id=10427369

Barbaresi, A. (2016). *Efficient construction of metadata-enhanced web corpora*. 7–16. https://doi.org/10.18653/v1/w16-2602

Barbaresi, A. (2019). The Vast and the Focused: On the need for thematic web and blog corpora. *Proceedings of the Workshop on Challenges in the Management of Large Corpora (CLMC-7) 2019*, 29–32.

Barbaresi, A. (2020). Generic web content extraction with open-source software. *Proceedings of the 15th Conference on Natural Language Processing, KONVENS 2019*, 267–268.

Barbaresi, A. (2021). Trafilatura: A Web Scraping Library and Command-Line Tool for Text Discovery and Extraction. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, 122–131. https://doi.org/10.18653/v1/2021.acl-demo.15

Barbaresi, A., & Lejeune, G. (2020). Out-of-the-Box and into the Ditch? Multilingual Evaluation of Generic Text Extraction Tools. *Proceedings of the 12th Web as Corpus Workshop*, *May*, 5–13. https://aclanthology.org/2020.wac-1.2

Baroni, M., Chantree, F., Kilgarriff, A., & Sharoff, S. (2008). CleanEval: A competition for cleaning web pages. *Proceedings of the 6th International Conference on Language Resources and Evaluation, LREC 2008*, 638–643.

Bawden, R., Di Nunzio, G. M., Grozea, C., Jauregi Unanue, I., Jimeno Yepes, A., Mah, N., Martinez, D., Névéol, A., Neves, M., Oronoz, M., Perez-de-Viñaspre, O., Piccardi, M., Roller, R., Siu, A., Thomas, P., Vezzani, F., Vicente Navarro, M., Wiemann, D., & Yeganova, L. (2020). Findings of the WMT 2020 Biomedical Translation Shared Task: Basque, Italian and Russian as New Additional Languages. In *Proceedings of the Fifth Conference on Machine Translation* (pp. 660–687). Association for Computational Linguistics. https://www.aclweb.org/anthology/2020.wmt-1.76

Biber, D., & Conrad, S. (2019). *Register, genre, and style*. Cambridge University Press.

Biber, D., Conrad, S., & Reppen, R. (1998). Corpus linguistics : investigating language structure and use. In *Cambridge approaches to linguistics*. Cambridge University Press.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2019). Unsupervised Cross-lingual Representation Learning at Scale. *CoRR*, 31–38. https://doi.org/10.18653/v1/p19-4007

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*(Mlm), 4171–4186. http://arxiv.org/abs/1810.04805

Egbert, J., Biber, D., & Davies, M. (2015). Developing a bottom-up, user-based method of web register classification. *Journal of the Association for Information Science and Technology*, *66*(9), 1817–1831. https://doi.org/10.1002/asi.23308

Gatto, M. (2014). *The web as corpus : theory and practice*. Bloomsbury.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Huang, Z., Xu, W., & Yu, K. (2015). *Bidirectional LSTM-CRF Models for Sequence Tagging*. http://arxiv.org/abs/1508.01991

Jantunen, J. H. (2018). Homot ja heterot Suomi24:ssä: analyysi digitaalisista diskursseista. *Puhe Ja Kieli*, *38*(1), 3–22. https://journal.fi/pk/article/view/65488/32762

Johansson, M., Kyröläinen, A.-J., Ginter, F., Lehti, L., Laippala, V., & Krizsán, A. (2018). Opening up #jesuisCharlie anatomy of a Twitter discussion with mixed methods. *Journal of Pragmatics*, *129*, 90–101. https://doi.org/10.1016/j.pragma.2018.03.007

Jurafsky, D., & Martin, J. H. (2009). Speech and language processing : an introduction to natural language processing, computational linguistics and speech recognition. In *Prentice Hall series in artificial intelligence* (2nd ed). Prentice Hall.

Kanerva, J., Ginter, F., Miekka, N., Leino, A., & Salakoski, T. (2018). Turku Neural Parser

Pipeline: An End-to-End System for the CoNLL 2018 Shared Task. *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, 133–142. https://doi.org/10.18653/v1/K18-2013

Karpathy, A. (2015). *The Unreasonable Effectiveness of Recurrent Neural Networks*. Andrej Karpathy Blog. https://karpathy.github.io/2015/05/21/rnn-effectiveness/

Kilgarriff, A. (2007). Last Words: Googleology is Bad Science. *Computational Linguistics, Volume 33, Number 1, March 2007*. http://aclweb.org/anthology/J07-1010

Kim, Y. (2014). Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1746–1751.

