Adrian Wærøe Langseth

Use of Spatial Information in News Recommenders

Master's thesis in Computer Science
Supervisor: Heri Ramampiaro
June 2022

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Kunnskap for en bedre verden

Adrian Wærøe Langseth

Use of Spatial Information in News Recommenders

Master's thesis in Computer Science
Supervisor: Heri Ramampiaro
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU

Kunnskap for en bedre verden

Adrian Wærøe Langseth

Use of Spatial Information in News Recommenders

Master Thesis, Spring 2022

Artificial Intelligence Group
Department of Computer and Information Science
Faculty of Information Technology, Mathematics and Electrical Engineering

Abstract

The transition from printed newspapers to online news portals has made a huge amount of news articles instantly available to users. This necessitates news recommender systems to guide users through the ocean of news articles. News recommender systems have become more personalized to achieve higher performance by using more data which can be perceived as intrusive to the user's privacy.

In this thesis, we examine the use of the user's position in the modelling of user profiles. Furthermore, we investigate how to mitigate the usage of their positions by modelling the user profile to include their interest in specific locations based on the news articles alone. The work is performed with the hypothesis that profiling the user's location interests can mitigate the usage of user position without significant performance loss and therefore increase users' privacy.

In this thesis, we have implemented and extended a state of the art news recommender with a spatial encoder. The spatial encoder generates a spatial profile of the user, either by location interests or by user position. We test the extended model and show that the model does not depend on user position. On the contrary, our experimental results show that the user position significantly lowers the performance of the state of the art news recommender. Overall, we conclude that using user position as part of user profile modelling has a privacy cost without any performance benefit.

Sammendrag

Overgangen fra papiraviser til nyhetsportaler på nett har gjort en enorm mengde nyheter tilgjengelig for brukerne til enhver tid. Dette nødvendiggjør gode nyhetsanbefalingssystemer som kan geleige brukeren gjennom havet av nyhetartikler. I et forsøk på å gi bedre anbefalinger har anbefalingssystemene blitt mer og mer personaliserte. Dette medfører bruk av data som kan oppfattes som påtrengende på brukerens personvern.

I denne masteroppgaven undersøker vi brukerens posisjon sin rolle i modellering av brukerprofiler for nyhetsanbefalingssystem. Dermed undersøker vi hvordan man kan unngå bruken av brukerens posisjon ved å anvende tekstuelt innhold i modellering av brukerprofilen til å inkludere brukerens lokasjonsinteresser.

Vi implementerte en *state-of-the-art* nyhetsanbefalingsmodell, og utvidet den med en område-enkoder. Område enkoderen genererer en område-profil for brukeren basert på dens nyhetslokasjonsinteresser eller dens posisjon. Vitester den utvidede modellen og viser med signifikans at bruken av brukerens posisjon påvirker den utvidede modellens ytelse negativt.

Våre resultater viser at bruker posisjonen ikke øker kvaliteten av anbefalingene til den implementerte modellen. Vi konkluderer dermed at bruken av brukerens posisjon påfører en kostnad for personvernet uten en økning i ytelsen av anbefalingssystemet.

Preface and Acknowledgements

I would like to thank my supervisor Heri Ramampiaro for his guidance and uplifting spirit throughout the period. I also thank Juan Carlos Lopez Calvet and Schibsted for sharing their enthusiasm and excitement for technology. Furthermore, I thank Jon Atle Gulla and Lemei Zhang for access to the Adressa Dataset, and the HPC lab for access to IDUN. I would also like to thank my family for their continued support. Finally, I thank all the friends who have made these five years a fun and exciting adventure.

Adrian Wærøe Langseth

Trondheim, June 13, 2022

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we will introduce the thesis. In Section 1.1 we will discuss the background and motivation of the thesis. Section 1.2 will outline the main goal of our thesis and the research questions which need to be answered to achieve the goal. Section 1.3 presents the contributions made to the field of study by the findings in this thesis. Finally, in Section 1.4, we will outline the structure of the thesis.

1.1 Background & Motivation

Online news has become a staple of the internet and has surpassed traditional newspapers as the way people consume news [34]. The ever-growing database of articles has resulted in a need for recommender systems that are able to present relevant articles to each user. Over the years, the recommenders have evolved from systems that present the newest or most popular articles to everyone, to systems that try to model the interests of every individual user and specialize the news feed to suit the individual's preferences. These systems collect more and more specialized information, beyond the user behaviour, to model the user's preferences with the aim at providing ever-more accurate and personalized recommendations. However, there has recently been an increased focus on user privacy. In 2016, the EU introduced General Data Protection Regulation (GDPR) which aimed to strengthen privacy by regulating the storage of user data. This was followed by the launch of Apple's App Tracking Transparency (ATT) in 2021, which required companies to get explicit permission from the users to track them across applications. This focus on privacy, combined with the need for high performance in news recommendation, are somewhat opposing forces showing a need for privacy gain without substantial performance loss. There is already much research and many

applications of this type of technology. For example, Figure 1.1 shows an ad for Kobler, a company specializing in ad placement for companies that do not wish to use personal information. The company states that they reach a larger audience by connecting ads to the context of their placement (e.g., an ad for a gym in news articles relating to fitness). Furthermore, the company states that the audience of the ads thinks more favorably about the advertiser when the advertiser relies on context rather than user data. The market size for such privacy-preserving, yet performance-enhancing system is aligned with its importance in our constant shift into a more and more digital society.

Figure 1.1: An ad for a company specializing in ad placements based on contextual factors without the use of personal data. The ad reads "If you sell insurance, you do not need personal data to reach your target audience.". The ad was taken from the news portal Dagbladet.no

The motivation of this thesis is based on several factors. Due to the online shift in news and the inherent scalability of the internet, the newspaper and media companies have seen an extensive consolidation in the past decades, to the point where only three large news media companies remain in Norway [25]. This consolidation of newspapers naturally leads to data sharing within the company and a consolidated tracking of users across all news portals in the company. Therefore, the area of news recommenders is interesting as it has a large impact on companies to increase their focus on user privacy.

News recommenders have gained infamy over the last couple of years on the basis of echo chambers and filter bubbles [47]. This was particularly evident in the years surrounding the 2016 United States presidential election, as the polarization in both the population and the media grew rapidly. Although this thesis does not concern filter bubbles in particular, the possibility of developing news recommenders towards more ethical grounds is enticing.

News recommenders have a particular benefit over other recommenders in reducing the amount of user data collection. Almost all item information is inherently available in the item itself. The body of a news article contains all the information of the article. Therefore, the road to a recommender using less user data appears shorter in news recommendation than in other recommender system application areas.

One of the most intrusive features used in news recommendation today is the constant tracking of the user's position. Some usages of the position may not be a serious privacy concern, such as the STPM model [43] which does not use it to model the user, but rather to distinguish between your search history at home and at work. However, on the other side of the spectrum is *Mobifeed* [66], which tracks the user's position to estimate a trajectory and speed of the user's movement, to queue news relating to the position it predicts you to arrive at in the future. This large range of use patterns motivates the riddance of it altogether. Furthermore, the usage is often motivated by recommending news to the user based on the assumption that spatially close news are also relevant news. One may, however, hypothesize that the user divulges a spatial interest through other and less obtrusive means, e.g., by examining previous news items the user has been interested in. Therefore, we are motivated to investigate the degree to which a news recommender is able to provide personalization without tracking the user's position.

Note that throughout this thesis, we will consistently use the term "user position" to refer to the geographical *position* of the user, as in where the user is geographically, whereas the term "news location" will be used to refer to the geographical *location* of the events referred to in the news article, which may include several locations per article. Although position and location are synonyms, the distinction in this thesis is made for clarity. However, this does not include the terms "Location-Aware" and "Location-Based", as these refer to the user position, since they are important research topics within recommender systems.

1.2 Problem specification

In this chapter, we present the main goal of this thesis and the research questions we aim to answer.

Goal *Investigate to which degree one can create a news recommender system which uses textual content to build a user profile containing enough information on the user's location interests to mitigate the need to track the user's position without a large loss in performance.*

The goal of this thesis is to investigate to what extent the user's news browsing history and the user's interest in news regarding specific locations can alleviate the need for collecting the user position. The hypothesis of this thesis is that the information on the user's position is mostly subsumed by the location interests, and that given a high performance news recommender that incorporates the location of news, the user position will not provide much additional lift to the performance. Although a general assumption in recommenders using locations is that a user prefers to read articles which concern their location [69, 54, 6, 66], it is reasonable to assume that a user would rather read news articles regarding locations the users has previously been interested in, than an article about a location nearby. This hypothesis is further based on the multifaceted nature of location interests. Users are often interested in several locations, such as where they are from, where they live, and where their family lives. These three locations of interest can be reflected in a location-interests oriented approach, but cannot be fully reflected by a user-location oriented approach. With this in mind, this thesis tries to answer the following main research question:

RQ *How can a news recommender use the inferred location interests of users to mitigate the need of using users' position?*

This main research question can be divided into the following more specific sub-questions:

RQ 1: *What is the current state of the art in location-aware news recommendation?*

RQ 2: *How can user position and news location be modelled for news recommendation?*

RQ 3: *Is the user's position a necessary component for achieving high performance news recommendation?*

Answering the research questions will enable the achievement of the goal of the thesis. RQ1 will examine the state of the art and provide knowledge for further research into the field. Answering RQ2 will provide the knowledge for implementing or extending a method for assessing the effectiveness of news recommendation based on user position data and the user's location interests. RQ3 will be answered by the implementation of a news recommender that uses the user's position and location interests in recommendation, and assessing the validity of the hypothesis through an ablation study.

1.3 Contributions

In this thesis we provide several contributions to the field.

To answer RQ1, we present a comprehensive review of the state of the art in news recommenders, specifically within personalized new recommenders, location-based news recommenders and location-aware recommenders. The thesis further contributes with a spatial extension of the NAML model, which enables the use of spatial data in the recommendation. Furthermore, the extension is independent of the underlying model and can be applied to any other news recommender that predicts a click probability. This contribution answers RQ2 by showing how spatial information can be modelled for news recommendation. Finally, to answer the last research question, RQ3, this work demonstrates that it is possible to develop a personalized news recommender system without violating the user's privacy, especially with regard to collecting their position information. To show this, we performed extensive experiments and deep analysis of the results, including ablation studies and comparative analysis with a statistical significance test.

Further contributions that do not directly relate to the research questions include a preprocessing approach for the Adressa Dataset to prepare it for research like ours. Datasets such as this one are generally messy. Thus, making the dataset suitable for the development of news recommendation is in itself a contribution. Furthermore, several custom TensorFlow layers are designed and built for this thesis which have general value. *Masked Mean Pooling* layer is proposed, which provides a general pooling layer with the option of masking inputs. An *Attention* Layer used in the original implementation of NAML [60] is ported to TensorFlow 2.0, which lowers the implementation difficulty of several state of the art neural news recommender methods which use this module, such as NRMS [62], NAML [60] and LSTUR [5]. The *Slicer* layer is a custom layer that slices out a defined section of a Tensor. The layer is a substitute for implementing the logic manually in Lambda layers, which reduces the implementational difficulty of several models.

The full extent of the contributions to the field provided by this thesis will be presented and detailed in Section 6.2 and Section 6.3.

1.4 Thesis Structure

The remainder of the thesis is structured as follows:

Chapter 2: Background Theory will introduce the core concepts and the necessary background information to understand the methodology and contributions of the thesis.

Chapter 3: Related Work will introduce and examine the state of the art in news recommenders with a focus on news recommender employing the users' position. This chapter will address RQ1.

Chapter 4: Technical Approach describes the design and implementation of the method we will use to answer RQ2 and RQ3.

Chapter 5: Experimental Evaluation describes the experiments used to evaluate the method, and presents the results.

Chapter 6: Discussion discusses the solution, the validity of the results, and the relevant future work.

Chapter 2

Background Theory

In this chapter we give the required background for the work performed in this thesis. The background description will cover recommender systems, with a special focus on news recommenders. We will also delve into some details regarding machine learning techniques that are particularly relevant in this area. The elements covered in this chapter suffice for appreciating the subsequent chapters, including the discussion of the state of the art in news recommendation that is discussed in Chapter 3, but in order to make the background as concise as possible, we do not discuss the broader topics in machine learning that are not directly related to the developments in this thesis. Rather, we refer the interested reader to Jordan and Mitchell [30], Russell and Norvig [51], Goodfellow et al. [20], Aggarwal [3], and Raza and Ding [47]. Furthermore, as the discussion here is meant to cover the important topics at a high level, instead of detailed results, we will give references to materials in which more information about certain results or definitions can be found instead of citing the specific works where a definition or result was initially published. This will help the interested reader to complete the picture, if required.

2.1 Recommender Systems

The purpose of a recommendation system is to retrieve and recommend items to users that the user will find relevant. This is formalized by Borges and Lorena [10] as given a set of users U, a set of items I, let s be a utility function which evaluates the item in regards to the user, so that

$$s : U \times I \to V. \tag{2.1}$$

where V is the completely ordered set of evaluations. The recommender system should then recommend the item i' which maximizes the utility function s for the user $u \in U$:

$$i' = \arg\max_{i \in I} s(u, i) \tag{2.2}$$

The central problem of recommender systems is that the function s is not observed over the entire $U \times I$ space, and therefore must be extrapolated from the observed user-item ratings. The utility can be defined by an arbitrary function which can be dependent on the application, the users, and the items [2]. The extrapolation of the utility function is typically based on specified heuristics or by approximating the function by optimizing some performance metric. An example of a heuristic-based extrapolation is recommending the most popular items based on the heuristic that the most popular items by prior users are most likely to be popular by the current user. The extrapolation of the utility function may be obtained by using heuristic formulas such as similarity measures, or Machine Learning (ML) methods such as Support Vector Machines (SVM), Neural Networks (NN) and clustering [10, 21, 2]. Recommender systems are typically divided into three main paradigms: content-based filtering, collaborative filtering and hybrid recommender systems. In the next sections, these paradigms will be presented and discussed.

2.1.1 Content-Based Filtering

Content-based filtering methods use information about the items to recommend items to users which are *similar* to the items which the user has preferred in the past. The similarity is a key component of the methods and is defined differently between methods and applications, but typically assesses the similarity between the features of each item by some mathematical operation. Similarity functions typically operate in a feature space, where the items are represented by a vector in a space with the features as axes. For a movie recommender this vector space could perceivable have axes such as "Comedy" and "Thriller". Then a film will be represented in this space as a vector indicating the degree to which the movie is comedic or thrilling. Similar to the movie, the recommender attempts to also view the user as a vector in the feature space as well, based on the items previously liked. The recommender will then recommend the movies which are *closest* in the feature space to the user, which is determined by the similarity of the vectors. The values for each item-feature must be set at some point, either manually by critics or users, or by some automated process that analyzes reviews or even the movie itself.

The same case is applicable to news recommender systems. A news article would be represented as a vector in some feature space with axes based on the generated features of the article. Although the movie may have structured data from which to generate features, the news recommenders have most of the information about a news article within the article itself. Therefore, content based filtering is very popular in news recommender, where Karimi et al. [31] showed that 59 of 112 analyzed papers used content based filtering, while only 19 used collaborative filtering, with the rest using a hybrid approach. However, text is unstructured data and requires features to be generated before it can be mapped to a feature space. This is usually done by embedders which map words to representations over some abstract undefined features, but keeps the meaning of the word embedded. This will be further discussed in Section 2.3.3. To compute some relevance score between the user and the item, the recommender uses a similarity function sim to compute a relevance score between the user and item representations,

$$rel_{u,i} = sim(u, i) \tag{2.3}$$

2.1.2 Collaborative Filtering (CF)

Collaborative filtering is a recommender approach based on the utilization of the ratings of all users. The systems draw inferences from the matrix of ratings to predict the ratings of a user on an item. If there are two users who are similar in ratings of other items, and one of these users particularly likes or dislikes an item that the other has not rated, the recommender system will predict that their ratings of that item will continue to be similar.

In a large database of users and items, most users will not have rated most items. This leads to a sparse rating-matrix, which is one of the challenges of CF systems. It is difficult for the CF systems to compute the neighborhood when the amount of items that users have in common might be small, which can lead to lower accuracy in predicted ratings.

In contrast to content based filtering, collaborative filtering systems do not consider the content data of the items. Therefore, this is an approach used in cases where there is little item information, but many users. Movie recommenders is a typical application of CF methods, as it contains little information about the contents of the movie, and users watch many movies.

CF systems are typically divided into two main approaches: memory-based and model-based systems.

Memory-based collaborative filtering systems use the entire rating-matrix to find the subset of users who are most similar to the target user. This is often called

the "neighborhood" of the target user. The neighborhood's ratings of the item in question are aggregated to predict the rating of the target user. This method can also be used on the basis of items, where the system finds the neighborhood of the most similar item to the target item. The user's ratings of the neighborhood items are aggregated to predict the rating of the target user.

Memory based methods are simple and explainable, but when the user-item rating-matrix is sparse, it may be challenging to find similar users which have rated certain items. Furthermore, the memory based methods need to keep the entire rating-matrix in memory, which can be computationally expensive.

Model-based collaborative filtering systems attempt to create models of the data which can be used for predicting the ratings, without the need for iterating over the rating-matrix. These systems employ machine learning to tackle the recommendation task as a matrix completion task, where the matrix is the user-item rating-matrix.

2.1.3 Hybrid

Hybrid recommender systems are recommenders that unify content-based filtering and collaborative filtering into a combined approach. This is typically done to mitigate the weaknesses of one approach by including the predictions from another. The approach is also taken when several inputs are available, such as if both the user-item rating-matrix and the content data are informative, then both a collaborative approach and a content-based approach can be taken.

2.1.4 Feedback

Recommender systems are inherently dependent on feedback from the users to determine the relevancy of items, and therefore to learn which items to recommend to which users. Feedback can come in many forms, but are typically categorized into two categories: *explicit feedback* and *implicit feedback* [29]. Table 2.1 shows some examples of explicit and implicit feedback.

Explicit

Explicit feedback is the feedback provided intentionally by the user when they give their subjective opinion on an item [11]. In the field of recommender systems, explicit feedback typically consists of a user rating an item or leaving a review [11]. As the feedback is intentionally given and a reflection of their own opinion, it is considered highly accurate and reliable [29, 4]. Due to its accurate reflection of

the user's preferences, explicit feedback is the most convenient type of feedback for the recommender system [58]. Explicit feedback is often harder to attain, as users are reluctant to provide explicit feedback on items, most likely because of the cognitive effort it requires [29]. Furthermore, the recommender systems are reluctant to inquire the user on their opinion as it may disrupt the experience of the user. An example of this is that a news recommender system would likely not ask the user to rate every article they read, as this would disrupt the reading experience of the user.