Laippala, V., Kyllönen, R., Egbert, J., Biber, D., & Pyysalo, S. (2019). Toward Multilingual Identification of Online Registers. *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, 292–297. https://www.aclweb.org/anthology/W19-6130

Laippala, V., Rönnqvist, S., Hellström, S., Luotolahti, J., Repo, L., Salmela, A., Skantsi, V., & Pyysalo, S. (2020). From Web Crawl to Clean Register-Annotated Corpora. *Proceedings of the 12th Web as Corpus Workshop*, 14–22.

Lample, G., & Conneau, A. (2019). Cross-lingual Language Model Pretraining. *Advances in Neural Information Processing Systems*, *32*. http://arxiv.org/abs/1901.07291

Le, H., Vial, L., Frej, J., Segonne, V., Coavoux, M., Lecouteux, B., Allauzen, A., Crabbé, B., Besacier, L., & Schwab, D. (2019). FlauBERT: Unsupervised Language Model Pre-training for French. *LREC 2020 - 12th International Conference on Language Resources and Evaluation, Conference Proceedings*, 2479–2490. http://arxiv.org/abs/1912.05372

Lehti, L. (2013). Genre et ethos : des voies discursives de la construction d’une image de l’auteur dans les blogs de politiciens. In *Thèse de doctorat*. Université de Turku. http://urn.fi/URN:ISBN:978-951-29-5556-5

Leonhardt, J., Anand, A., & Khosla, M. (2020). Boilerplate Removal using a Neural Sequence Labeling Model. *The Web Conference 2020 - Companion of the World Wide Web Conference, WWW 2020*, 226–229. https://doi.org/10.1145/3366424.3383547

Luoma, J., Oinonen, M., Pyykönen, M., Laippala, V., & Pyysalo, S. (2020). A broad-coverage corpus for Finnish named entity recognition. *Proceedings of the 12th Language Resources and Evaluation Conference*, *May*, 4615–4624. https://aclanthology.org/2020.lrec-1.567

Malmsten, M., Börjeson, L., & Haffenden, C. (2020). *Playing with Words at the National Library of Sweden -- Making a Swedish BERT*. http://arxiv.org/abs/2007.01658

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133. https://doi.org/10.1007/BF02478259

Olah, C. (2015). *Understanding LSTM Networks*. Colah's Blog. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Partington, A. (2013). Patterns and meanings in discourse : theory and practice in corpus-assisted discourse studies (CADS). In *Studies in Corpus Linguistics*. John Benjamins Publishing Company.

Pomikálek, J. (2011). Removing boilerplate and duplicate content from web corpora. *PhD En Informatique, Fakulta Informatiky*.

Repo, L., Skantsi, V., Rönnqvist, S., Hellström, S., Oinonen, M., Salmela, A., Biber, D., Egbert, J., Pyysalo, S., & Laippala, V. (2021). Beyond the English Web: Zero-Shot Cross-Lingual and Lightweight Monolingual Classification of Registers. *EACL 2021 - 16th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Student Research Workshop*, 183–191. http://arxiv.org/abs/2102.07396

Rönnqvist, S., Skantsi, V., Oinonen, M., & Laippala, V. (2021). Multilingual and Zero-Shot is Closing in on Monolingual Web Register Classification. *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, 157–165. https://aclanthology.org/2021.nodalida-main.16

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain [Article]. *Psychological Review*, *65*(6), 386–408. https://doi.org/10.1037/h0042519

Salminen, J., Hopf, M., Chowdhury, S., Almerekhi, H., & Jansen, B. (2020). Developing an online hate classifier for multiple social media platforms. *Human-Centric Computing and Information Sciences*, *10*(1), 1–34. https://doi.org/10.1186/s13673-019-0205-6

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, *45*(11), 2673–2681.

Stemle, E. (2010). *The KrdWrd CANOLA Corpus – Gathering Training Data for Sweeping Web Pages*. 1–17.