Implicit

Implicit feedback is feedback implicitly derived by the system based on user activity [27]. This feedback is typically not intentionally given, and the user may not be aware it gives it, but the system can track it in abundance without the risk of interrupting the user.

Hu et al. [27] lists 4 prime characteristics of implicit feedback. The (1) first characteristic is that there exists no negative implicit feedback. The system can only infer the relevance of an item, but not the non-relevance of an item. The case of news recommender systems is a clear example, as the click on a news article is typically interpreted by the news recommender system as a sign that the news item is relevant to the user. However, low feedback can be used as a negative indicator in certain domains where the implicit feedback is abundant and repeated [46]. TV shows are an example of this, as users who only watch the pilot, but not any more, could be assumed to find the show non-relevant. The (2) second characteristic of implicit feedback is its inherent noise. The interaction with an item, without explicit information on why the interaction happens, is not necessarily based on the item's relevance [27]. A user may not like an item they purchased, but the purchase is the only feedback collected, and therefore introduces noise. The (3) third characteristic describes that while explicit feedback gives the preference of the user, implicit feedback gives an indication of the confidence the system has in the user's preference. This is due to the fact that explicit feedback is freely given by the user, and we can have total confidence in the user's reported preference. However, implicit feedback is only indicated by their actions, which means that the system must assess its *confidence* in the preference of the user. This is demonstrated by the following example: a user purchasing an item once may be an indicator of the user's preference of the item, but the confidence is low as this is not repeated. However, if the user purchases an item every week, then the confidence in the user's preference of the item is high. This is connected to the first two points. The confidence must be used as the user cannot indicate non relevance, and the data is inherently noisy, so the system cannot be fully confident

Explicit	Implicit
Like/dislike	Time Spent
Ratings	Purchases
Review	Clicks
Like/dislike	Bookmarking/Saving

Table 2.1: Examples of implicit and explicit feedback.

in the user's preference unless the first two characteristics are addressed. The (4) final characteristic is that the use of implicit feedback requires special evaluation metrics. The models have no simple way of accounting for the circumstances of the feedback, such as the availability of the item or repeated feedback. If the item is never available to the user, or the user interacts with the item several times, traditional measures are incompatible.

In news recommendation, the feedback is generally implicit and comes in the form of either clicks on news articles or the time spent reading an article [21, 65]. In news recommendation, we can not make the assumption that low feedback can be interpreted as negative feedback [46], as very few users revisit articles after the first read. However, in state of the art news recommendation, a common assumption is that an article that is shown to the user, but not clicked, is considered given negative feedback. This will be further discussed in Section 3.5. News recommendation is, as described by Hu et al. [27], inherently noisy as a user may click the article and still find the article non-relevant. This may be due to the way news is presented to the user, by title and thumbnail. A user may view the article title as interesting and then click the item and find the article was not as interesting as first thought. This may be exacerbated by the rise in *clickbait* news reporting, where the title tries to inflate the importance of the article but leave out as many details as possible, to persuade the user to click the article. This introduces noise as the click itself does not indicate the relevance of the article, but rather the degree to which the article can persuade the user to click it. Additionally, many other noise sources exist in the news recommendation space. Nevertheless, the assumption that a click on an article is an indicator of positive preference is common in news recommendation. As the maximum normal amount of clicks on an article is 1, the confidence in such a preference is usually assumed to the 100%. This is due to that there is largely no implicit feedback to fit the confidence on a scale from 0 to 100%, so the assumption is made that we have equal confidence in all preferences. Finally, as we assume equal confidence in the preferences, we can utilize standard evaluation metrics of recommender systems.

2.2 News Recommender Systems

The definition of the recommender system shown in Section 2.1 can be adapted to news recommender systems as follows: given a set of users U, a set of news articles A, let s be a utility function which evaluates the item in regards to the user, so that

$$s : U \times A \to V \tag{2.4}$$

where V is the completely ordered set of evaluations. The recommender system should then recommend the news article a' which maximizes the utility function s for the user $u \in U$:

$$a' = \arg\max_{a \in A} s(u, a) \tag{2.5}$$

2.2.1 Characteristics of News Recommendation

News recommendation can be characterized as more difficult than regular recommendation tasks due to several domain characteristics. The following section gives an overview of the most important characteristics which define the news recommendation domain.

On most news portals, users are able to remain pseudo-anonymous while browsing news articles by not logging in, which disables cross-session tracking. This results in a lack of long-term behavioural data of the users as they are considered new users for each new session. This is a repeating cold start problem where the system never stores enough data on the user to produce an efficient model.

News recommenders have recently come under scrutiny for their involvement in filter bubbles. Recalling the approaches discussed in Section 2.1, the collaborative methods attempt to show the user the preferred items of other users who are similar in preferences, and the content based methods attempt to show the user items which are similar to the created user profile. The commonality is that the system is incentivized to recommend more items of the kind previously consumed. In the domain of news recommendation, these goals may result in showing articles that reinforce the beliefs and biases held by the user. Therefore, an additional goal of the news recommendation system is to have diversity in its recommendations, in order to broaden the news landscape of the user.

The number of views of news articles on a news site shows long-tail behaviour, meaning that the bulk of interactions is concentrated among a few very popular

articles, while the rest of the articles receive much fewer views. A news recommender could therefore take into account the popularity of an article, to either counter the bias to learn to recommend articles regardless of popularity, or lean into the popularity and use it to recommend popular articles in the belief that popular articles are relevant to everyone.

News articles have a very short life span from the first view to the last. This is due to the relevance of a news story may be very short, due to contextual factors. Therefore, a large portion of the articles available to recommend are not relevant, regardless of user preference. Furthermore, the continued input of new articles makes the cold-start-item problem especially pronounced for news recommenders.

The majority of the data in a news article is in the body and title. The text is unstructured and cannot be used directly without some form of feature extraction. Therefore, a key part of any news recommender is the processing of the textual data.

2.2.2 Personalization of news recommender systems

A non-personalized news recommender system is a recommender that does not attempt to use the user's preference in their recommendation of a news item. Examples of these are Top-10 recommendations or a tab showing the most popular news. Although these systems exist and are effective in certain use cases, little research is made on their development. This thesis will concern personalized recommender systems, and when we address recommender systems, it is implied that they are personalized.

2.2.3 Architecture of a news recommender system

Figure 2.1 shows a typical architecture of a personalized news recommendation model [64, 63]. The system consists of news profiling and user profiling. News profiling is used to take in the candidate news, which is the news that is to be considered for recommendation, and encode it to a news profile. The news profile will be a structured representation of the contents and context of the news article. The user encoder takes in the user context, which is the information on the user and the situation. This is usually any available demographic information on the user, such as age, gender, location or income, as well as session information such as the current time.

The items in the user history are profiled in the same manner as the candidate news, and combines with the user context in the user encoder to form the user profile. The user history is often used to indicate an interest in the respective

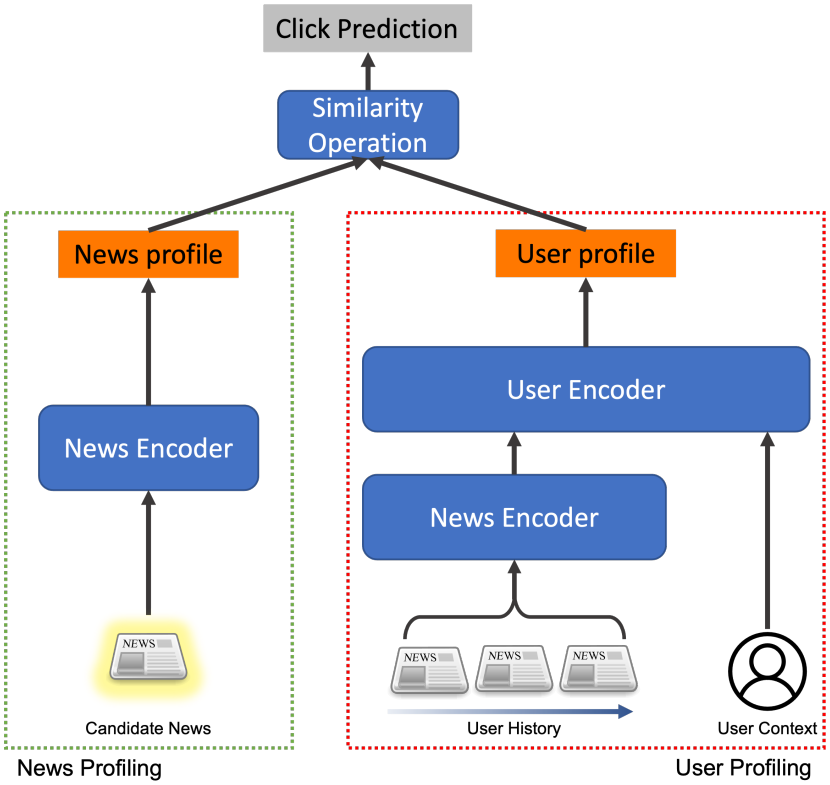


Figure 2.1: Typical personalized news recommender architecture

articles.

The user profile is compared with the candidate news profile by a similarity measure to get a click prediction, which is a number indicating the likelihood that a user will find the article interesting or willing to click on the article. The broader usage is in a larger context of several candidate news, with the task becoming a ranking problem.

2.2.4 News Profiling

News profiling techniques and models aim to convert rich news articles into an encoding of their contents. This encoding should accurately represent the contents of the news article and its topics. The encoding is referred to as a news profile. News profiling techniques are mainly feature-based or deep learning-based [63].

Feature-based profiling typically requires manually generated features to represent the articles, such as keywords, categories or entities [63]. In addition to these, several methods incorporate various contextual factors into the news profiling,

such as article popularity and recency, to help the system avoid old news and promote popular articles. Further features commonly used in news profiling are shown in Table 2.2.

Manual feature engineering is accurate, but requires a large amount of domain knowledge and manual tagging. Deep learning methods can learn to automatically extract features from the news articles as a part of the recommendation model. Because of this, deep learning methods have become very popular in news profilers to extract features from the large amounts of data that is stored in the unstructured text of the article. The deep learning methods employed for this task are typically Natural Language Processing (NLP) methods which generate representations of words and texts. These are called word embedders and will be briefly discussed in Section 2.3.3.

A hybrid combination of deep learning and features has become prevalent in the state of the art due to the availability of human-labeled features combined with the article text in large news recommendation datasets.

2.2.5 User Profiling

User profiling is the process of generating a representation which is representative of the user's preferences. The representation is typically based on the representations of the news the user has previously read. As shown in Figure 2.1, the user history is first encoded to the news profile, and then used in the user encoder to generate the user profile. This structure means that the user profile is based on the news profiles, and the user profile will be a function of its user history. Therefore, if a candidate news is similar to the previously read articles, the news profile of the candidate will be similar to the user profile, giving it a high click prediction.

2.2.6 Click Prediction

When the user and news profiles are generated, the click prediction module estimates the relevance of the candidate news to the user, and based on this recommends a suitable article to the user. Formally, the utility function R of a candidate news c for a user u, is given by some similarity function:

$$R(u, i) = \text{sim}(\vec{\mathbf{r}}_u, \vec{\mathbf{r}}_c) \tag{2.6}$$

Where $\vec{\mathbf{r}}_u$ and $\vec{\mathbf{r}}_c$ are the user profile and candidate news profile, respectively.

Prediction of relevance based on a traditional similarity function, such as cosine similarity, between the news profile and the user profile is widely employed in the

Content Features	
Semantic	Topic Model
Entity	Keyword
Emotion	Multimodal

(a) Extracted from news content

Property Features	
Category	Cluster
Location	Publisher
Publish Time	

(b) Instrinsic or static property

Context Features	
Popularity	CTR
Recency	Novelty
Dwell Time	Bias

(c) Dynamic property

CF Features
News ID
User ID
User/News Graph

(d) Collaborative Filtering Signal

Table 2.2: Overview of news features, adapted from Wu et al. [63]

state of the art, despite its simplicity. Deep learning has also been employed to implement the similarity function, however the state of the are overwhelmingly uses inner products (dot product and cosine similarity).

2.3 Machine Learning for Recommender Systems

Machine Learning (ML) is a broad branch of artificial intelligence consisting of several categories, each spanning several methods and applications. Mitchell [42] provided the following definition of learning:

> "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

This was later simplified by Jordan and Mitchell [30], who described machine learning as the discipline which "[...] addresses the question of how to build computers that improve automatically through experience".

This section will serve as a brief introduction to the main approaches and themes of machine learning and the machine learning methods relevant to the application of news recommendation.

2.3.1 Machine Learning Approaches

Machine Learning approaches are traditionally categorized into three main categories: *supervised learning*, *unsupervised learning* and *reinforcement learning*. The three categories will be briefly introduced and described, before focusing on the aspect relevant to the thesis.

Supervised Learning is the branch of ML which learns a function which maps the input to an output based on give input-output pairs [51]. We select a method and train the function f on the given labelled data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ by minimizing some loss function L. Formally, we wish to find a function $\hat{f}$ in some predefined set of functions $\mathcal{F}$, which minimizes the loss L between the predicted target $\hat{f}(\mathbf{x}_i)$ and the target y_i. This is shown in equation Equation 2.3.1.

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} L(f(\mathbf{x}_i), y_i) \tag{2.7}$$

Unsupervised Learning is the branch of ML which is based on the lack of targets in the dataset. The branch spans a multitude of approaches, whose tasks are varied. A common approach in unsupervised learning is clustering, where given a set of inputs $\{x_1, x_2, ..., x_n\}$, the objective is to group the inputs into a number of clusters with the aim of maximizing the similarity within each cluster and dissimilarity between clusters. Despite the lack of targets, the model can learn the clusters of similar users, which can further be used in a large amount of applications such as recommender systems. In recommender systems, unsupervised learning is used in memory-based collaborative filtering, where the rating prediction of a user is given by the aggregation of the ratings of a group of the most similar users to the user in question.

Reinforcement Learning (RL) is a very different approach from the other two branches. Reinforcement learning is based on the model interacting with the environment it is placed in and receiving rewards and punishment from the actions of the model. Reinforcement learning in news recommendation is focused on the sequential nature of the task, where a user asks for recommendations and selects and article from the recommendations or rejects all. This repeats until the session is over. This form of news recommendation problem could be suitable to a reinforcement learning system as it can continuously adapt in accordance to the rewards, and that it does not need to continuously store the user ratings, but rather update its internal model. However, to the extent of our research, few state of the art RL models exist in the domain of news recommendation.

2.3.2 Machine Learning Task

In supervised news recommendation there are two main tasks which the model can be assessed by. Classification is a popular and simple task for recommendation. In news recommendation it calculates the loss of the model on the basis of the distance between the predicted click probability and the binary target indicating whether the user clicked the article. However, the model is often assessed on its ability to rank the candidates. This makes the predicted scores only matter relative to the other candidates, as the order of the candidates is the way in which the model is assessed. Typically, the classification task is applied when the candidates are presented to the model individually, for the model to calculate a relevance score. Ranking tasks are more often used in Top-n news recommendation, where a large amount of candidates are presented to the model simultaneously, but only the top-n most relevant are to be presented to the user. Several more tasks exist, however these are not commonly applied to news recommender systems.

2.3.3 Neural Methods

In this section we will present and describe the neural methods employed in the state of the art presented in Chapter 3, and in the technical approach presented in Chapter 4.

Recurrent Neural Networks (RNN)

The recurrent neural network is a type of Artificial Neural Network (ANN) which supports processing sequential data by keeping an internal state and using the outputs from the last time step to calculate the output of the next time step. This allows the information from the earlier inputs to flow through the hidden state and propagate through the time steps to the later parts of the sequence. This allows the ANN to keep a "memory" of what has happened, and let it affect the later events.

Figure 2.2 shows a RNN. On the left is the basic RNN, while on the right is the RNN unfolded across the timesteps. On the left the input is a sequence, while on the right, the sequence is distributed over the inputs. A RNN can take a arbitrary amount of inputs, as it can unfold however long is required.

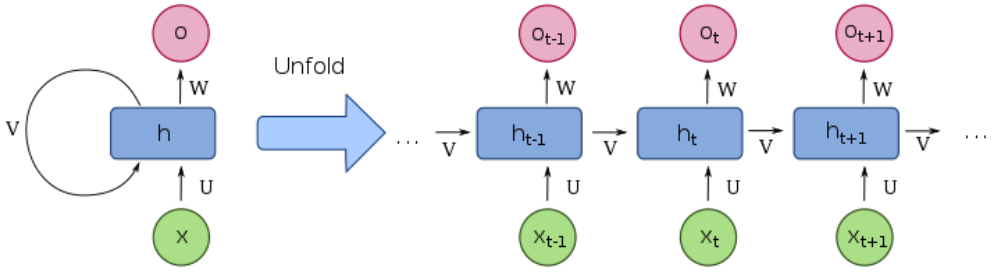


Figure 2.2: A basic RNN in its folded and unfolded position [28]

RNN's have several applications, mostly within tasks which are dependent on the specific order of the inputs. In news recommendation, this could be used capturing the order of the user history, or the order of read news articles in a session.

Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN) is a type of ANN which uses convolutional layers and pooling layers to extract features from the inputs and classify based on the extracted features. Due to its ability to extract features over successive layers,

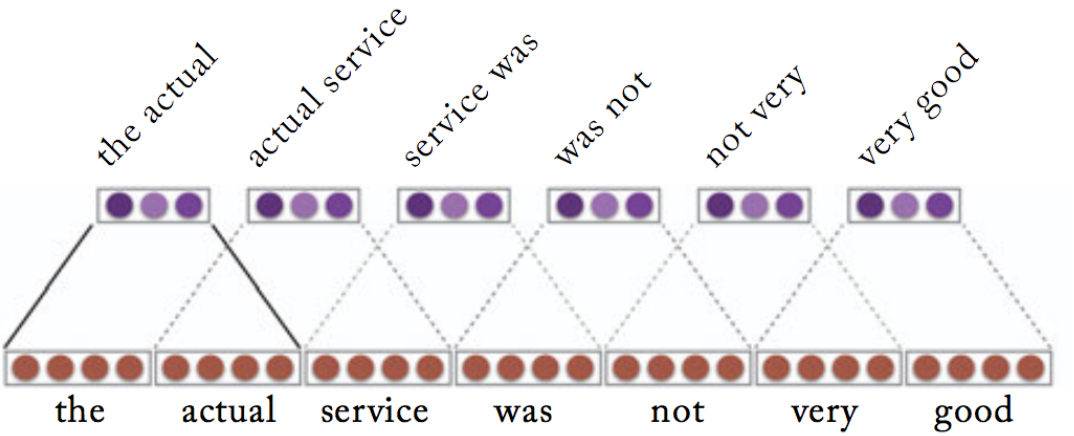


Figure 2.3: An example of the function of a 1D convolution over a sentence [19]
.

means that the combination of the layer and pooling combination successively in a network can extract higher-level features. Figure 2.3 shows the main function of the convolutional network in a text embedding task. The inputs are words which are separately embedded. The convolution of size 2, combines two and two embeddings together, using the convolution operation over their embeddings to receive a representation for the combined two words. As we see in the last convolution, the words "very" and "good" have been combined to "very good". This is a demonstration of how context is added by convolutions over text. If we were to use a convolution of size 3, we would have "not very good" as the last convolution. This is a single layer of convolution, and a convolutional neural network is typically several such layers which extract larger features than the combination of 2 and 3 words. This utilized in news recommender systems to find context and features in text, to represent the textual information in the news articles.

Attention

The attention mechanism in artificial neural networks is a technique which attempts to mimic the attention in humans. When humans read a sentence, different attention is payed to each word. Furthermore, the attention given to a word is related to the attention given to all other words. This cognitive process is reproduced in the attention mechanism. The attention mechanism takes in a sequence of inputs and calculates the amount of attention it should afford each input. The attention layer proceeds to calculate the weighted sum of the inputs. The calculation of attention varies with each type of attention layer. Further details and

specifics will be described in Chapter 3 and subsection 4.3.1.

Embedder

An embedder is a NLP technique which represents words as real-value vectors in a vector space, and are representative of the meaning of the words. The embedder generates similar vectors for words which appear closely together in the input texts, as to retain the meaning of the words. Furthermore, if the meanings are retained, one can do mathematical operations on the vectors to view the validity of the embeddings. Given that $\vec{V}_{word}$ is the vector for a word, then embedding typically respects rules like

$$\vec{V}_{prince} - \vec{V}_{man} \approx \vec{V}_{princess} - \vec{V}_{woman}. \tag{2.8}$$

The equation states that the difference between the embeddings of the word "prince" and "man", should be roughly equal to the difference between the embeddings of "princess" and "woman". This should hold to show that embeddings are representational of the words which it embeds.

Word embedding is a very important technique in news recommendation, as it respresents the textual words in the news articles by a real-valued vector. This enables the models to use the words in a dense representation that carries semantivc meaning.

2.3.4 Topic Modelling

The field of topic modelling is based on the idea that a document can be described as a mixture of several topics, where each topic is a probability distribution over specific words [9]. A topic model is *generative*, meaning that it describes how to generate new documents by specifying a distribution over topics, and then sample words from the topics based on both the probability distribution over words for each topic and over the given probability distribution of topics [53]. This, in itself, is not very applicable for encoding of a news article. However, by using statistical techniques, the process can be reversed, so that for each document,the topic distribution of the document can be generated, and thereby the document can be represented by a set of learned topics. This is the core of topic modelling, where through statistical methods, documents can be represented by a distribution over a set of topics [50]. Typically, the topics are individually interpretable by examining the words allocated to it, which carries the benefit of having a representation based on interpretable topics. This can be compared to traditional vector space

models, where the representation is given by a vector over unspecified and arbitrary axes [50].

Chapter 3

Related Work

The goals of this thesis are not methodology focused, but rather methodology-enabled, meaning we do not aim to provide a new state of the art method, but rather aim to provide research that requires the use of such methods as a vessel. Therefore, related work and methods will be examined and detailed in terms of their usability for the research. The related work will be examined in order of progressive specialization towards a user-position and news-location aware recommender. The state of the art in personalized recommenders is presented and examined first, then the location-based recommenders, and finally the location-aware recommenders. When these specializations have been established, a conclusion will be made on whether an appropriate method is found. Furthermore, the datasets available for the project are subsequently described before a decision is made on which dataset to use in the experiments.

3.1 Personalized News Recommenders

The state of the art in personalized news recommenders has shifted over the last decade from traditional recommender methods such as content-based, collaborative, and hybrid methods, to incorporating deep learning techniques [72]. Several of these methods of the traditional approaches heavily rely on manual feature engineering to build representations of users and news articles, which requires massive domain knowledge [5]. Furthermore, the lifetime of an item in the news recommender space is extremely short, where 84.5% of articles in the MIND dataset have less than two days between the first and last interaction containing the article [65]. The constant influx of new news articles also leads to a perpetual cold item start problem. In addition, several news portals do not track users across sessions, which incurs a perpetual user cold-start problem. These problems have led to a focus on

non-traditional methods to recommend news, and with the development in neural NLP methods to learn deep representations of news, neural-based methods have been at the forefront of the state of the art in news recommendation. The goal of the task requires a medium to examine the effects of news location and user positions, and to avoid manual feature engineering; we will consider the neural methods.

3.1.1 Gated Recurrent Unit-based Model (GRU)

Okura et al. [44] proposed to learn representations of news bodies using autoencoders. The autoencoder is a neural network that learns efficient and accurate representations of unlabeled data, such as news bodies [20]. Okura et al. [44] train the autoencoders by taking a random base article, an article from the same category as the base article, and one article from a different category. The aim of the autoencoder is to represent the two articles of the same category more similarly than the base article and the article from a different category. The underlying assumption is that articles from the same category are more similar than articles of different articles. The autoencoder is penalized by the loss function L_T defined as:

$$L_T(h_0, h_1, h_2) = \log(1 + \exp(\text{sim}(h_0, h_1) - \text{sim}(h_0, h_2))), \tag{3.1}$$

where h_0 is the encoded representation of the base article, h_1 is the encoded representation of the article of the same category, and h_2 is the encoded representation of the article of a different category. Thereby the autoencoder is penalized as a function of the difference in similarity. This penalty clearly encourages the autoencoder to represent topically similar articles similarly.

Okura et al. [44] further proposes a news recommender method based on the autoencoder embeddings. The news article bodies are embedded by the autoencoders, which are used by a GRU [13] network to learn a user representation. The system subsequently takes the inner product similarity operation on the representations of the user and news articles, to get a prediction on the probability of the user clicking the article. An et al. [5] argue that it is very difficult for RNN networks such as GRU to capture all information in a very long news browsing history. Furthermore, Wu et al. [62] state that such a method is inefficient and cannot capture the context of words.

3.1.2 Deep Knowledge-Aware Network (DKN)

Wang et al. [56] make the observation that the news articles are full of knowl-

edge entities, but that the then-current state of the art did not incorporate it into news recommendation. Knowledge entities are entities in the text which contain specific knowledge, suitable for knowledge graphs. The authors pose the example of the article "Boris Johnson Has Warned Donald Trump To Stick To The Iran Nuclear Deal" should indicate a strong possibility of interest in the article "North Korean EMP Attack Would Cause Mass U.S. Starvation". Although the articles themselves contain no specific correlation on words, the articles contain several related knowledge entities, in "Donald Trump", "Boris Johnson", "Iran" and "Nuclear Deal", should all map to "North Korea" "Congressional", "EMP" and "US". Wang et al. [56] propose that a knowledge-aware system would solve these problems, as the system would embed the entities based on their relation to other entities, causing such words which are related closely in knowledge graphs to be embedded closely as vectors in a low-dimensional space. Furthermore, DKN uses the embeddings of the entities in a convolutional neural network to create a news embedding. To generate the user profile, the news embeddings of the user history are applied to an attention net, where the attention weights are generated by a neural network applied to the respective user history items, and the candidate news. Finally, a second neural network is applied to the concatenation of the embedded candidate and embedded user profile, to receive the predicted click probability.

The main difference between DKN and the rest of the state of the art is the usage of a knowledge graph to embed the entities in the news title. The desired outcome of the embedding is similar to that of the GRU method. However, the GRU method must learn these embeddings based on the words in articles based on categories, while DKN uses external pre-defined knowledge to learn the embedding.

3.1.3 Long- and Short-Term User Representations (LSTUR)

An et al. [5] note that the interests of online users are diverse and varying in durability, as was previously shown by Li et al. [36, 37]. An et al. [5] propose that although both long-term and short-term interests are important for personalized news recommendation, distinguishing between them may help to provide more accurate user representations. To incorporate the presence of both long-term and short term-interests, LSTUR models short-term interests from recently clicked news, and the long-term interests from the whole click history. The short-term user profile is learned from a GRU network which captures the sequential news reading patterns of the short-term history. The long-term user history is implemented as a user ID embedder, where the representation for the long-term profile of each user is learned during training, as a mapping from the user id to a user profile. The process of generating the long-term user representation is formalized as follows:

$$\mathbf{u}_l = \mathbf{W}_u[u], \tag{3.2}$$

where $\mathbf{u}_l$ is the long-term user representation, $\mathbf{W}_u$ is the look-up table of the embedder, and u is the user ID.

An et al. [5] propose two methods of unifying the long-term user representation and the short-term user representation. The first method uses the long-term representation to initialize the hidden state of the GRU network. The final user representation is the output of the last GRU-cell in the GRU network. The second method uses a random initial hidden state for the GRU network, and then concatenates the GRU network output with the long-term user representation to form the final user representation. To get a score of a candidate news, the model takes the inner product of the final user representation and the news encoded candidate.

3.1.4 Neural News Recommendation with Personalized Attention (NPA)

Wu et al. [61] observe that informativeness of both words and news vary highly. The news read by a user do not all reflect the user's interests to the same degree. Similarly, not all words in a news article inform of the contents of the news article to the same degree. In addition to this, the same word in a news article may be of varying relevance to different users (e.g. the title "celebrity wins the lottery" may cause an entertainment enthusiast to click because of the word "celebrity". However, someone else may click the article because they are interested in the lottery). Therefore, Wu et al. [61] examine the utility of modelling word and news informativeness for revealing preferences of users in news recommendation, by proposing a neural approach with personalized attention. The personalized attention is based on the user ID, which is embedded into a representative vector. This vector is used as the input to two separate dense layers, which creates the word-level attention query, and the news-level attention query. The word-level personalized attention uses the user-specific query to weight the embedded words. The aim here is that the user-specific query will weight heavily the words which the user will find informative, and therefore create a better, personalized news representation. The news-level query is used similarly in the user encoder, but is used to apply the personalized attention on news representations. The resulting vector from the news-level personalized attention is the user representation. To score the candidate news, the dot function is applied between the user representation and the representation of the candidate news. The dataset used for the experiment of NPA is highly imbalanced in favor of negative candidates, as most people are presented with more news than they read. Therefore, Wu et al. [61] propose to

formulate the task of news recommendation as a $K + 1$-way classification task, where K is the ratio $\lfloor \frac{\#negative}{\#positive} \rfloor$, meaning the model is given several candidates, of which one is positive, and the model predicts the click probability using the softmax function, which is defined as follows:

$$p_i = \frac{\exp(y_i)}{\sum_{j=0}^{K} \exp(y_j)}, \tag{3.3}$$

where p_i is the score of candidate news i, and y are the non-normalized scores from the inner product between the user representation and the news representation.

3.1.5 Neural News Recommendation with Multi-Head Self-Attention (NRMS)

Wu et al. [62] note three observations as motivation for their work. (1) The interactions between words are important, especially for context. The words "Rockets" and "Bulls" change meaning when they are considered in relation to each other, when the meaning transforms from spaceships and bovines to professional basketball teams. Wu et al. [62] argue that the use of CNNs in DKN [56] does not allow the model to make these relations across the whole article, as CNNs are local in nature, whilst attention can be global. (2) Furthermore, the observation on interactions is also valid for news articles, as several news articles relate to each other thematically. (3) The last observation of Wu et al. [62] is similar to the one made by Wu et al. [61], that the informativeness of words in a news article varies. To address these problems, Wu et al. [62] propose a neural news recommender based on *multi-head self-attention*. As has been the case for most of the state of the art of personalized news recommenders, the recommender is based on a news encoder that takes in news and outputs news representations, a user encoder that takes in news representations and generates a user representation, and the click predictor which compares a candidate news representation with a user representation and predicts the click probability of the user on the candidate news. The news encoder uses a word embedding layer to embed the sequence of words into a sequence of representative vectors. The encoder then uses multi-head self-attention to learn the contextual representations of words by capturing the interactions between them. Self-attention is a variant of the attention layer, where the inputs are additionally used as the queries for the attention model [55]. This means that the model computes an attention weight based on its interaction with other words in the news, which is an effective way of capturing the interactions between the words [59]. Furthermore, the self-attention is multi-headed, meaning it is done several times with a trainable weighting of each head to the output, allowing for

the model to capture significant interactions between a word and multiple others, whereas single-headed self-attention is limited to one head to model interactions, which means two significant interaction with a word, deemphasizes each other [59]. The combination of these two effects aims to incorporate contextual information to address the first observation. A separate additive attention network with a trainable query is used to unify the output vectors of the self-attention. Wu et al. [61] found that using additive attention to unify several word representations to a single news representation was effective in incorporating the aspects of Observation (3), and is therefore reused by Wu et al. [62]. The resulting vector is used as the news representation. The News encoder is equal in structure to the news encoder, but uses the contextual news representation generated by the news encoder as inputs and outputs of the user representation. In the same way that the user encoder addresses Observation (1), the news encoder addresses Observation (2). The final step in the model is the click predictor which uses the dot product of the candidate news representation and the user representation to generate the predicted click probability of the user on the candidate news. The experiment of NRMS uses the same dataset as NPA [61] and follows in formulating the recommendation task as a $K + 1$-way classification task to utilize the large amounts of negative samples in the dataset.

3.1.6 Neural News Recommendation with Attentive Multi-View Learning (NAML)

Wu et al. [60] view the then-current state of the art to be limited to a single type of information on the news articles. Several models only utilize the title [62, 61, 5, 56] or body [44]. Furthermore, the different types of news article information are very different in structure and meaning, and should therefore be treated differently. Moreover, Wu et al. [60] makes the same observation of different informativeness for each word, as Wu et al. [61, 62]. To address the first observation, NAML takes the input types: the title, the body and the news categories. Wu et al. [60] designed the model with a multi-view structure, where each input type is regarded as a different *view* of the news item. This also carries an individual encoding structure for each type, which are unified by an attention mechanism to a single news representation. This flexibility allows for future extension into other input types, simply by adding another view. The user encoder is a news-level additive attention network with a trainable query. The attention networks address the third observation, as seen in NPA [61] and NRMS [62]. The experiment of NAML uses the same dataset as NPA [61] and shares the observations made on the imbalance dataset. Therefore, Wu et al. [60] has defined the recommendation task in NAML as a $K + 1$ classification task to utilize the large amounts of negative samples in

the dataset.

A more detailed explanation of the implementation of the underlying structures will be given in Section 4.1.

3.2 Location-Aware News Recommenders

In this section we will examine the state of the art in location-aware personalized news recommenders, focusing on the spatial aspects of the recommenders, both the user position and the news location. The focus on these aspects is based on the thesis goal.

3.2.1 Spatial Topical Preference Model (STPM)

Noh et al. [43] describe the setting of someone who works in a finance company and reads business news relating to their job when they are at work, but reads entertainment news at home. Therefore, they posit that position is one of the most important determinants of user preferences. However, most state of the art in news recommendation at the time, such as [35, 36, 45], focused on topic modelling of user preference and disregarded the position of the readers. Therefore, Noh et al. [43] proposed to learn user profiles for each of the corresponding locations (e.g., home, gym, office, bar). This would solve the split-personality nature of news preferences, by treating each position as a separate topic interest area. However, as STPM learns directly from the user's history in the specific position, it suffers from the cold-start problem each time the user visits a new position, as it has no previous history there [12].

3.2.2 CHAMELEON

CHAMELEON is a deep learning meta-architecture for news recommender systems [14], which achieved state of the art results at the time. In the module where the spatial information is used, the articles and interactions are fed into a feed forward neural network together with all the contextual factors, such as device, previous clicks in the session, the article popularity and the article recency. The output of the FFNN is an individualized article embedding, which is used as a candidate or as a part of the user history. Although this is a state of the art method, the complexity of the implementation of the user position makes it hard to clearly discern the effect of the user position, as well as the overall complexity of the meta-architecture leading to significant overhead on the work required for the experiment. Nevertheless, Gabriel de Souza et al. [17] examined the effects of different input feature configurations on the recommendation quality, and found

that the addition of user context was beneficial in accuracy metrics and a novelty metric, however, on the diversity metrics, the addition did not significantly benefit the recommendations, and in some cases lowered the score. The user context contains user position as one of seven features and shows that further research on this topic specified towards user position is necessary.

3.2.3 Location-aware Personalized News Recommendation with Explicit Semantic Analysis and Deep Semantic Analysis (LP-ESA & LP-DSA

Chen et al. [12] argue that the research effort on location-aware recommender systems emphasizes recommending news located close to the user, and places little to no emphasis on the personal preferences of the user. This results in all users in the same location receiving the same recommendations regardless of the user itself. Furthermore, although personalized news recommenders existed, these did not utilize spatial information. Therefore, Chen et al. [12] propose two hybrid methods: location-aware personalized news recommendation with explicit semantic analysis (LP-ESA), and location-aware personalized news recommendation with deep semantic analysis (LP-DSA). These methods attempt to combine the location-aware aspect with personalized news recommendation to create a unified accurate recommender that incorporates user preference and user context.

The methods employ explicit semantic analysis (ESA) to make the recommendations location-aware. ESA is a method that explicitly represents the meaning of any text, such as news articles, in terms of Wikipedia-based concepts [18]. LP-ESA and LP-DSA use these concepts as topics by which to represent both news articles, users and locations.