Sutskever, I., Martens, J., & Hinton, G. (2011). Generating text with recurrent neural networks. *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, 1017–1024.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural

Networks. *Lausanne, EPFL*, *2366*. https://doi.org/10.5075/epfl-thesis-2366

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems*, *2017-Decem*(Nips), 5999–6009. http://arxiv.org/abs/1706.03762

Virtanen, A., Kanerva, J., Ilo, R., Luoma, J., Luotolahti, J., Salakoski, T., Ginter, F., & Pyysalo, S. (2019). *Multilingual is not enough: BERT for Finnish*. http://arxiv.org/abs/1912.07076

Vogels, T., Ganea, O.-E., & Eickhoff, C. (2018). *Web2Text: Deep Structured Boilerplate Removal*. http://arxiv.org/abs/1801.02607

Wang, C., Li, M., & Smola, A. J. (2019). Language Models with Transformers. *2019 16th International Computer Conference on Wavelet Active Media Technology and Information Processing, ICCWAMTIP 2019*, 249–253. https://doi.org/10.1109/ICCWAMTIP47768.2019.9067534

Wenzek, G., Lachaux, M.-A., Conneau, A., Chaudhary, V., Guzmán, F., Joulin, A., & Grave, E. (2019). CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data. *LREC 2020 - 12th International Conference on Language Resources and Evaluation, Conference Proceedings*, 4003–4012. http://arxiv.org/abs/1911.00359

Appendices

Appendix 1 Line-wise Annotation Guidelines

Main content:

- Body of text(s)
- titles
- Info about the author or the site/organization (incl. Contact information)
- Comments
- Discussion forums, comments: info about the author (posted by... etc); if this kind of information is lacking, any text (e.g., signatures, ranking, rating [4.0 stars out of 5.0] etc.) that can be used to separate the posts from each other is marked as ok
- News site front pages or similar: excerpts from the articles + their titles
- Lists that are part of the text
- In discussion forums, the most important parts to retain are WHO + WHAT (incl. Previously published messages that are being replied to) +WHEN (NB. this means that repetition is ok in discussion forums)
- If a line ends with e.g., ‘Leave a comment’, ‘Continue reading’, ‘See full summary’ these are kept and the entire line is marked ok (i.e., no cutting of lines)
- Links within a text and that are part of a text are ok (e.g., “Join our community today!” with the first three words constituting a hyperlink and the entire phrase being a part of the entire text)
- Wikipedia: ‘[edit]’ etc.
- Wikipedia: all the parts that feature in the list of contents and layout-wise are part of the body of the text are ok, meaning that e.g., External links and See also are ok
- Poster and date of posting in e.g., Community Blogs
- Junky-looking lines with non-junky information (such as date and/or author) that might be informative for the understanding of the text

- Repetitive text that is part of the body of text, or (almost) the same text appears twice, on the actual website
- Table of Contents ok if they describe the main text of the page (and the main text is included)
- Captions within a text (e.g., in a news text)
- Wikipedia change logs → both versions ok
- Short news text listings (title + headnote) are ok if they consist of 2 or more sentences

Boilerplate:

- Lists of links (old posts, related, web store product listings, discussion forum topics etc)
- Generated "buttons" or meta information about the site
- Texts about cookies, the site platform (e.g., "this site was built with WordPress"), JavaScript, copyrights etc
- (unrelated/generated) Ads
- Requests to comment, share, follow, download, etc
- Headers if titles in text
- if the main title for the text appears in the body of text and in the header, the header is considered as junk since it usually includes other meta information as well. However, if the only title for the text appears in the header, it is saved.
- Repetitive text (if the same "ok" line appears more than once)
- The first line is annotated ok, the second (repetitive) one as junk
- Hotel texts (incl. Lists, terms and conditions)
- Short excerpts of other texts