When a user u is at a location l, the system generates a general user profile based on the user's news history projected onto the Wikipedia topic space. This gives a distribution over topics indicating the user's preference. A similar operation is done to the candidate news, which gives a general news profile based on the Wikipedia topic space. The location-awareness is introduced by the local topic distribution. The local topic distribution for location l is generated by collecting a set of documents that are geotagged with the location l. These are then projected onto the Wikipedia topic space to get the probability distribution over the topics, given the location, $p(z_i|l)$, where z_i is the topic and l is the location. The two proposed methods now diverge. LP-ESA uses the local topic distribution with the general user profile to generate the localized user profile, which reflects the user's topic preferences at location l. Furthermore, it uses the local topic distribution to integrate the location topics into the news profile, creating the localized news

profile. The relevancy score is defined as the cosine similarity of the localized user profile and the localized news profile. LP-DSA uses deep neural networks to map the profiles and distribution onto a low-dimensional, abstract and dense feature space. The abstract representations of the local topic distribution is impressed onto the abstract user profile and the abstract candidate profile, individually. The similarity function of the resulting profiles is the relevancy score.

The origin of the user positions and how these are perceived is unclear. The article states that city names are extracted from the news articles, and these are considered as geotags for the news article. However, it is stated that for each sample (u, v, l), where u is the user, v is the news article and l is a location, if the training set contains another sample related to the same user u and the same location l, then l is seen as a city that u has *visited* before. This makes it unclear whether l is regarded as a news location or a user position, or if the two are regarded as one and the same. Furthermore, if the preprocessing is done as described, with interchanging user position and news location, then an assumption is made that a user u who reads about a location l must physically be in location l. This assumption is not solid, especially in the current circumstances where a significant part of news articles concerns the Russian war on Ukraine, despite none of the readers being physically located anywhere near Kyiv. The authors were attempted reached regarding the matter, but no response has been obtained.

3.2.4 Dynamic Attention-Integrated Neural Network (DAINN)

A central assumption in several state of the art news recommender systems is that all events in the user history are equally important. However, as described by Zhang et al. [69], this does not take into account certain news articles being more descriptive of user preference than others, such as large breaking news are less informative of long-term interests, as well as the implications of a real-world scenario, such as accidental clicks and temporary curiosity. Therefore, the proposed DAINN consists of three modules: session-based public behaviour mining, user long-term interest modelling, and dynamic spatio-temporal attention. The session-based public behaviour mining is motivated by the statistic from their dataset that less than 20% of users are subscribers and therefore have long-term historical records [22, 69]. Therefore, the sequence of events within a specified time frame of the user interaction is encoded into a representation of behaviour patterns of the general user base. The long-term user interest modelling attempts to model the user's long-term interests by looking at the user's historical records and using topic modelling to generate an embedding of the user's topic interests. To integrate spatio-temporal context, DAINN employs an attention scheme, where the time of day and location are embedded, and used as the query in an attention network.

This means that the user's click history is weighted with the spatio-temporal context (e.g., the attention weights soccer news on the weekends and business news when the user is at work.) Finally, the click predictor, which is implemented as a layer of Gated Recurrent Units (GRU), generates predictions on top-k items.

3.3 Concluding Remarks

Although the state of the art in location-aware personalized news recommenders is broad, the implementations of user position and news location is often complex and abstract from the view of the output, such as DAINN. Although DAINN employs the user position in the model, and uses a dataset which contains news locations, the user position is implemented deep in the model, which hinders a clear display of the user positions effect on the final prediction. Therefore, it is not a model fit to use as the baseline method for the research of this thesis. The complexity and abstraction of these methods do not comply with the goals of this thesis. Achievement of the goals of the thesis, and to test our hypothesis, requires a news recommender that has a simple and comparable implementation of the news location and user position, and where the effect of the spatial addition is clear from the view of the output. To the extent of our research, no location-aware personalized news recommenders fit our requirements. Therefore, we select a state of the art personalized news recommender to implement and extend with user position and news location modules, to examine the effect of the spatial data clearly. NAML is a state of the art personalized news recommender which employs a simple multi-view architecture. The method's flexibility, simplicity and modularity makes it a strong candidate to use for examining the effects of news location and user position. Although the model itself does not incorporate a location aspect, the structure is viable to extend with a custom location encoder which would allow for controlling the usage of the encoder to keep the comparison fair. Therefore, we use the NAML model and extend it to align with the goals of this thesis.

3.4 News Datasets & Sampling

3.4.1 Adressa Dataset

The Adressa Dataset for News Recommendation [22] is a public (available upon request[1]) dataset for recommendation of Norwegian news articles. The dataset comes in two versions: the larger 20M dataset of 10 weeks of traffic on the Adresseav-

[1] `http://reclab.idi.ntnu.no/dataset/`

isen news portal, and the small 2M dataset containing traffic of one week [22]. Gulla et al. [22] note the most prevalent recommendation datasets at the time, such as Netflix [7], Yahoo! Music [16] and Movielens [23], focus on enabling collaborative recommendation techniques, but lack enough textual information for content-based techniques [22]. Gulla et al. [22] argue that news recommender systems are time- and location-dependent, make use of implicit feedback, and often combine content-based and collaborative components. The Adressa Dataset includes content data in the article information, such as the title, body, categories, as well as keywords and location generated by an NLP algorithm. Furthermore, it contains contextual information on both the interaction itself, such as time of interaction, and the user, such as position and device. The feedback from users on articles is implicit and is suggested to be implied from click counts and reading times [22]. The reading times are logged in the interaction, and the click counts can be generated from the amount of interactions by a certain user u with article i. The dataset includes interactions by a users on articles, but it does not contain shown-but-not-clicked articles. This entails that it does not contain any direct negative samples.

3.4.2 MIND

The MIcrosoft News Dataset (MIND) [65] is the largest open benchmark on news recommendation with rich textual features [70]. The MIND dataset is collected from user behaviour logs on Microsoft News [2] of $1,000,000$ randomly sampled users over a 6 week period. The dataset contains rich textual features such as title, body and categories. Similar to the Adressa dataset, MIND also contains named entites extracted with internal NLP-based tools. The MIND dataset does not keep data on the reading times. The implicit feedback supplied in the interactions comes from the click itself and the amounts of clicks.

Although MIND is the largest and most commonly used dataset, it does not contain information on the user position. Therefore it does not suit the motivation of this thesis.

3.4.3 Twitter Dataset

Abel et al. [1] presents a dataset generated from the collection of tweets from $1,619$ users resulting in $2,316,204$ tweets related to $63,485$ news articles. Chen et al. [12] further extract city names from the news articles, which it uses as geotags. This dataset does not contain any explicit information on the position of the users, however Chen et al. [12] use the news locations generated from the dataset entities as the position of the user. In contrast to MIND and Adressa, the

Twitter dataset contains the action of posting the tweet as the implicit feedback of interest/relevance.

3.4.4 Private Datasets

Several of the state of the art methods use internal and private datasets. Although not usable for external parties, these are important to be aware of to provide context for the data with which the state of the art is developed. A dataset generated from news traffic on Microsoft News [23] over the span of a month is used by [62, 60, 61]. Okura et al. [44] sampled 12 million user from Yahoo! JAPAN [4] over two weeks resulting in a dataset of over 2 million articles and over 1 billion interactions. Wang et al. [56] use a dataset collected from the server logs of Bing News [5] and contains $1,025,192$ interactions from $141,487$ users on $535,145$ news articles.

3.4.5 Choice of Dataset

The hypothesis and research questions of this thesis are related to the performance of recommenders with user position information and therefore a valid dataset must have position information on the users. As shown in Table 3.1, MIND is the largest dataset. It is also the most commonly used open dataset, but it does not contain information on the user position, as seen in Table 3.2. The Adressa dataset is a large open dataset which contains user location information. Furthermore, most state of the art methods employ news article clicks as the target. Therefore, selecting a dataset that provides this information allows a broader range of methods for selection to extend, without major modifications of the baseline. Due to these factors, we use the Adressa dataset for the experiments. Specifically, we use the smaller Adressa 2M dataset to align with the computational resources available. The drawback of the Adressa dataset is a lack of negative samples, which requires that we perform negative sampling. Negative sampling will be covered in the next section.

[3]Although collected from the same news portal as MIND dataset, they are collected over different time periods.

[3]`https://microsoftnews.msn.com`

[4]`https://www.yahoo.co.jp`

[5]`https://www.bing.com/news`

Dataset	# Articles	# Interactions	# Users	Density (%)
Adressa (20M)	48,486	27,223,576	3,083,438	0.0360
Adressa (2M)	11,207	2,286,835	561,733	0.0182
MIND	161,013	24,155,470	1,000,000	0.0150
Twitter	63,485	98,321	1,619	0.0957

Table 3.1: Quantitative comparison of open datasets, density is given by $\frac{\#\text{Interactions}}{\#\text{Users} \cdot \#\text{Articles}}$

Dataset	Language	Content	Ratings
Adressa	Norwegian	Title, Body, Categories, News Location, User Position, Entities, Keywords	clicks, read time
MIND	English	Title, Abstract, Body, Categories, News Location, Entities, Keywords	clicks
Twitter	English	Title, Body, Entities	posts

Table 3.2: Qualitative comparison of open datasets

3.5 Negative Sampling

In datasets with a lack of negative labels or a clear indication of non-relevance, one must generate negative candidates by negative sampling. This is especially common in implicit feedback datasets, such as news recommendation.

Hu et al. [27] state that one of the unique characteristics of implicit feedback is that there is no negative feedback. By observing interactions of the users, a recommender can infer which items a user finds relevant, but the interactions alone hold no explicit information on the non-relevancy of an item. This is relevant in the domain of news recommendation, as most large-scale datasets only log implicit user feedback, most common in the form of user clicks and impressions on the user [65]. However, most state of the art news recommenders, make the assumption that the news article selected by the user is deemed more relevant than the articles shown and not selected by the user [60, 61, 62]. Furthermore, some state of the art news recommenders assumes that the article selected is more relevant to the user than a randomly selected article [69, 38]. This assumption enables use of the information in the context of a supervised learning problem, where we have the interaction as a positive indicator ($y = 1$), and an impression without interaction as a negative indicator ($y = 0$). However, as a user is typically shown many more articles than which are read, the assumption leads to an imbalanced dataset [61]. Furthermore, some datasets do not log impressions, but only clicks, which leaves solely positive

candidates. Negative sampling techniques are often used in these cases, either to generate negative samples from the item population [26], or to balance the dataset in order enable efficient and scalable learning [67]. Typically, there are three main approaches in negative sampling methods: Random Negative Sampling, Popularity-biased Negative Sampling and Model-based Negative Sampling.

Random Negative Sampling Random negative sampling is randomly selecting items in the item set $A \setminus a_i$, where a_i is the correct item of the interaction, and A is the set of items. This is a computationally cheap approach which samples each item in the sampling pool with a uniform probability distribution. Zhang et al. [69] use the method to upsample negative candidates. Wang et al. [56], Zhai et al. [67] use random negative sampling to downsample a dataset which contains a large amount of negative candidates for each of the positive candidates, to a balance 1:1 ratio of negative and positive candidates. Wu et al. [61] argues that such random downsampling of the negative articles to balance the dataset both loses the rich information from all the negative samples, as well as leading to a computationally heavy procedure being added to the data generation during training. To address this, Wu et al. [61] randomly sample $K + 1$ negative samples and jointly calculates a click probability for each sample and approaches the problem as a pseudo $K + 1$-way classification problem. This approach is also employed in several other state of the art news recommender methods [60, 62, 5].

Popularity-biased Negative Sampling Popularity-biased negative sampling is a sampling method which is based on randomly selecting items in the item set $A \setminus a_i$, where a_i is the correct item of the interaction, with a probability distribution based on the popularity of the item. This entails that more popular items are selected as the negative sample more often than unpopular items. This is often employed with long-tail distribution of popularity, to counteract popularity bias in the training of the model. The popularity bias is often introduced where a model is shown certain popular items often as the positive sample in training, and then learns to recommend the popular items. However, if the model is shown the items a corresponding amount in the negative samples, this bias is avoided, at the cost of computational complexity.

Model-based Negative Sampling Traditional negative sampling methods are static during learning and is agnostic to the recommender model being trained. Dynamic negative sampling utilizes the models predictions to select the negative samples which provides the most information for the model [71]. This is done by examining the predictions of the previous round of training and selecting the items which the model scored highly, as negative samples. Zhang et al. [71] found that

dynamic negative sampling both reduces training time, but also lead to significant performance gains. A different direction was explored by Wang et al. [57], where a separate adverserial model was created to generate negative samples. The generator of the negative samples aims to create negative samples which are difficult to distinguish from the positive samples. The recommender model aims to discriminate between the positive samples and the generated samples. This creates a competition between the two models, which should benefit the performance of both models. Wang et al. [57] found a significant performance gain over the baselines.

Chapter 4

Technical Approach

This chapter will introduce the technical approach. Section 4.1 will introduce the details of the base NAML model. The Location-based extension of the base model will be described in Section 4.2. The base model and extension depend on certain custom layers, which will be introduced in Section 4.3. Lastly, the data generation and dataset sampling are detailed in Section 4.4.

4.1 Base NAML model

The NAML model is a neural news recommendation model which uses a multi-view structure to learn representation for news and users, and uses the representations to predict the probability a given user clicks a given news article. The model is shown in Figure 4.1. This section will be an extension of the description of the model given in subsection 3.1.6, and will focus on the modules of the method and their functionality.

4.1.1 Task

As described in subsection 3.1.6, the task of the original model was defined as a pseudo $K + 1$ classification task due to the large amount of negative samples in the dataset, meaning the model attempted to classify the K negative samples and one positive sample. To adapt the model to the dataset of this experiment, where there is a lack of native negative samples, K is set to 1, making the task a *binary* classification task. Thus, we follow the approach set out by the original implementation, adjusting the K to reflect the change in the dataset.

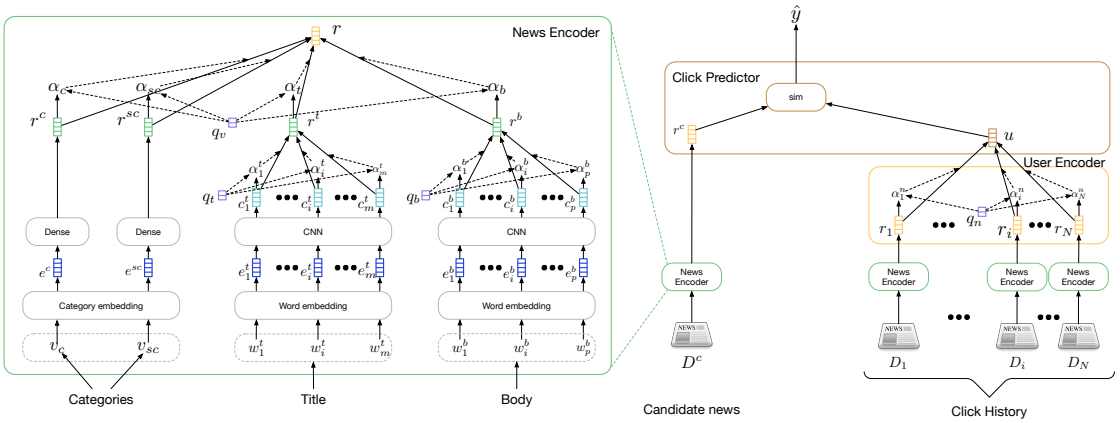


Figure 4.1: The base NAML model, as proposed by Wu et al. [60].

4.1.2 News Encoder

The news encoder is the module that encodes a news article and its contents to a vector representing the contents of the article. The structure of the NAML model is visualized in Figure 4.2. The news encoder is an attentive multi-view learning framework to learn news representations from the article data, where the category, title and body are implemented as separate views. The news encoder is used to encode the candidate news into a candidate representation, and to encode each news in the user history into a news representation which is used by the user encoder to generate the user representation. The encoders of the titles, bodies and categories are incorporated as separate views of the articles, with an attention network to combine the individual representations of an article's title, body and categories. The news categories are encoded by an embedder and a feed-forward network, annotated as "dense" in Figure 4.2, to generate the representations of the categories, denoted r_c and r_{sc}.

The encoders for the body and title are equal in structure. Each word of text is encoded by a word-embedding layer and a convolutional layer. The encoding of each word is subsequently combined by a word-level attention network to allow the model to learn which words are important and should be emphasized. Although the structure is equal, the variables of the CNN's and attention networks are not shared, enabling the model to learn different representation functions for the title and body. The same reasoning of using an attention network in the textual views is applied to the views themselves, where the informativeness of each type of news information varies for each news article. Therefore, an attention network is applied to the representations of title (r^t), body (r^b), category (r^c) and subcategory (r^{sc}), in order to weight the views in the final summation, by their calculated attention weights α_t, α_b, α_c and α_{sc}, respectively. A detailed description of the attention

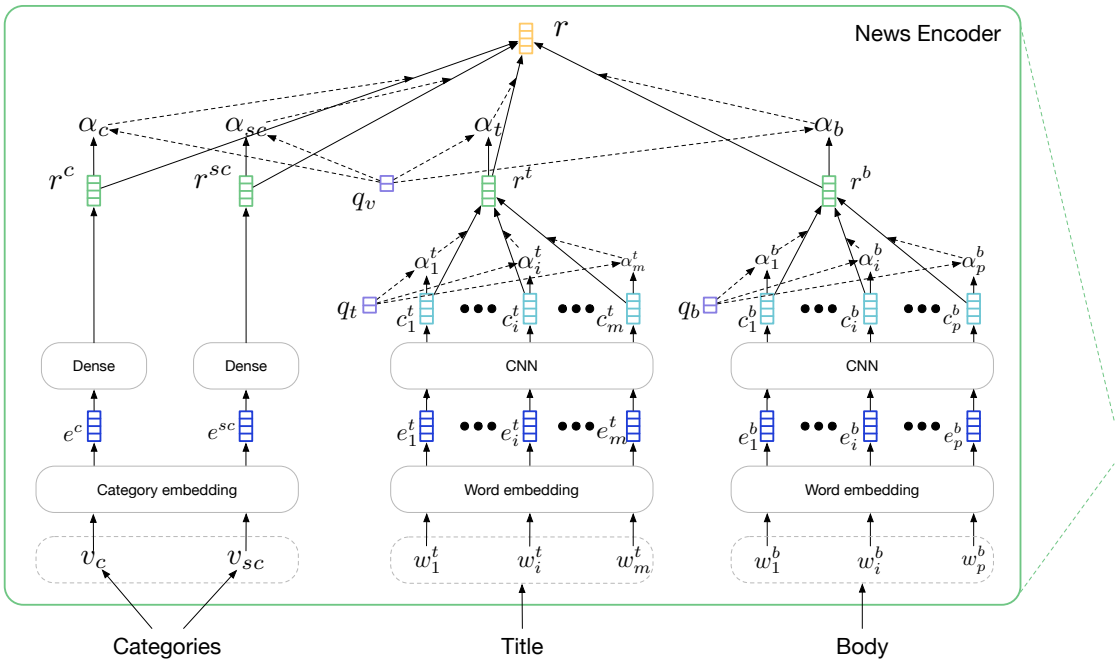


Figure 4.2: The news encoder of the base NAML model.

layer is provided in subsection 4.3.1. The output of the attention network, denoted r in Figure 4.2, is a unified representation of the news article.

4.1.3 User Encoder

The user encoder, shown in Figure 4.1, is a simple structure of a single attention network. The user encoder is applied over the user history, and takes in a representation of the news articles in the user click history, each individually encoded by the news encoder, as the inputs. These are denoted r_k for the k-th article in click history. The attention in the user encoder is applied to allow the model to weigh important and defining news articles heavier in the user representation, than news which generally indicate less about the user. An example of this is that an article about a football game would be more indicative of the user's preference than an article about a terrorist attack, as the latter is an article most users read, while the former is an article only users with a football preference would read. The output of the attention network is the unified representation of the click history, which is used as the user representation, denoted u in Figure 4.1.

4.1.4 Click Predictor

The click predictor is the final module of the original NAML framework. This module utilizes a similarity function, denoted "sim" in Figure 4.1, to estimate the click probability based on the inputs of the encoded user and the encoded candidate news, or the relevance of the candidate news to the user. The concept of using a similarity function is based on the assumption that a user and a news candidate which are a good fit are encoded similarly, or formally, the relevance R of a candidate news i for a user u, is given by some similarity function:

$$R(u, i) = \text{sim}(\vec{\mathbf{u}}, \vec{\mathbf{r}}_c), \tag{4.1}$$

where $\vec{\mathbf{u}}$ and $\vec{\mathbf{r}}_c$ is the user profile and candidate news profile, respectively.

Prediction of relevance based on a traditional similarity function between the news profile and the user profile is widely employed in the state of the art, despite its apparent simplicity. Similarity functions such as cosine similarity [15, 12, 14] and inner product [60, 61, 62, 48, 44, 5] are popular for the simplicity and efficiency.

More advanced methods have been used as well. Wang et al. [56] and Gabriel de Souza et al. [17] use a deep neural network between the user profile and the candidate news profile to estimate the probability of the user clicking on the news. The parameters of the DNN are trainable parameters of the recommender system. Zhang et al. [69] use a GRU (Gated Recurrent Unit) Network to generate top-k items with the highest click probability, based on the user's session-based representation, long-term interests embedding and a dynamic attention scheme.

In the original NAML method, Wu et al. [60] explored the usage of a multi-layer neural network as a click predictor, but found that the simple inner product was more efficient and provided better performance. To follow the baseline, the click predictor in the model in this thesis is implemented with an inner product.

4.2 Location-based Extension

To address the goals of this thesis, we design and implement a spatial extension to NAML, which encodes the news location and the user position, and utilizes the encodings for the classification. The extended model is shown in Figure 4.3 without the encoder details, and in Figure 4.4 with the encoder details. Recall from Section 1.1 that we use the term "position" to refer to the user, and the term "location" to refer to the news. This will be important for the distinctions made in the approach. The extension has a main focus on the comparison of the outcomes for using the news location or the user position. Therefore, the design should have certain key characteristics. Primarily, the extension should be designed as fairly

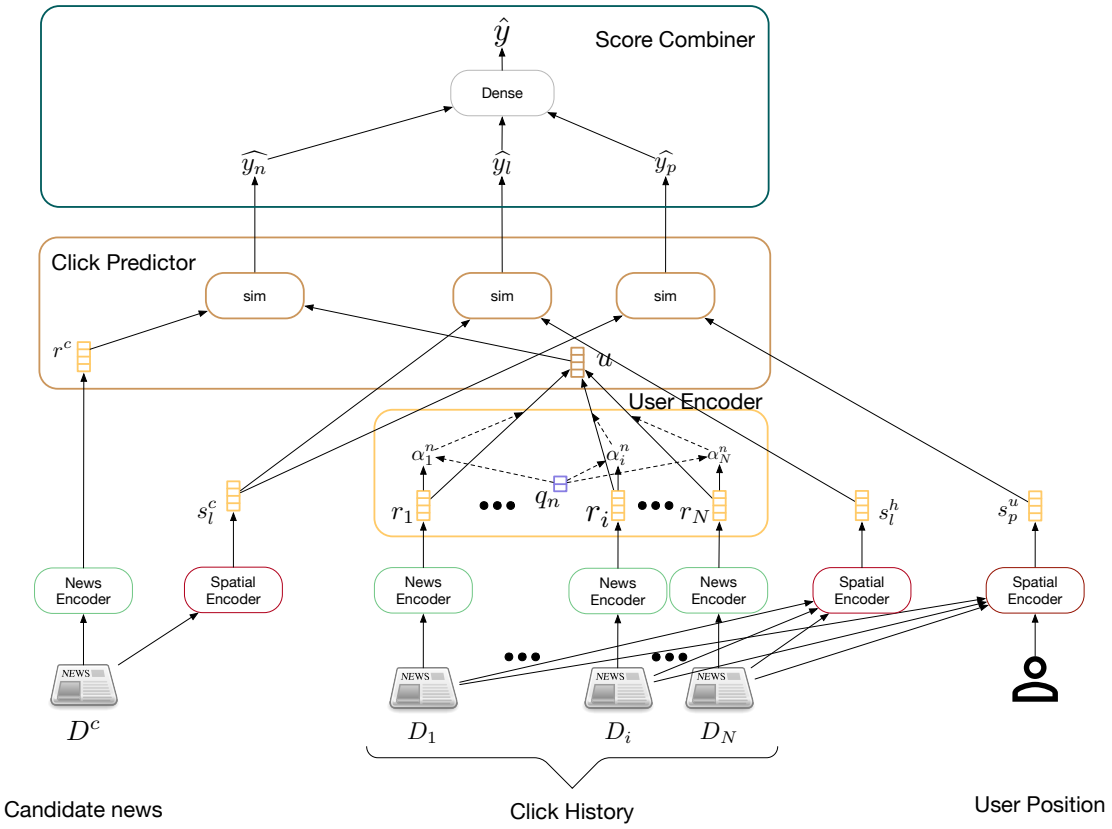


Figure 4.3: The full model including the base NAML model and the spatial extension, excluding the details of the encoders.

as possible in regards to the usage of the user position and the news location, as to avoid introducing any bias to the experiment. Furthermore, the extension should include a clear path for the user position encoding and the news location encoding to the output. This gives the total model the clear possibility of using the spatial information, and will give a clearer picture of its impact on the predictions.

4.2.1 Spatial Encoder

The spatial encoder, shown in Figure 4.5, is the module which takes in spatial information and generates a vector representing the combination of these values. The spatial encoder is used as the encoder for the locations and the positions equally, to keep the comparison as fair as possible. Implementationally, there is a small deviation in how the user position and the news location are treated by the encoder, as the input formats are slightly different, however all significant operations (transformations and embeddings) are the same.

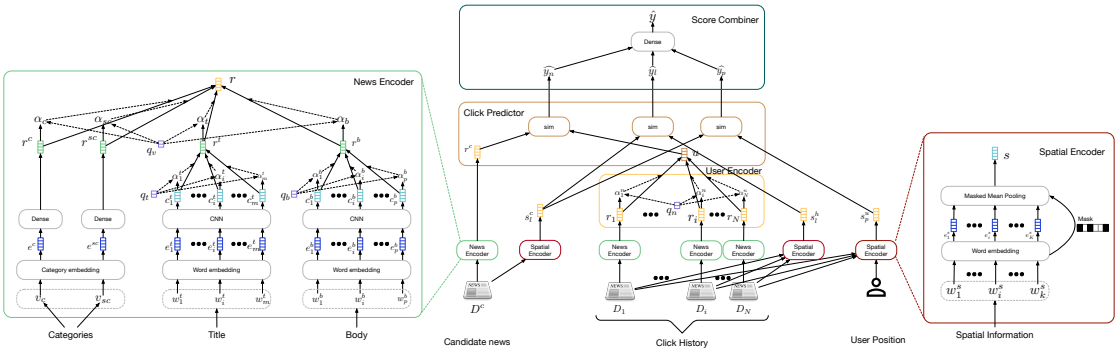


Figure 4.4: The full model including the encoders.

The spatial encoder takes in the textual representation of the spatial information, w_i^s, as input and uses the shared word embedder to embed each of the locations separately to create the spatial embeddings e_i^s. The embeddings are then averaged by the custom Masked Mean Pooling layer (see subsection 4.3.2) to receive a single representative embedding s which is the final representation output of the encoder.

The design of the encoder is chosen to use an embedder to limit the need of external knowledge. Many sota methods use external information to model locations [12, 56]. However, as we wish to view the specific impact of locations and positions, we do not involve external knowledge bases, and the system is simply using an embedder. The embedder functions similarly to a knowledge base, but without the explicit categories of information. It rather learns a vector-representation for the locations [12], by use of the large corpus of text available from the base of the news article. As the word embedding trained through the news articles contains information on the locations mentioned in the article in the same way as any other word, the embedder used in the news encoder is reused in the location extension. This allows the location encoder to employ the vast amount of information in the news articles to gain a meaningful representation of the locations.

The location encoder uses an averaging operation to reduce the embeddings of potentially many locations of a news article to a single unified representation. The max operation was considered as well, but the averaging operation was deemed more suitable for the task. The essence of the task of news recommendation is to understand the preferences of the user. A significant problem with the max operation over an axis of a sequence is that it does not consider repeated elements different than singular elements, which means that repeated interest in a location is not distinguished from a single occurrence. A user history of ten news articles where one is about Trondheim and nine are about Stavanger, is given the same encoding as a user history of ten news articles where nine are about Trondheim

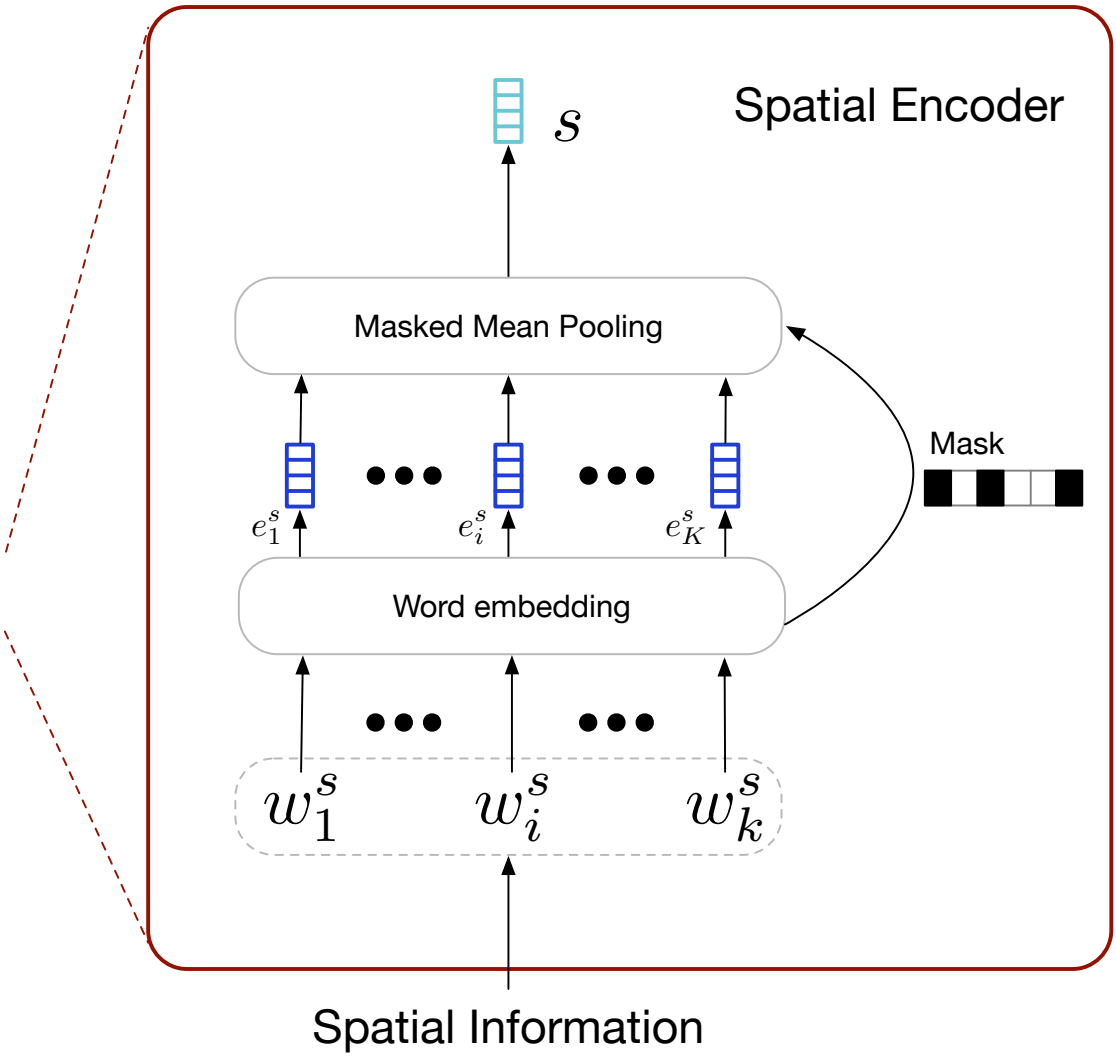


Figure 4.5: The spatial encoder used in the full model to encode the user position and the news location.

and one is about Stavanger. This is especially important in the task of news recommendation where repeated interests should be weighted highly compared to a single interaction, as in real-world scenarios, a user is prone to mistakes and short-term curiosity [69].

The location encoders are consistent, which means that if $enc(L) \rightarrow V_L$ for a given L at one point in the model, then $enc(L) \rightarrow V_L$ always holds. This means that if the user has previously read about the locations L, and a candidate news which is about the locations L, then the location encoder of the candidate news and the location encoder of the user history will encode the same representation.

Furthermore, the spatial encoder is order-agnostic which means that if L_1 is a sequence of locations and L_2 is any permutation of L_1, then $enc(L) = enc(L_2)$. This is done to allow all parts of the historical locations or positions to be represented in the encoding, ensuring that if a location signal exists, it is not weighted out of the final encoding. This is based on the goal of the thesis, to examine the effect of user position and news location, rather than to build the highest performing news recommender. Furthermore, the attention layers in the base NAML model are order-agnostic, and seeing as the locations are not necessarily more order-dependent than the browsing history, the order-agnosticity of the overall model is kept.

4.2.2 Location Click Predictor

The click predictor in the location extension of the model is implemented equal to the implementation of the base method, i.e., using an inner product. The same assumptions and goals as the base method holds for the location extension, where the encoding of a historical interest in news about certain locations should have high similarity to the encoding of the location in the news. This is grounded in the encoding method of the locations, where the news-location embedder on the candidate side is equal to that of the location-interest-based user profiling module. It follows that a historical interest in a location will have high similarity with the location of a candidate news about the same location.

4.2.3 Score Combiner

The score combiner is the module which combines the click probabilities from the separate categories into a single unified click prediction, which takes into account the news contents, the news-location interests and the user position. Specifically, the model takes in the click prediction based on news preferences, $\hat{y_n}$, the click prediction based on location interests, $\hat{y}_l$, and the click prediction based on user position, $\hat{y}_l$, see Figure 4.3. The model then combines the score by using the

combination function, which is implemented as a neural network. The selection of a neural network as the score combiner function is based on a hyperparameter optimization, further described in subsection 5.2.4.

Flexibility of score combiner

The score combiner adds flexibility to the model for the test, as it allows several inputs, but can also operate using only a single score. In addition to this, the score combiner is implemented as a module. The combination of these makes the score combiner more autonomous, as its only parameter is the operation type. The model takes a variable input, and combines them using the given operation to a single output score. This modularity and flexibility removes the need for intricate instructions for the ablation study, where pieces of the model prior to the score combiner is deactivated, altering the amount of inputs to the score combiner.

4.2.4 Embedder

A central part of the model is the word embedder. In the model, the word embedder is responsible for 777 000 of the 803 075 trainable parameters when implemented with the basic keras embedder. In addition to this, a large amount of textual data is readily available in the dataset from the titles and bodies. Therefore, the available data was utilized to create a trained embedder with the aim to reduce the amount of trainable parameters, and ease the training task of the model. The embedder was implemented as a Word2Vec model, using the gensim framework [49]. Word2Vec is a technique for learning vector representations of words from large amounts of text by using a neural network [41]. The option of embedder, trained Word2Vec or trainable keras embedder, was passed to the hyperparameter optimization. It turned out so that the hyperparameter optimization achieved lower validation loss with the standard keras embedder than with the Word2Vec embedder.

The embedder in the spatial encoder is trained on linguistic semantics, as it uses the same embedder as the news encoder. However, this could have been done in several other ways, which also would be a valid proposition as the embedder in the spatial encoder, as to provide a more domain-specific embedding. Embedding the locations based on spatial features was considered, especially using 3D coordinates to map distance between the locations and position.

4.3 Custom Layers

The model was implemented with the keras functional API. However, certain intricacies of the model requires functionality not natively supported by keras or

TensorFlow. Therefore, these modules are implemented as custom layers.