- if the main body of text is complete: e.g., news article + excerpts from related articles -> related excerpts are junk. If there isn’t a visible main body of text, the excerpts are treated as ok, including their titles.
- If there are several different (seemingly) complete texts, they are all ok
- Discussion forums: information about the poster (admin, ‘ranking’ according to number of posts etc.)
- Hyperlinks in the end of a Wikipedia text
- Pronunciation sites
- Wikipedia: side banners (e.g., boxes with information about a species)
- Labels/keywords/tags
- Word conjugations in Wiktionary
- Table of Contents of a book when the text itself not included
- A text consisting of captions only (e.g., stock photos)
- Captions that are illogically placed in a text (e.g., after a text as a chunk)
- # + ‘tag’ (e.g., #politics)
- Incoherent text
- In (product) reviews, e.g., Amazon, “the most useful positive/negative comment” is junk, if the same post appears later in the text, i.e., it appears twice
- Short news article listings (title + headnote) if they’re under 2 sentences per text

Appendix 2 Suomenkielinen lyhennelmä

Automatisoidut hakuohjelmat kuten Common Crawl (Wenzek et al., 2019) keräävät vapaan internetin sivustoja valtaviksi aineistoiksi, joita voidaan hyödyntää kielitieteellisessä ja -teknologisessa tutkimuksessa. Ongelmana tämänkaltaisissa aineistoissa on se, että ne sisältävät usein generoitua tekstiä ja metatietoja, jotka saattavat vääristää lingvistisen analyysin tuloksia (Laippala et al., 2020). Tärkeä askel verkkoaineistojen keruussa onkin niiden siistiminen, mutta sitä dokumentoidaan usein harmillisen vähän (Kilgarriff, 2007). Olemassa olevat sovellukset eivät aina ole riittäviä, ja jo kertaalleen putsattu teksti saattaa sisältää ylimääräistä generoitua tekstiä (Laippala et al., 2020). Tämän opinnäytetyön tavoitteena on kouluttaa koneoppimismalleja tunnistamaan tämän kaltaista tekstuaalista ”hälyä” eli vakiotekstiä (engl. boilerplate) leipätekstin lomasta sekä vertailla mallien suorituskykyä keskenään ja aiempien toteutusten kanssa. Lisäksi aion tarkastella tekstivariaation ja monikielisyyden vaikutusta mallien suorituskykyyn.

Teoreettinen viitekehys

Vapaa internet tarjoaa valtavan määrän tekstiaineistoja, joita on mahdollista käyttää kielitieteellisessä tutkimuksessa: aktiivisia verkkosivuja on yhteensä 1,9 miljardia tammikuussa 2021. Tällaisia aineistoja voidaan hyödyntää esimerkiksi korpuslingvistisessä tutkimuksessa, kuten diskurssianalyysissä (Partington, 2013), mutta niiden avulla voidaan myös analysoida jonkin tietyn ryhmän kielen käyttöä ja piirteitä (Biber et al., 1998). Perinteisesti kielitieteellinen korpus edustaa jotain tiettyä osa-alaa kielestä, kuten tekstilajia, aikaikkunaa tai käyttötapaa (Baker, 2009). Koko Internetin edustaminen yhdessä korpuksessa eroaa tästä monin tavoin. Internetin tekstiaineistot ovat erittäin vaihtelevia sekä laadultaan että sisällöltään (Barbaresi, 2021), eikä koneellisesti haettujen korpusten sisältöä usein tunneta ennen niiden annotointia (Barbaresi, 2019; Laippala et al., 2020). Lisäksi koneellisesti haettu aineisto saattaa sisältää duplikaatteja, konekäännöksiä ja metatietoa kuten linkkilistoja, sivustojen muotoilua sekä hakupalkkeja. Tällaisen ”ylimääräisen” tekstin suodattaminen verkkoaineistoista on välttämätöntä, mikäli halutaan laadukkaita tekstiaineistoja (Barbaresi, 2021; Kilgarriff, 2007).

Vakiotekstin (eng. boilerplate) poistaminen on olennainen osa verkkoaineistojen luomisprosessia (Barbaresi, 2021). Karkeasti verkkoaineistojen sisältö voidaan jakaa kahteen kategoriaan: leipätekstiin[18] ja vakiotekstiin (Pomikálek, 2011; Vogels et al., 2018). Vakiotekstiksi määritellään yleensä ”ei-informatiiviset osat verkkosivun leipätekstin ulkopuolella”[19] (Pomikálek, 2011), ja siihen lukeutuu esimerkiksi navigaatiopalkit, linkkilistat ja mainokset. Kahtiajako vakiotekstin ja leipätekstin välillä ei kuitenkaan ole aina yksinkertaista: esimerkiksi julkaisuun yhteyteen liitetty tiivistelmä voidaan näkökulmasta riippuen tulkita joko vakiotekstiksi tai leipätekstiksi. Vaikka yleisesti hyväksyttyä rajausta vakiotekstin ja leipätekstin välille ei ole, suuntaviivaa antavat erinäiset annotointiohjeistukset kuten CleanEval (Baroni et al., 2008) sekä KrdWrd (Stemle, 2010).

Useimmat olemassaolevat vakiotekstinpoisto-ohjelmat hyödyntävät verkkosivujen HTML-koodia, jonka avulla voidaan eritellä verkkosivujen eri osaset myös silloin, kun eri sivujen HTML-käytänteet eroavat toisistaan (Barbaresi, 2021). Esimerkiksi verkkosivun leipäteksti on usein jaettu kappaleisiin (paragraph), joita merkitään notaatiolla <p> … </p>. On kuitenkin otettava huomioon, että koska HTML-käytänteet vaihtelevat verkkosivustojen välillä suuresti, on hankalaa kehittää yleistettävissä olevaa mallia (Barbaresi, 2021).

On olemassa sekä sääntöpohjaisia (Barbaresi, 2021; Pomikálek, 2011) että koneoppimiseen pohjautuvia (Leonhardt et al., 2020; Vogels et al., 2018) vakiotektinpoisto-ohjelmia. Esimerkkinä sääntöpohjaisesta ohjelmasta, jusText (Pomikálek, 2011) jakaa ensin HTML-tiedoston sen koodissa esiintyviin pätkiin, ja luokittelee ne joko hyväksi, huonoksi, lyhyeksi tai melkein hyväksi tekstiksi[20] pätkän sisällön ja pituuden perusteella. Näiden luokittelujen perusteella teksti jaetaan vakiotekstiin ja leipätekstiin sillä periaatteella, että leipätekstiä (”hyvä”) rajaa ulommaiset ”hyvä” tai ”melkein hyvä” -lohkot, ja sen ulkopuolelle jäävä teksti luokitellaan vakiotekstiksi. Pomikálek huomauttaakin, että useimmiten vakioteksti ja leipäteksti esiintyvät keskenään lohkoissa. Tätä huomiota hyödyntää myös LSTM-verkkoihin perustuva BoilerNet (Leonhardt et al., 2020).

Neuroverkkoja on hyödynnetty laajalti luonnollisen kielen käsittelyssä esimerkiksi konekääntämisessä (Bahdanau et al., 2014; Bawden et al., 2020), tekstin koneellisessa tuottamisessa (Karpathy, 2015; Sutskever et al., 2011) ja erilaisissa luokittelutehtävissä (Kim, 2014; Laippala et al., 2019; Repo et al., 2021; Rönnqvist et al., 2021; Salminen et al., 2020). Neuroverkot pyrkivät jäljittelemään ihmisaivojen toimintaa laskennallisin menetelmin

[18] Engl. main content, oma käännös
[19] “non-informative parts outside of the main content of a Web page”, oma käännös
[20] Good, bad, short, near-good

(McCulloch & Pitts, 1943). Yksinkertainen sovellutus neuroverkosta on Rosenblattin perseptroni (Rosenblatt, 1958), joka koostuu yhdestä neuronista, joka laskee neuronille syötetyille muuttujille painotetun summan ja määrittää tuloksen (engl. output) raja-arvofunktion perusteella. Tulos voi olla esimerkiksi ennustettu luokka. Tämänkaltaisia perseptroneja voidaan yhdistää monikerroksiseksi perseptroniverkoksi (engl. multilayer perceptron).