4.3.1 Attention layer

The attention mechanism implemented in the base model is to the extent of the research not available for the newest versions of the utilized frameworks and packages. Therefore, these were reimplemented as custom layers. The attention layer is used in several parts of the model. In the title encoder and the body encoder the attention layer is used to provide attention over each of the words in the text. In the news encoder, the attention layer provides attention over the different *views* of the encoder, combining the title, body, category and subcategory. The attention layer is also used in the user encoder over the news in the user history. These three application areas have different types of inputs and produce outputs of different meanings, such as the attention layer in the body encoder takes word encodings as inputs and gives a representation of the body as output, whilst the attention over the user history takes in the different news in the user history and provides a representation of the user history as output. Although these inputs, outputs and purpose differ, the underlying mechanism is equal. The attention mechanism takes in the input $\mathbf{x}$ of size N and computes the attention weight, α_i of each part of the input $\mathbf{x}_i$ by the equations:

$$a_i = \mathbf{q}_i^T \tanh\left(\mathbf{W} \times \mathbf{x}_i + \mathbf{b}\right) \tag{4.2}$$

$$\alpha_i = \frac{\exp(a_i)}{\sum_j^N \exp(a_j)}, \tag{4.3}$$

where $\mathbf{W}$ and $\mathbf{b}$ are the projection parameters, and $\mathbf{q}$ is the query vector. These are trainable parameters individual for each attention layer. Equation 4.3 is the softmax function over all input $\mathbf{a}_i$ in 4.3. The final calculation of the output is simply the average of all inputs weighted by the attention weight:

$$\mathbf{r} = \sum_i^N \mathbf{x}_i \alpha_i \tag{4.4}$$

Furthermore, to allow variable input size, such as a user with a short history or a news article with a short title, the layer is made compatible with zero masking. The zero mask of the inputs, described in subsection 4.2.1, is considered in the attention mechanism and removes the attention of the masked inputs. This results in the masked inputs receiving no attention, and therefore not affecting the softmax

calculation or the weighted average. This removes the influence of the padding applied to the inputs, so the maximum body size can be set high without the risk of the padding influencing representation.

4.3.2 Masked Mean Pooling

In the spatial encoder, a mean pooling function is used to average the embeddings over the inputs in order to obtain a representation of the same shape as the individual embeddings, with the encoding of all the embeddings. However, the dataset contains a variable amount of news locations, and a variable amount of historical user positions, but tensors must be of a fixed shape in all dimensions. Therefore, the inputs are appended pad tokens to fit the set size. Pad tokens are a special token which holds no meaning, but is used to fill the tensors to a fixed shape. This will skew the average of the embeddings to the arbitrary embedding of the pad tokens. To achieve a flexible model which allows a varying input size while mitigating the influence of the padding tokens, the spatial encoder uses a masking strategy. The encoder checks the inputs to the embedder for the padding token, and generates a mask, which is essentially a matrix of binary values indicating the presence of padding tokens. This mask is kept in the encoder to use in the Masked Mean Pooling layer. When the locations have been embedded, the embeddings themselves hold no trace of the originating word. Therefore, the spatial encoder uses the aforementioned mask to distinguish which embeddings to disregard during the operation. The result is the average of the embeddings of only valid locations. Thus, the spatial encoder can be flexible, despite the rigidity of tensors. The necessity of this implementation is clear in the example of a user with a single previous position, but where the maximum history size is large. The resulting encoding without the masking will not reflect the single position, but rather reflect the embedding of the padding token.

The spatial encoder uses averaging to reduce several spatial representations to a single unifying representation. This pooling could also use a max function, which would emphasize the large values in the embeddings, meaning stronger emphasis on the outliers. This could be a valid option, and would most likely push the model to focus on the extreme locations. The word embedding used in the location encoder attempts to build a reasonable representation of each word, including locations, which defines the word on unspecified parameters, where each of the definitive parameters would be a scalar in the embedding. One can imagine a embedding which through news embeds the locations to the topics of the news, such as Oslo and Trondheim would be embedded highly in the parameter of "Big City", and Trysil and Åre would receive high values in the parameter of "Skiing", however London would score very low. Considering this, in a location interests

history of [London, London, Trondheim, London, Trysil] with average function, the parameter of skiing would score low, despite the presence of Trysil, as the averaging function would lower it due to the presence of London thrice. If the model notices that the a certain topic, such as skiing, is important, then the model and specifically the embedder can learn to amplify the value of that topic, and then force the signal through, regardless of the average function. However, a max function does not consider repeated presences, and in the case of representing the user browsing history, the case of reading multiple articles about a location, should not be disregarded, as this may be one of the clearest implicit signals of relevance in news recommendation.

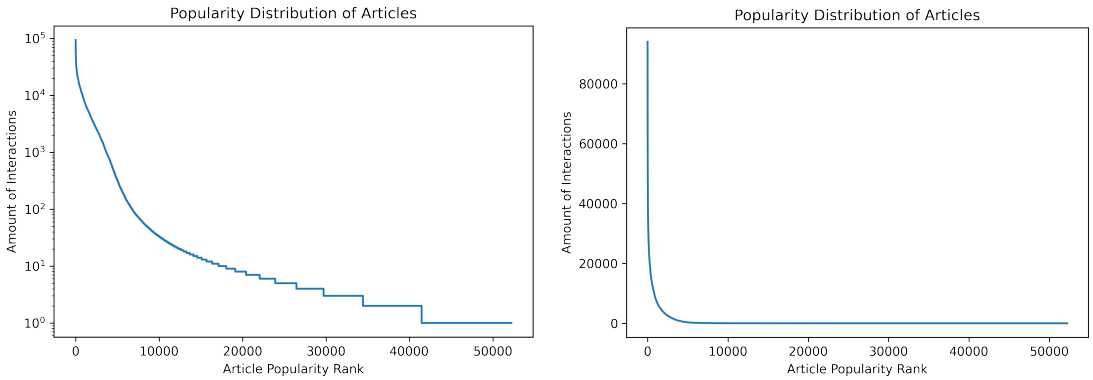
4.3.3 Slice

The input to the model is the interaction data and the article data for the candidate and the history. As the model is based on several submodels, and each submodel only needs a specific part of the data, the first layer of the model must deliver the correct data to the correct modules. This layer is the Slice layer. The Slice layer selects slices of the user data across the batch, and serves the data to the correct encoders. The layer is initially built for a one-dimensional input, such as the candidate news, but is combined with a *TimeDistributed* layer to function over the user history and the batch, in two dimensions.

4.4 Data Generator & Sampling

The data generator is the part of the model which creates and continuously serves the dataset to the model from the pre-processed data. The dataset of inputs and targets (x, y) is generated from iterating over the interaction log in chronological order. For each interaction in the log, the data generator retrieves the relevant article data for the interaction, as well as the data for the interaction history of the corresponding user up to the point of the interaction in question. This is used as a positive sample of user interest.

The data available is based on an interaction log and, therefore, does not contain explicit negative samples. To avoid an unbalanced dataset, the data generator infers negative samples by sampling a random article from the article base weighted by popularity. Popularity-biased negative sampling is further described in Section 3.5. The negative sampled article is swapped in for the positive article, with the same interaction information (user, time, user position). This allows for a equal comparison for the model, in which the interaction is proposed with both the actual article in the interaction as well as the negative sampled article.



(a) Logarithmic Y-axis (b) Linear Y-axis

Figure 4.6: The popularity distribution of the articles given by the amount of interactions with (a) logarithmic scale and (b) linear scale, with the articles ranked by the amount of articles.

4.4.1 Negative Sampling of the dataset

The Adressa dataset, as described in subsection 3.4.1, does not contain negative samples and requires negative sampling to balance the dataset. Therefore, negative sampling was utilized to generate negative samples in the dataset, to enable the system to learn to discern the relevant from the non-relevant. However, as seen in Figure 4.6 the majority of interactions is distributed among a small part of the articles. In the first day of the used dataset, the top 9 articles received more views than the other 4001 articles viewed in the day. On the full dataset, there were 17351 articles which were viewed at least once, however, the centre of mass of the distribution is at rank number 42.

These figures indicate that a popularity-biased random sampling is required to keep a fairness in the dataset, and to force the recommenders to learn to recommend news that are relevant rather than to remember which articles are popular. Therefore we implement popularity biased negative sampling in this work. Although more advanced methods exist , the implementation of such methods fall outside the scope of this thesis.

The popularity bias was based on the popularity of articles on the day of the interaction. In the domain of news recommendation, popularity has high peaks, but fades rapidly, i.e. an article about a significant event in January will not be a suitable negative sample in March, even though it was the most popular article in January. This is, though, partially offset already by the model being chronologically agnostic. The model randomly sampled articles based on the daily

popularity, but excluded the articles already read by the user.

4.4.2 Effects of Negative Sampling

The dataset used in the experiments is based on a log of interactions (clicks or views) of users on articles and does inherently not contain negative samples. However, the model requires negative samples to train. As discussed in Section 3.5 and subsection 4.4.1, popularity biased negative sampling was utilized.

Popularity based sampling does not fully remove the popularity bias, as the popularity biased negative sampling was done among the articles excluding the articles in the history of the user and the positive candidate. This incurs a slight popularity bias in the dataset, as the most popular samples will appear more often in the positive candidate and history, and therefore be in these cases ineligible for negative sampling. The incurred popularity bias is, however, less than during the random sampling.

Furthermore, the act of sampling the negative candidates from the article set induces an assumption that the article read by the user is more relevant to the user than any other article in the item set. This is a problematic assumption as the user is not presented the entire article set to make a selection from. However, the dataset does not log which articles are presented to the user.

The assumption is made more plausible by reducing the item space form which negative sampling is performed to consist of only articles read by any user within the same period. This removes mostly old articles and items which are most likely not presented to the user. The usage of popularity biased negative sampling prefers negative sampling of articles which are most likely to be shown to the user, as these articles must have been presented to other users for them to become popular.

In addition to this, the assumption does not need to hold for all cases for the model to learn efficiently. Every positive candidate may not be more relevant than every other article for the given user, but rather most positive candidates should be more relevant than the corresponding negative sampled article. Furthermore, a user does not always select the most relevant article each time, which incurs similar noise.

Chapter 5

Experimental Evaluation

5.1 Introduction

This chapter will detail the experiments carried out in this thesis. Section 5.2 will present the experimental setup and all the specifics of the experiments required to repeat the experiments and reproduce the results. The results will be presented and discussed in Section 5.3.

5.1.1 Ablation Study

An ablation study is the study of a machine learning model where components of the model are successively removed to view the impact of each component [40]. The experiment is performed as an ablation study. The model will be tested with and without three aspects of the model: the news location interests of the user, the current user position, and the historical user position. The base model of NAML will be present in all $2^3 = 8$ variations of the model. The ablation study will enable the examination of the effects of the different aspects of the model. This is in line with the aims of the thesis, and will answer **RQ3**.

5.1.2 Evaluation

To evaluate the performance of the models and, therefore, the validity of the hypothesis, the models will be measured on metrics based on their predictions on a test set. Recommender systems are typically evaluated on either their ability in information retrieval with such metrics as Precision, Recall and F-score, or their ability in ranking, with NDCG and MRR [3]. However, NAML is designed as a classification system where the model attempts to learn which out of a set of news

items is relevant to the user, and Wu et al. [60] use AUC as their main metric. This is further discussed in subsection 3.1.6. For reasons of fairness to the NAML architecture and the concepts, the extended model will primarily be evaluated on classification metric AUC, with supplementary information provided by mAP. The performance metrics will serve as a basis with which to draw conclusions on **RQ3**. The next chapter will describe how the evaluation is performed and how the experiments are set up, including the dataset and metrics.

5.2 Experimental Setup

This section describes and discusses the aspects of the experiments required for reproduction. subsection 5.2.1 presents the steps of preprocessing of the data to form the dataset, which is described in subsection 5.2.2. The use of evaluation metrics is outlined in subsection 5.2.3. subsection 5.2.4 presents the hyperparameter optimization and the resulting parameters, with additional hyperparameters included in subsection 5.2.5.

5.2.1 Preprocessing of the data

The Adressa dataset requires heavy preprocessing and conversion. The data consists of two directories of files, the content and the interactions, which are preprocessed individually. The attributes of the preprocessed content and interaction data is shown in Table 5.1 and Table 5.2, respectively.

Attribute	Type	Description
Article ID	string	The document Id
Title	string	The title of the article.
Body	string	The body of the article
Locations	list	The locations of the news article.
Category	string	The main category of the article.
Subcategory	string	The subcategory of the article.

Table 5.1: The attributes of the preprocessed content data.

Content Preprocessing

The preprocessing of the content is inspired by the approach taken by Gabriel de Souza et al. [17]. The content is stored in JSON files and must be extracted before preprocessing. The data is converted into the respective data types, from the values in the JSON files, where all the data is formatted as strings. Furthermore,

Attribute	Type	Description
Event Id	int	The event id.
Time	integer	The time of the interaction.
User Id	string	The id of the specific user in the interaction.
User position	string	The most specific user position recognizable by the embedder.
Article ID	string	The document id of the article visited in the interaction.

Table 5.2: The attributes of the preprocessed interaction data.

several values are empty strings, which are converted to the appropriate empty values for each type (e.g., an empty news location is set to an empty list, so as to be able to be accepted by the model). Furthermore, some news articles lack a publishing time, but contain a creation time. In such cases, we set the publish time to the creation time, assuming the times are relatively close temporally. The textual content are in some cases null values, which are handled by setting these to the empty string. The categorical data is encoded from the textual categories to IDs representing the categories. The textual data is tokenized using the Natural Language Toolkit (NLTK) package [39]. Furthermore, the body of the articles often start with the text "Saken oppdateres.", which is often used in articles that are being updated in live situations or on breaking news, and is removed from the article later. It seems the content data logs the first version of the article, as 8952 of the 73309 articles (12.2%) have this text in the first sentence of the body. This is removed from the text during preprocessing. In addition, several articles have the publishing time included in the body as text, but where all characters in the word "publisert" and the publishing time are separated by whitespace. These are removed during preprocessing.

Interaction Preprocessing

The preprocessing of the interaction data is mainly focused on the two larger issues of the available data: (1) the connection between the content data and the interaction data, and (2) the encoding of the user position.

The interaction data contains a property for the article's id in the interaction, but this is not consistent and has several missing values. Nevertheless, the canonical URL of the visited article is always present. Therefore, to connect the interaction data to the content data, a mapping between the canonical URL of the article and its article id is constructed and applied to the interaction data, to allow efficient lookup in the content data by the primary key of the content data, the article id. This also filters out the interactions with the home page, as the home page has no content data. This is done as the home page does not provide any rele-

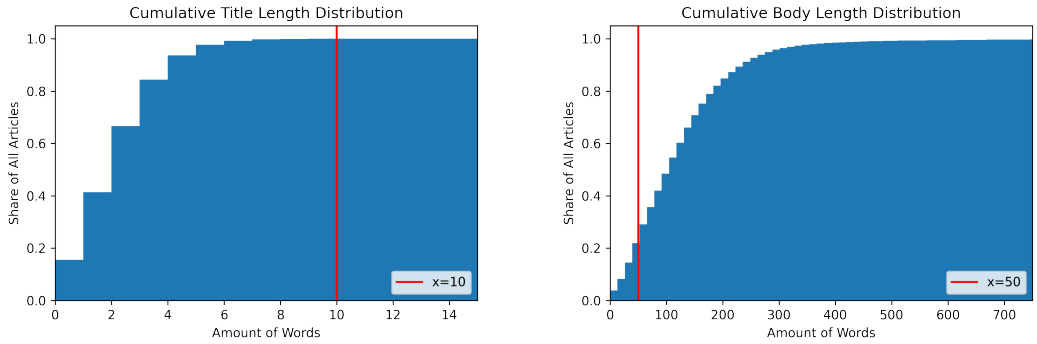
vant information in the scope of this thesis. Furthermore, articles of the category "Abonnement" and articles from the site "kundeservice.adressa.no" are excluded from the dataset. These items are advertisements, service announcements from the newspaper, and information about the premium service. These items are not news items and do not fit in with the recommendation task, and are therefore not included in the dataset.

The position of the user is given by the city, region and country. The model is given the most specific position recognized by the embedder, meaning it must be mentioned a minimum of 25 times in the corpus of all articles ($|A| = 73309$). This results in a user positioned in Malvik is recognized as in Malvik. However, a user who is in the town of Tjøme will be assigned the region of the user's location, Færder, as Tjøme does not appear enough in the corpus. Finally, a user in Quito will be assigned to the country of Ecuador. This is to be as specific as possible, while also recognizing the geographical limitations of a dataset from a local news portal. The country of the user position is denoted in the interaction data by a modified alpha-2 country code. This requires a custom manual mapping from the modified alpha-2 encoding to an encoding understandable to the model. In this case, the encoding was converted to the country's name, to enable it to be recognized by the embedder, as it relates to the locations in the news articles.

5.2.2 Final Dataset

The final dataset consists of three collections: the candidate news, the user histories and the targets.

The news candidates are vectors of length 73 containing the interaction and article data of the corresponding candidate. The input of size 73 is ten words from the title, fifty words from the body, the category, the subcategory, ten news locations, and the user position. The mentioned sizes are the maximums, and if there are fewer, padding is applied to fit to the set size of 73. In the case of the title and body containing more words than the maximum, the first ten and fifty are selected, respectively. The maximum title size was set based on the cumulative distribution shown in Figure 5.1a. The Figures 5.1a and 5.1b show the distribution of length in number of words for the article titles and bodies, respectively. The red lines show the maximum size set for the respective type. The maximum size of the titles and bodies were set based on these distributions. However, the maximum body size was primarily based on the hardware limitations, as the body data is responsible for 68.5% of the total data, and an increase in the size would be reflected in the computational cost in training. Notice, however, that although only 20.2% of articles contain 50 or fewer words, the dataset still retains 35.9% of



(a) Title (b) Body

Figure 5.1: The histogram of cumulative distribution of the amount of words in the article (a) title and (b) body. The maximum (a) article and (b) body size shown as red line in (a) as $x = 10$ and (b) as $x = 50$.

the total words.

The user histories are 10×73 matrices with interaction and content data of the corresponding interactions in the user history. The data in the user histories are on the same form as the candidate news, making the user history essentially a collection of up to ten previously read news articles. The user histories are kept in the same order as the candidate, meaning the i-th candidate news corresponds to the i-th user history.

The targets are the binary labels of whether the candidates are the suitable match to the corresponding user history ($y = 1$) or a negative sample for the user history ($y = 0$).

The combination of the news candidates, the user histories and the targets make up the dataset. The news candidates and the user histories combine to make the inputs in $X = [x_1, x_2]$, where x_1 is the news candidates, and x_2 is the user histories. X is then the input to the model.