Edellämainitut perseptronit ovat eteenpäinkytkettyjä neuroverkkoja (engl. feed-forward neural network), jotka eivät ota kielen kontekstia huomioon. Luonnollinen kieli kuitenkin noudattaa järjestystä muun muassa kirjainten, sanojen ja lauseiden suhteen, ja asioiden merkitys on usein riippuvainen kontekstista: esimerkiksi sanajärjestyksellä voidaan muuttaa virkkeen merkitys täysin. Kielen sisäinen järjestys tulee ottaa huomioon, kun kehitellään malleja esimerkiksi tekstin koneelliseen tuottamiseen, konekääntämiseen, nimettyjen entiteettien tunnistamiseen tai sanaluokkien jäsentämiseen. Tätä varten kehitettiin ajatus takaisinkytketyistä neuroverkoista (engl. recurrent neural network), sovellutus eteenpäinkytketystä neuroverkosta, jossa tietoa kulkee askeleelta toiselle (Olah, 2015). Pitkäkestoinen lyhytkestomuisti -verkot (engl. long short-term memory networks, LSTM) (Hochreiter & Schmidhuber, 1997) ovat edelleen sovellutus takaisinkytketyistä neuroverkoista, joissa on muistiyksikkö tiedon kuljettamiseen pitempien sekvenssien yli. Eduistaan huolimatta takaisinkytketyt neuroverkot ovat usein kalliita kouluttaa, jonka vuoksi niitä voi olla hankalaa skaalata suurempiin projekteihin. Transformer-mallien arkkitehtuuri toimii ilman takaisinkytkentää ja on täten laskennallisesti tehokkaampi (Vaswani et al., 2017).

Aineisto ja metodologia

Tämän opinnäytteen aineisto pohjautuu FreCORE ja SweCORE -aineistoihin (Repo et al., 2021), jotka ovat rekisteri (genre) -annotoitu otanta Common Crawlista. Aineisto on deduplikoitu Onionin avulla sekä siitä on jo kertaalleen poistettu vakiotekstiä Trafilaturalla (versio 0.3) (Barbaresi, 2020). Aineiston rekisteriannotointi seuraa englanninkielisen CORE:n rekisteriluokittelua, jossa tekstit jaetaan kahdeksaan ylärekisteriin: kerronnallinen (Narrative), informatiivinen (Informative), mielipiteellinen (Opinion), ohjeistava (How-to), puhuttu (Spoken), informatiivinen vaikuttaminen (Informational Persuasion), lyyrinen (Lyrical) sekä vuorovaikutteinen (Interactive Discussion). Nämä yläkategoriat jakautuvat vielä useisiin alarekistereihin: esimerkiksi Uutiset kuuluvat kerronnalliseen ylärekisteriin, tietosanakirjaartikkelit informatiiviseen kategoriaan. Aineisto ei sisällä tekstejä, jotka ovat selkeästi konegeneroituja tai -käännettyjä tai jotka sisältävät pelkästään vakiotekstiä.

Koska aiemmista vakiotekstinpoistoyrityksistä huolimatta aineistoon oli jäänyt ylimääräistä generoitua tekstiä ja muuta metatietoa, joten se päädyttiin annotoimaan vielä riveittäin vakiotekstiin ja leipätekstiin. Rivillä tarkoitetaan tässä tapauksessa rivinvaihtoihin rajautuvaa tekstinpätkää, kansantajuisesti voitaisiin siis useimmiten puhua kappaleesta.

Yhteensä aineistossa on 1 982 ranskankielistä ja 2 399 ruotsinkielistä tekstiä, jonka lisäksi aineistoon on otettu mukaan pienet erät englannin-, suomen-, saksan- ja espanjankielistä riviannotoitua dataa monikielisiä kokeiluja varten. Kaiken kaikkiaan aineisto kattaa 200 430 riviä tekstiä, joista 57,7 % on annotoitu leipätekstiksi ja 42,3 % vakiotekstiksi. Riviannotoinneille on ranskan- ja ruotsinkielisten osuuksien suhteen laskettu kahden annotoijan välisten annotointien yhdenmukaisuus tarkkuutena (ranska 0,86; ruotsi 0,80) sekä Krippendorffin alphana (ranska 0,65; ruotsi 0,55). Nämä luvut viittaavat siihen, että tekstin luokittelu vakiotekstiin ja leipätekstiin on haastavaa myös ihmiselle.

Rakennettavan koneoppimismallin tavoitteena on antaa luokiteltavan tekstin rivien luokille leimat siten, että ympäröivä konteksti (ympäröivät rivit) otetaan huomioon. Tätä lähestymistapaa tukevat aiemmat tutkimukset (Leonhardt et al., 2020; Pomikálek, 2011), jossa vakiotekstin ja leipätekstin todettiin esiintyvän useamman kappaleen lohkoissa. Mallin rakentamisessa hyödynnetään transfer-oppimista kouluttamalla jo olemassa oleva XLM-RoBERTa- kielimallia riviannotoidulla datalla. Koulutus on kaksivaiheinen: ensin koulutetaan malli, jossa jokaiselle riville ennustetaan luokkaleimat. Tämän mallin viimeistä ”piilokerrosta” (engl. hidden layer) hyödynnetään seuraavassa askeleessa syöttämällä se inputina mallille, johon on lisätty kaksisuuntainen LSTM-kerros. Tämä malli antaa riveille lopulliset leimat.

Tulokset

Ruotsinkielinen LSTM-malli toimii 0,92 tarkkuudella, ranskankielinen 0,84 tarkkuudella ja monikielinen malli 0,88 tarkkuudella. LSTM-mallin suoritus parantui pohjalla olevan XLM-R -mallin suorituksesta: ruotsinkielisen mallin tarkkuus oli 0,84; ranskankielisen 0,82 ja monikielisen 0,84. Tämä tulos on tilastollisesti merkitsevä ($p < 0{,}05$). Lisäksi mallien suoristusta arvioitiin zero-shot-testeillä, joissa monikielinen malli koulutettiin aineistolla, joka ei sisältänyt dataa testiaineiston kielellä. Näissä mallin suoritus oli enimmäkseen huonompi kuin monikielisellä mallilla, mutta esimerkiksi ranskan testitulos oli verrattavissa yksikielisen mallin suoritukseen. Espanjan- ja saksankielisten aineistojen testitulokset taas olivat korkeammat, mitä ne olivat monikielisen mallin kanssa. Tämä saattaa viitata siihen, että näissä aineistoissa on joitain systemaattisia ongelmia.

Helpottaakseni vertailua aiempiin tuloksiin laskin malleille myös F1-scoret, jotka ovat 0,84 ranskankieliselle mallille, 0,92 ruotsinkieliselle mallille ja 0,88 monikieliselle mallille. Verrattuna aikaisempiin tuloksiin, kouluttamani mallit suoriutuvat samankaltaisesti muiden vastaavien sovellusten kanssa, ruotsinkielinen malli jopa edeltäjiään paremmin. Vertailu on kuitenkin hankalaa, sillä toisin kuin käyttämäni aineisto, useimmat olemassa olevat toteutukset hyödyntävät vakiotekstin poistossa HTML-koodia. Työssäni käsittelemistäni sovelluksista Trafilatura ohittaa sekä ranskankielisen että monikielisen LSTM-mallin suorituksessaan F-scorella 0,912, mutta jää hieman jälkeen ruotsinkielisen mallin suorituksesta.

Koska kielellinen variaatio saattaa vaikuttaa riviluokitteluihin, LSTM-mallien suorituskyky arvioitiin myös rekisterikohtaisesti. On otettava huomioon, että koska rekisterien hajonta aineistossa ei ole tasaista, voi mallien suorituskyky eri rekisterien välillä vaihdella suurestikin. Esimerkiksi lyyrisiä tekstejä on koko aineistossa vain kourallinen, joten on luonnollista, ettei malli kykene ennustamaan riviluokkia niille yhtä hyvin kuin esimerkiksi uutisteksteille, joita on aineistossa satoja. Yleisesti ottaen ruotsinkielinen malli ennustaa luokkia paremmin kuin ranskankielinen malli kaikkien paitsi lyyrisen rekisterin suhteen. Parhaiten se suoriutuu Mielipide, Puhuttu ja Informatiivinen vaikuttaminen -kategorioihin lukeutuvien tekstien suhteen. Ranskankielisen mallin parhaiten onnistuneet rekisterit ovat Narratiivinen, Mielipide sekä Informatiivinen vaikuttaminen.