Splitting the dataset

The final dataset of 7 days was split as the first five days (1. jan 2017 - 5. jan 2017) to training data, and the last two days (6. jan 2017 and 7. jan 2017) as the test data. The first two days of the training set were used in hyperparameter optimization, with the first day as training and the second day as validation, before the full five days of the training set were utilized in training. The split gives a

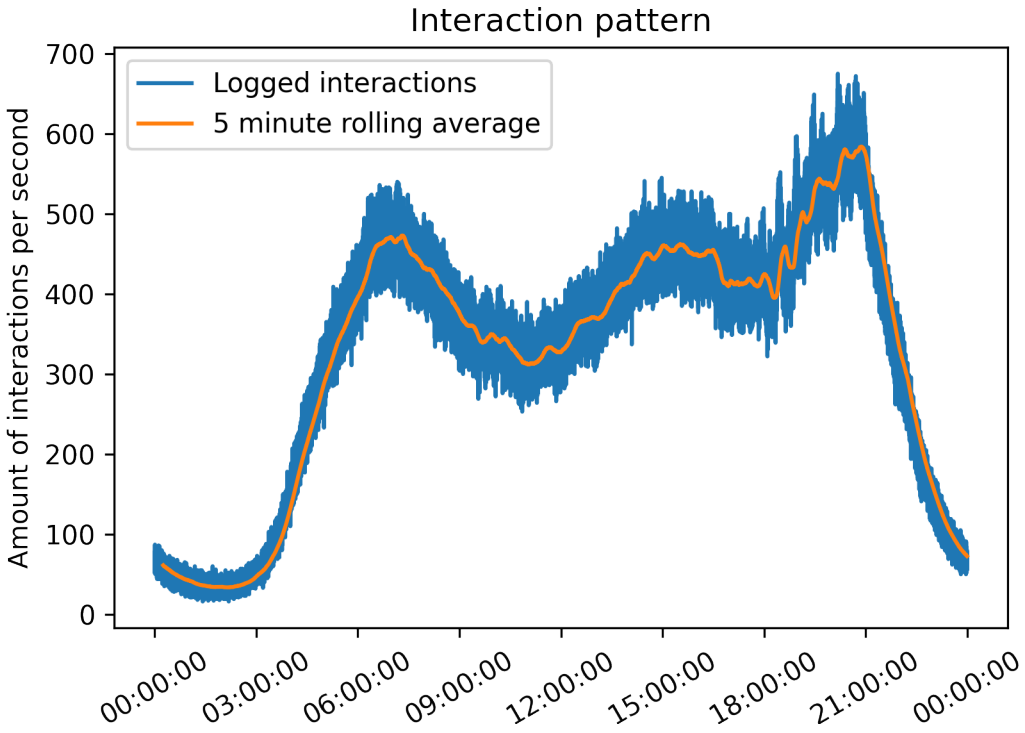


Figure 5.2: The time of day of interactions to show general viewing pattern across all days. The rolling average is shown in orange.

final training set of 1 425 331 samples, and a test set of 539 397 samples. The test set interactions are distributed almost equally, 268 794 and 270 603, over the two days, Friday and Saturday, respectively. The inclusion of two days rather than one in the test-set is to mitigate the influence of the inherently cyclical nature of the data, due to the 7-day week, as the first day of the dataset is a Sunday, meaning day six and day seven are Friday and Saturday. Therefore, we include both days to show the effect on a day during the work week and a day during the weekend.

The usage of defined days as points of splitting is based on the simplicity of implementation due to the file structure of the dataset, where each day has its own file containing all interactions. This is backed up by the graph of interactions per time during the day, shown in Figure 5.2. The Figure shows a deep dip in interactions between midnight and 03:00, making the valley a good point to split the days.

5.2.3 Evaluation

As described in subsection 5.1.2, the models are to be evaluated on the ability to classify due to the models being implemented as a pseudo-classification task [60]. Therefore, AUC is used as the main metric, as it reflects the overall ranking performance of a classifier [24]. In addition to AUC, the mAP@k metric will provide further information on the models from the secondary perspective of information retrieval. Nevertheless, AUC will be of primary concern. The mAP is calculated at 10, 20, 50, 100, 500, 1000, and 10000. mAP is primarily interesting at lower values as it represents the ability of the recommender to provide accurate recommendations within the first k recommendations. Therefore, the mAP@1000 and mAP@10000 are largely irrelevant to the underlying meaning. However, they are included to provide perspective on the value to which the mAP score converges.

5.2.4 Hyperparameter optimization

To ensure optimized performance, hyperparameter optimization was performed. The optimization was done using a Bayesian hyperparameter search. The optimization is performed by a Gaussian process which learns a function of the hyperparameters to the validation loss [52]. The process estimates an expected validation loss and its standard deviation for each permutation of hyperparameters. The process then selects the permutation with the lowest value for $E(\text{val loss}) - std(\text{val loss})$, and builds a model based on the hyperparameters and tests the model [8]. The process then updates its learned function with the validation loss incurred for the hyperparameters and generates new predictions based on its new information. This is performed iteratively until stopped. The optimization is performed on the fully extended model without ablation. The full dataset is too large for the hardware available to perform efficient optimization. Therefore, the interactions from day one were used as training data, and interactions from the second day as validation data. The optimization of the model ran for a total of seven days of runtime, distributed among seven cooperating hyperparameter optimization processes. The optimization was set to a maximum of 50 epochs as this was the maximum amount of epochs in training of the full model, while keeping the training time within a reasonable time limit (60 hours).

The score combiner was implemented with several different operations, shown in Table 5.3, and was tested during hyperparameter optimization. The base method was tested with all the operations, and found that the neural network had the best performance, although incurring a minor time cost. The inner product operation had notably poor performance. Although the dot product performs best in the similarity operation of the click predictor [44, 60], the score combiner is an operation on scores that do not necessarily have high similarity. A candidate news may

Operation	Description
Inner Product	The simple product of the scores: $s_1 \cdot s_2 \cdot s_3$
Sum	The sum of the scores: $s_1 + s_2 + s_3$
Max	The maximum of the scores: $\max(s_1, s_2, s_3)$
Average	The average of the scores: $\frac{s_1+s_2+s_3}{3}$
Neural	A neural network with one hidden layer with $2 \cdot \text{count}(scores)$ nodes.

Table 5.3: The implemented operations of the score combiner, and descriptions of the operations.

be in a location of interest for the user, but with content that does not match the user's interests. The sum and average operations are also dependent on a high score in all probabilities and showed good performance on the highest predicted values, but struggled with the less, but still relevant, articles. The max operation has the opposite problem, where it overemphasizes the score of the most impacted and disregards the information from the other sections of the model. The neural network is able to learn a non-linear representation and can consider the impact of a single high score and several medium scores. Due to the outperformance of the neural method, the method was implemented with the neural score combiner. The hyperparameters and the resulting optimized values are shown in Table 5.4.

5.2.5 Further Experimental Settings

The batch size was increased by a factor of 16 from the batch size used during hyperparameter optimization to utilize the hardware available and thereby reduce the training time. This was necessary to adapt to the dataset size increasing by a factor of 38. Therefore the training batch size was increased from 4 096 to 65 536. Following [33], when the batch size is increased by a factor k, the initial learning rate should be increased by $\sqrt{k}$. The optimizer Adam [32] was used as the optimization algorithm, following the original implementation [60].

5.3 Experimental Results

This section presents the evaluations of the models on the test set. The test set, as described in Section 5.2.2, consists of the two last days of the dataset. The section will briefly present the different configurations of the model, before the results are presented. The results and their implications will be discussed in subsection 5.3.2. As detailed in subsection 5.2.3, the main focus will be on the performance as measured by the AUC metric.

Hyperparameter	Possible values	Optimized value
Learning rate	range between [5.5E-4, 1.3E-6]	2.2E-6
Attention hidden dimensions	Integers in [20-100]	88
Category embedding dimensions	Integers in [5-20]	8
Convolution activation function	selu, relu, sigmoid, tanh	selu
Convolution window size	Integers in [3,10]	7
Convolution filters	Integers in [5-20]	6
Dense Activation Function	selu, relu, sigmoid, tanh	tanh
Dropout Probability	[5%-30%]	15.04%
Word embedder Type	Pre-trained or Not pre-trained.	Not pre-trained
Word Embedding Dimension	Integers in [15-300]	259
Score Combiner	Average, Sum, Max, Dot Product, Neural Network	Neural network
Click Predictor	Dot Product, Cosine	Cosine

Table 5.4: The hyperparameters which are optimized with respect to validation loss, their descriptions and the optimized values.

5.3.1 Results

The results of the experiment are shown in Table 5.6. These results are further broken down into the individual days for analysis, see Table 5.7 and Table 5.8 for Friday and Saturday, respectively. The models were trained and tested twice, and the average results are reported.

Models and Ablations

As described in subsection 5.1.1, the evaluation was performed as an ablation study, where we start from the full model and then remove data and modules iteratively. The data subject to ablation was the location data of the news in the click history, the current user position data and the historical user position data. These are denoted "loc", "pos" and "hist pos", respectively. The results are presented as additions to the base method, denoted NAML in the following tables. This is in line with the research goals to examine the effect of the user position and news location on recommendation quality. Therefore, the models in the results tables are denoted by the addition it makes to the base model (e.g., the "+ hist pos + loc" is the base model with the addition of the historical user position and news location, as well as the modules required to use the data, see Table 5.5 for a full description). Implementationally, the models are ablations of the extended model, rather than additions to the base model. It follows that the "NAML + hist pos + pos + loc" is the unablated model, and "NAML" is the model with all user position and news location information ablated. All models contain the news data, while news location data, user position data, and historical user position data can be added or removed. Furthermore, the ablation levels will consistently be referred to as *models* for simplicity. Although they are technically ablation levels and not standalone models, it is simpler to understand them as individual models that each uses a different amount of the input data. The models are further described in Table 5.5.

Performance Evaluation

Table 5.6 show the experiment results on the entire dataset. These results show that NAML achieves the best result, with the highest AUC score. The model using user position performs second-best. The third best model is the model which uses the click history's news location. The top three achieve similar performance, with the rest of the model performing substantially worse. The next four performers all score relatively similar and above the 50% mark, which is the expected score of a random classifier. It is interesting to observe that the lowest score is achieved by the full extension with all the data included.

Model	Description
NAML	The base NAML model without extension
+ loc	The base model with the news location of the click history
+ pos	The base model with the current user position
+ loc + pos	The base model with the news location of the click history and the current user position
+ hist pos	The base model with the historical user position
+ hist pos + loc	The base model with the historical user position and the news location of the click history
+ hist pos + pos	The base model with the current and historical user position
+ hist pos + pos + loc	The base model with the current and historical user position and the the news location of the click history

Table 5.5: The description of the models used in the ablation study.

The results for day 6 (Friday), is shown in Table 5.7. Again, the base NAML model performed the best on all metrics, tied with "+ hist pos + loc" on MAP@10 and MAP@20. "+ hist pos + loc" performs second best on AUC, and "loc" performs third best. The results for day 7 (Saturday), are shown in Table 5.8. Although the base NAML model outperforms on the mAP metrics, the "loc" model performs the best on the AUC metric. The second best is the base NAML model, with almost equal performance, and the "+ hist pos + loc" is close behind as the third best

For each of the three periods (both days separately as well as the days combined), the results on the AUC metric is generally split into two groupings. The high performers are "NAML", "+ loc" and "+ hist pos", which represent the base model, the extension which uses the news locations in the click history, and the NAML extension which uses the user's historical positions, respectively.

5.3.2 Discussion of the Results

The performance of the best models is very good, far outperforming a random guess (AUC = 50%). The results, shown in Figure 5.3, imply a negative effect on using the current user location (models which use "pos"). The effect of using either the news locations or the historical user position seem relatively neutral. However, the models combining several parts of the extension seem to affect the performance negatively.

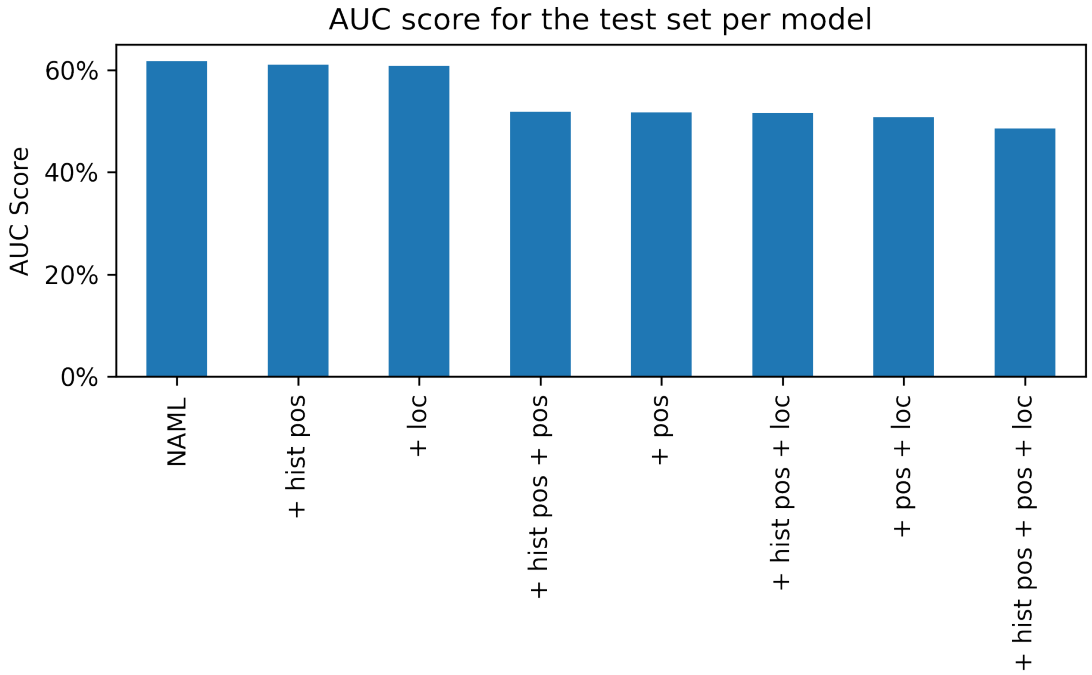


Figure 5.3: The performance of each model for the entire test set.

News and location seem to perform very similarly, which may indicate that the textual content (news title, body and categories) is processed in a sensible way by the models, and that the textual content is the primary driver of recommendation.

The negative effect on the models using the current user position may imply that the user position does not carry very meaningful information for news recommendation, but rather confuse and disrupt the base NAML model. This may be due to the model learning a way to incorporate the position that is beneficial for the training set, but not for the test set. Therefore, including the current position may not be beneficial in such a news recommender system. As shown in Figure 5.4, the news location seems to benefit recommendation in the case of the recommendation on Saturday (day 7) compared to Friday (day 6), while the other top performing models appear to do better on Friday than on Saturday. This motivates a discussion on the effect of the weekend, which will be done in Section 5.3.2.

Although the model which uses the historical user position ("+hist pos") performs well, the model using the current user position performs poorly, despite the models being based on the addition of a similar type of data. This can in part be explained by the averaging function of the spatial encoder: the encoding for historical positions of the user may become a description of the type of person the user is,

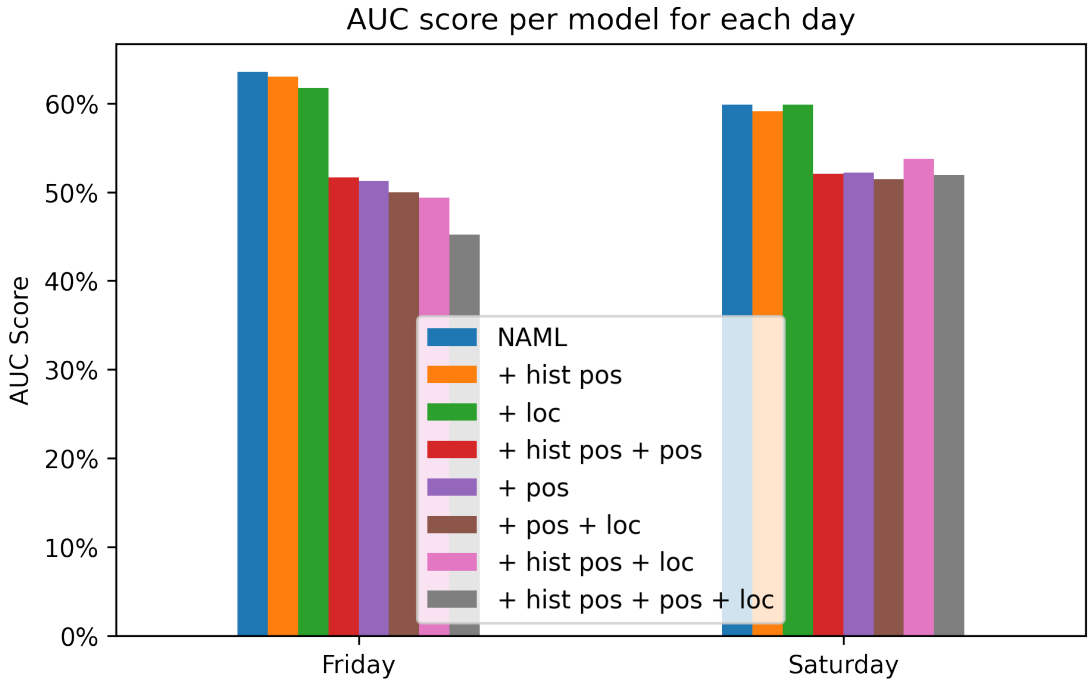


Figure 5.4: The performance of each model for the test set broken out into the individual days.

while the single current position becomes a type of contextual feature, and the context may not be accurately reflected by the position alone for the given data. Furthermore, the lack of outperformance of the historical position over the base NAML indicates that the modelling of the user is more effective on the history of clicked news articles.

As outlined in Section 4.2, the user position score and location interests scores are included in the score combiner, which is very close to the output node of the model, to let the effect be clear. However, this could lead to overfitting. The possibility of overfitting is further supported by the training loss shown in Table 5.9, where the base model has a higher loss, and the worse performing "+ pos" has a lower loss. Although several measures were taken in addressing the risk of overfitting, with regularization and hyperparameter optimization, a valid hypothesis based on the aforementioned evidence suggests the model has overfitted for the position data. This begs the question of why the location interests are not affected. This may be because the location data stems from the article contents and does not add any novel information to the system, and therefore the model does not lean into the location interests score as much as it does the positional score. The close correlation of the user history and the base NAML model may indicate that the

user location history is not considered very meaningful for news recommendation, and is thus not weighted heavily. It would support the initial hypothesis that the user's historical position does not contribute to a news recommender that already performs well. The model infers that the historical positions of the user are not necessary when the user preference is modelled well, and therefore, the cost of the invasion of the user's privacy may not be substantiated.