Ranskankielinen malli tunnistaa leipätekstin paremmin kuin vakiotekstin (F1-scoret 0,86 ja 0,72). Tämä saattaa johtua siitä, että ranskankielisestä aineistosta vain noin kolmannes riveistä on annotoitu vakiotekstiksi. Tämä voi myös selittää sen, miksi ranskankielisen testiaineiston evaluointi monikielisellä mallilla oli yksikielistä mallia korkeampi. Mallilla on vaikeuksia luokitella rivejä sellaisista teksteistä, joissa sitä on paljon suhteutettuna leipätekstiin. Esimerkiksi keskustelufoorumiteksteissä malli usein ennustaa enemmän rivejä vakiotekstiksi, mitä on annotoitu. Lyhyissä mainosteksteissä taas malli saattaa ennustaa leipätekstiksi sellaisetkin rivit, jotka ovat tosiasiassa linkkejä tai muuten generoitua tekstiä.

Ruotsinkielinen malli ennustaa vakiotekstiä paremmin kuin leipätekstiä (F-scoret 0,93 ja 0,91). Testiaineisto on tasaisemmin jakaantunut näiden kahden luokan välille kuin ranskankielinen aineisto ja vastaa hajonnaltaan koko ruotsinkielistä aineistoa. Yleisesti ottaen malli toimii hyvin. Kuten aiemmin kuitenkin jo todettiin, sillä on haasteita Lyyristen tekstien kanssa, mikä näkyy erittäin selkeästi testiaineiston esimerkkejä tarkastellessa: malli ennustaa lähes kaikki tekstin lyyriseen osuuteen kuuluvat rivit vakiotekstiksi. Myös muunlainen normista poikkeava kieli on mallille hankalaa. Esimerkiksi blogipostauksen yhteydessä olevat kommentit, jotka sisältävät epäformaalia kieltä ennustetaan vakiotekstiksi.

Monikielinen malli tunnistaa leipätekstin paremmin kuin vakiotekstin (F1-scoret 0,91 ja 0,86). Ranskankielisen mallin tavoin sillä on hankaluuksia paljon generoitua tekstiä sisältävien tekstien kanssa. Kuten aiemmin todettiin, malli ei suoriutunut kovin hyvin espanjan- ja saksankielisten testiaineistojen kanssa. Tämä saattaa osin selittyä sillä, että kyseiset aineistot sisältävät tekstejä, joissa on pelkästään generoitua tekstiä. Lisäksi monikielisessä testiaineistossa oli jonkin verran tekstejä, jotka sisälsivät tekstinsisäisiä duplikaattirivejä, jotka malli luokitteli säännönmukaisesti leipätekstiksi.

Malleilla oli siis ylipäänsä vaikeuksia generoidun tekstin, norminvastaisen kielen ja tekstinsisäisten duplikaattien kanssa. Tässä on muutama selkeä kehityskohde tulevaisuuden tutkimuksia varten. Käyttämäni aineisto on verrattain pieni, ja monikielisistä testeistä huolimatta on vaikeaa sanoa, kuinka hyvin mallit ovat yleistettävissä kielille, jotka eivät ole indoeurooppalaisia tai jotka eivät käytä latinalaisia aakkosia.

Yhteenveto

Tässä opinnäytetyössä koulutin ranskankielisen, ruotsinkielisen sekä monikielisen koneoppimismallin luokittelemaan tekstitiedoston rivejä leipätekstiksi ja vakiotekstiksi. Kaikki kolme mallia suoriutuvat tehtävästään suhteellisen hyvin, ruotsinkielinen malli jopa ylittää suorituskyvyssään aiemmat sovellukset. XLM-R mallin tulosten pohjalta rakennettu LSTM-malli paransi luokittelutuloksia: tämä vastaa hypoteesia siitä, että järjestystä noudattava neuroverkko soveltuisi vakiotekstin tunnistamiseen, sillä vakioteksti ja leipäteksti usein esiintyvät omissa lohkoissaan.

Mallit eivät kuitenkaan toimi virheettä, joten tulevaisuudessa olisi hyvä kiinnittää huomiota aineiston laajentamiseen siten, että malleja voitaisiin kehittää mahdollisia kompastuskohtia silmällä pitäen. Malleilla on hankaluuksia norminvastaisen ja generoidun kielen kanssa, joten tällaista aineistoa lisäämällä voitaisiin saada parempia tuloksia. Lisäksi olisi hyvä kerätä aineistoa myös tässä opinnäytetyössä käytettyjen kielten ulkopuolelta mallien yleistettävyyden varmistamiseksi.