Locality of a local news paper

Adressa is the second-largest regional newspaper in Norway [34]. However, Table 5.10 shows that a very large part (32.6 %) of the user base accesses the news portal from the same city, Trondheim, and several of the top user positions are geographically close or even overlapping, such as the positions of "Heimdal" and "Tiller" are only 2.1 km apart. These positions are not different enough to necessitate different treatment of the user in serving news. Although the embedder should learn from the articles that these positions are very close, it is not reasonable to assume that the distinction between the positions incurs a large enough increase in recommendation performance to warrant the invasion of privacy stemming from such specific tracking of the user position. In addition to this, the locations tagged in the news articles, shown in Table 5.11, show that among the top news locations, 6 out of 11 refer to locations in the Trøndelag area, where several of the locations are simply different designations for the same geographic area ("Midt-Norge" and "Trøndelag"). Furthermore, tags such as Europa (Europe), Norge (Norway), and Sverige (Sweden), are far too broad to enable the serving of local news to users, as these locations are not local in nature. A user in Hungary reading a local Norwegian newspaper should not be recommended news about Europe simply because it is the most local tag to the user.

The case for using the position of a user to recommend local news is based on the assumption that news that concerns areas close to the user are relevant to that user. However, the Adressa dataset may be too local for this to be an effective strategy. It is reasonable to assume that readers of a specific local newspaper do not aim to read news specifically local to *their* position, but rather local to the area of the news portal they are using. Therefore, the assumption that underpins the use of user position information to recommend relevant news may not be valid for a local newspaper. Furthermore, a similar argument can also be made for a national news portal: a user of a national news portal, such as VG[1] may not want to see local news, as if their objective was to see local news, they would consult a local newspaper, as they would naturally have more to offer on news local to the area it covers. Therefore, the user may want *less*, and not more news local to the user's

[1] `www.vg.no`

position, which causes the user position to be of little value for recommending news items. An argument can be made about the position being relevant for filtering out local news, but this would require further research to examine.

The weekend effect

Focusing on the best performing models, "NAML", "+ loc" and "+ hist pos", we can see a clearly lower performance on Saturday compared to Friday in Figure 5.5. This is especially prevalent in the model using the historical user position. The model which models the location interests of the user is much more robust over the Friday-Saturday shift than the other two models. Furthermore, as seen in Figure 5.4, the performance of all models using the news location interests increased by 5.2% on Saturday relative to the performance on Friday.

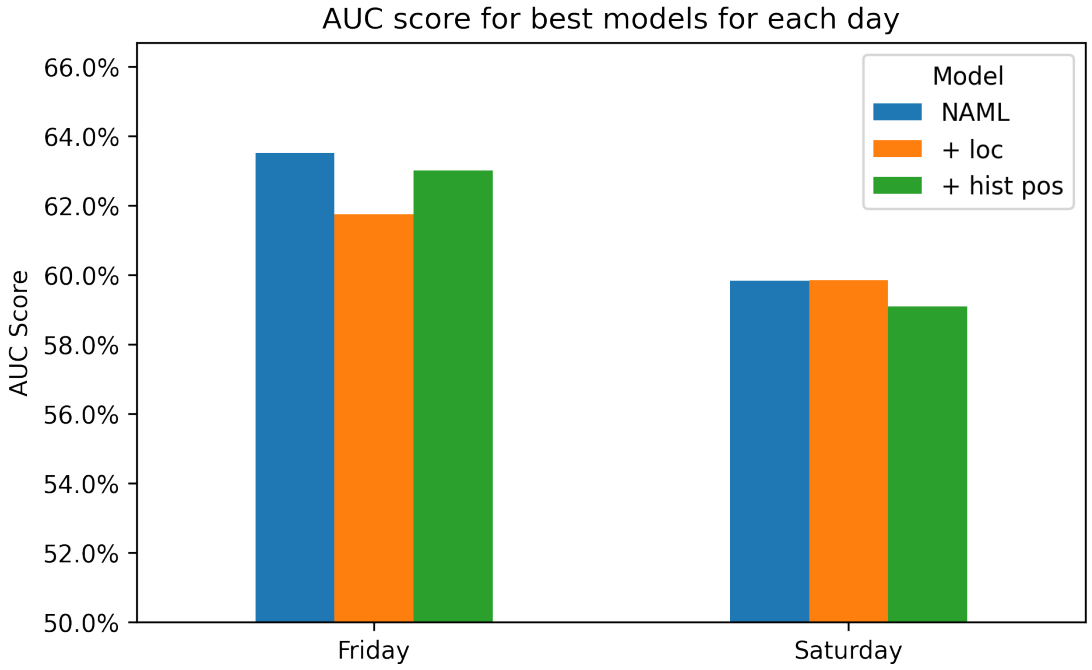


Figure 5.5: The performance of the Top-3 models for the test set broken out into the individual days.

One possible explanation for the poor performance of the historical user position is that the positions of the user is much more volatile on the weekends. During the week, most people spend the majority of their time at home or at work, while during the weekends there is more free time to go on trips to other locations, such as to a cabin or to events.

Furthermore, the news reading habits of a user on the weekend may differ from those of a user during the week. The increased amount of free time could conceivably cause a shift in reading habits and preferences. This may be the weekend effect that causes the models to underperform on the weekend relative to the weekdays. This effect could be learned by the model over many weeks, but the base model (NAML) does not incorporate the days or time of the interactions in the recommendation process, and as such, the model will not be able to simply distinguish the periods apart. Therefore, the models have no foundation to know there is an effect or how to distinguish between the time periods.

As the model is trained, the model learns in the "score combiner" how to weight the scores from the news, the user position and the news locations, see Figure 4.3. Therefore, if there is a shift in user behaviour or context which makes one of the modules uninformative or counterproductive on certain days, the model has already learned its weights of the scores, and may overweight sources that are faulty as a cause of the behaviour or context shift.

It is harder to justify that a large shift should occur in the location interests of a user due to the weekday/weekend shift, as location interests are less dependent on such a context. This may cause the seemingly more robust performance from Friday to Saturday than that of the other two high-performing models. Therefore, the news location interests of a user may be a useful component to add robustness to a model when the weekend comes or other such shifts in user preference.

Statistical significance test

To investigate the validity of our results, we complete a statistical significance test. We do this by building a linear additive effects model with Gaussian noise, which is formulated by the regression model

$$AUC = \beta_0 + \beta_{day} \cdot i_{day} + \beta_{loc} \cdot i_{loc} + \beta_{\text{hist pos}} \cdot i_{\text{hist pos}} + \beta_{pos} \cdot i_{pos} + \text{noise} \quad (5.1)$$

Here, β_0 is the effect of the base model on AUC, i_{data} is the binary index representing whether the specific data is used, β_{data} is the unknown effect of data on AUC, and the noise is a Gaussian distribution with zero mean and unknown variance. The effect of the base model on the AUC is a constant as we have no models which do not use the base model NAML.

As seen in Equation 5.1, the model assumes additivity of effects, meaning the AUC can be explained by the addition of the individual effects of the separate data. While this model, therefore, is quite simplistic, it can still shed some light on the effects of including each data source.

Table 5.12 shows the calculated values for β_0, β_{loc}, $\beta_{histpos}$ and β_{pos}, and their respective p-value. The selected model ensures that a statistical t-test can be used to determine significance. Here we tested the hypothesis $H_0 : \beta_{data} \geq 0$ against the alternative $H_A : \beta_{data} < 0$. Although the estimated effects of loc, pos and hist-pos were all negative, only the effects of pos were statistically significant (p <1E-5). We can assert with statistical significance that the effect of including the current user position in the model is negative for the performance on the AUC metric. To conclude on the other data sources, more iterations of the model must be done. This is left for further work.

Model	AUC	MAP@10	MAP@20	MAP@50	MAP@100	MAP@500	MAP@1'000	mAP@10'000
NAML	**61.67%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**99.83%**	**98.28%**	**81.66%**
+ loc	60.80%	76.58%	80.31%	82.02%	81.65%	80.92%	80.22%	72.44%
+ pos	51.71%	51.91%	49.96%	49.67%	49.14%	48.80%	49.56%	48.72%
+ pos + loc	50.72%	55.52%	54.52%	47.36%	43.74%	42.08%	42.41%	46.97%
+ hist pos	61.06%	66.79%	66.04%	63.28%	62.59%	63.65%	66.50%	71.35%
+ hist pos + loc	51.57%	**100.00%**	99.27%	93.81%	85.70%	68.31%	63.30%	55.65%
+ hist pos + pos	51.86%	82.24%	83.20%	81.92%	80.27%	72.98%	70.33%	59.96%
+ hist pos + pos + loc	48.55%	94.75%	93.44%	87.92%	82.10%	67.16%	62.13%	51.94%

Table 5.6: Test set results on the metrics AUC and mAP. mAP is measured over several values of k.

Model	AUC	MAP@10	MAP@20	MAP@50	MAP@100	MAP@500	MAP@1'000	mAP@10'000
NAML	**63.52%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**99.81%**	**98.15%**	**81.63%**
+ loc	61.75%	71.51%	78.22%	82.52%	83.16%	82.52%	81.26%	72.77%
+ pos	51.24%	62.86%	59.53%	59.21%	57.97%	54.14%	53.87%	48.21%
+ pos + loc	49.99%	74.61%	67.74%	53.42%	46.65%	39.38%	39.08%	45.87%
+ hist pos	63.01%	74.44%	70.33%	66.57%	66.06%	64.29%	66.30%	72.68%
+ hist pos + loc	49.38%	**100.00%**	**100.00%**	99.09%	92.30%	72.71%	67.19%	53.32%
+ hist pos + pos	51.64%	94.78%	92.10%	87.67%	84.55%	76.63%	74.55%	62.56%
+ hist pos + pos + loc	45.18%	90.60%	91.14%	83.28%	75.53%	59.82%	55.00%	47.52%

Table 5.7: Results from day 6 (Friday) on the metrics AUC and mAP.

Model	AUC	MAP@10	MAP@20	MAP@50	MAP@100	MAP@500	MAP@1'000	mAP@10'000
NAML	59.83%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**99.85%**	**98.40%**	**81.69%**
+ loc	**59.85%**	81.65%	82.41%	81.52%	80.14%	79.32%	79.18%	72.10%
+ pos	52.17%	40.97%	40.39%	40.13%	40.31%	43.45%	45.25%	49.22%
+ pos + loc	51.46%	36.44%	41.29%	41.31%	40.82%	44.77%	45.74%	48.07%
+ hist pos	59.10%	59.13%	61.75%	59.98%	59.13%	63.01%	66.70%	70.01%
+ hist pos + loc	53.76%	**100.00%**	98.53%	88.53%	79.09%	63.91%	59.41%	57.98%
+ hist pos + pos	52.08%	69.69%	74.30%	76.16%	75.99%	69.33%	66.10%	57.36%
+ hist pos + pos + loc	51.92%	98.89%	95.74%	92.55%	88.67%	74.51%	69.26%	56.36%

Table 5.8: Results from day 7 (Saturday) on the metrics AUC and mAP.

Model	Training Loss
NAML	0.7311
+ loc	0.6759
+ pos	0.6759
+ loc + pos	0.6896
+ hist pos	0.6813
+ hist pos + loc	0.6871
+ hist pos + pos	0.6826
+ hist pos + pos + loc	0.6905

Table 5.9: The training loss of the models used in the ablation study.

Position	Count	Proportion
Trondheim	9249390	32.55%
Oslo	5162199	18.17%
Norge	2157390	7.59%
Akershus	631729	2.22%
Tiller	599759	2.11%
Ranheim	526649	1.85%
Melhus	517980	1.82%
Heimdal	453220	1.60%
Steinkjer	388836	1.37%
Levanger	291119	1.02%

Table 5.10: The most occurring user positions and the proportion of total interactions out of 941 distinct positions in the dataset of 28 413 829 interactions.

Location	Count	Proportion
Trondheim	21715.0	12.32%
Norge	13416.0	7.61%
Oslo	7211.0	4.09%
Trøndelag	6781.0	3.85%
Sør-Trøndelag	6658.0	3.78%
Nord-Trøndelag	3015.0	1.71%
Europa	2607.0	1.48%
Sverige	2348.0	1.33%
Midt-Norge	2320.0	1.32%
Bergen	2248.0	1.28%
Stjørdal	1822.0	1.03%

Table 5.11: The most occurring tagged locations (proportion $> 1\%$) and the proportion of the total amount of locations out of 5535 distinct locations tagged in the dataset of 73 309 articles.

Part	Effect	p-value
NAML	62.10%	
loc	-3.12%	6.9E-2
hist pos	-2.42%	1.2E-1
pos	-8.79%	**9.3E-6**

Table 5.12: The effect of the data and their p-value.

Chapter 6

Evaluation and Conclusion

This chapter contains the points of discussion about the thesis and its approach, and the final remarks of our thesis. Section 6.1 will provide a discussion on several topics on the merits and limitations of our work. The contributions made to the field are detailed in Section 6.2. In Section 6.4 we present some proposals for future work related to our thesis. Finally, Section 6.3 draws conclusions on the work.

6.1 Discussion

This section will discuss several aspects of the thesis and the approach. This section continues on the discussion started in subsection 5.3.2, but will discuss the approaches of the thesis rather than the results themselves to shed light on some underlying factors of the results.

6.1.1 News Location Extraction

The news location information in the dataset is automatically extracted with Named Entity Recognition (NER) methods, which extract entity names into specified categories. Therefore, the location information does not provide any additional data on the contents of the news article, but instead provides a highlighting of specific parts of the news texts. The highlighting provides the additional information on which words are locations, but as the NER extracts words from the texts, the locations are not new information. Exceptions to this are locations appearing later than the text's max size. The max title size is set to 10, the max body size is set to 50, and any words beyond the maximum are not included in the dataset. The locations appearing later than the text's max size are then additional data to the model. It is interesting that the models using this data perform comparatively to

the position models, which have access to additional data. It may be supportive of the hypothesis of the thesis that the highlighting of specific mostly-known data is as beneficial to the model as the addition of position data.

Furthermore, the NER method is a source of error in the dataset. As described in Zhang and Liu [68], the extracted locations may be ambiguous or unrelated to the location of the news, such as the Norse god of *Frøya*, and the afterlife concept of *Hell* are also the name of a location in Trønderlag. In addition to this, *Hell* means *Luck* in Norwegian. Therefore, the NER method may classify these poorly, as some locations map to several entities. This is a source of error for the news locations.

6.1.2 Order agnosticity of location encoder

The location encoder is implemented as order agnostic, meaning that it does not differentiate between the first and last element of the location set it receives. The user history is used as an indication of interest, which Gulla et al. [21] assumes fades over time unless renewed. However, in this thesis, an assumption is made that news location interests are more robust than topic interests. Furthermore, the experiment used a history size of 10, which is small enough to assume little fading between the first and the tenth read article.

6.1.3 The limitations of location interests

Although a strong historical interest in specific locations may indicate a corresponding interest in the same location in the future, the news recommendation may not understand the underlying factors for the interests, when the recommendations look at the location relevance and topic relevance separately. As an example of this consider a Norwegian fan of English football. Despite his interest in their sports, he does not care about their politics. However, as a Norwegian citizen, he is interested in Norwegian politics. This example shows that contextual factors are important. The method of the extended NAML model combines the location score and the news score and does not consider the underlying factors of the scores. Therefore, the model may infer that as the user reads many articles on English football and Norwegian politics, an article about English politics will score high on location and high on news content. Therefore, as the current implementation calculates the individual scores first, and then combines these scores without the underlying context, the method cannot reflect the context of the interests. This is similar to the XOR problem, but in this case, the problem is the computation of the individual scores before the combination of the scores. This is also the case for the user position. Noh et al. [43] showed that a user's news interests were very

dependent on their position, however, this was based on user positions which were more specific than what is available in the Adressa dataset, down to the degree of an office vs the Food court vs at Home. Chen et al. [12] explained that obtaining user position to such a high granularity is very difficult, sometimes impossible, in a real-world context. Nevertheless, the interest-position dependency shown by Noh et al. [43] may still be valid, and the same effects as those observed in the example of the American Football fan are still present and logical. A combination that can combine the per-topic relevancy and the per-location/per-position relevancy, such as a neural network, may prove to solve this limitation. It is left for future work to examine if the findings hold for a combination over the per-topic relevancy.

6.1.4 The effect of movement

The current user position models will struggle to recommend relevant news articles to a user at a new location, especially if the movement since the last interactions is large. Although a user *might* want to read about the place where they are, the news portal may not be able to supply it. Adressa will for instance probably not have any positionally relevant news to offer if a user is on Mallorca.

The historical user position will quickly adapt to the travel as the click history fills, and will end up in the same position as the current user position models. However, until that is achieved, the models will recommend news recommendations to where they were previously, before gradually transitioning. In the experiments of this thesis, the history size was set to a maximum size of 10 articles. The consequence of this is that when a user browses ten articles, the user history has been reset. Therefore, the model will not distinguish between a user who browses ten articles quickly in a visited position and a user who lives there. An example of this is a user which works in the city but lives in the countryside, where the user browses news at his job. They would quickly read enough articles to fill the click history, and the model will be unable to distinguish the user by its previous positions. Therefore, the historical user positions with a relatively small history size are essentially the same as the user position, but with a smoothing effect between positions.

The news location based models attempt to model the user's location interests. A user who is set to travel is naturally more interested in the place they are about to travel to, and this will then be reflected in the news they read to such a degree as the news portal is able to present news about the area. A location that the news portal does not cover will not be able to model the real location interest in these locations, however as the news portal does not cover the location, the lack of reflection on the location interests does not affect the quality of the recommendation.

Although these descriptions seem based on the infrequent case of travel, the underlying assumption of models which use the user position (current or historical) is that the user will change their position and will want news histories about the area around them.

6.1.5 Size of history

The size of the history in the experiment was limited to a maximum of 10. This was mainly due to the hardware limitations and the size of the dataset. The training set spanning 5 days was 1.6 million interactions. Combined with the encoding of the articles and context data, the training dataset was of size $1.6 \cdot 10^6 \times (10+1) \times 73$. Attempts were made with a history size of 15, but this adds another 2GB data and became a too large slowdown to accept. Furthermore, the history size of 15 is also very small in the context of the news reading history and may only cover a single browsing session. The effect of this is that the model will have less data from which to infer preference.

6.1.6 Implementation of baseline method as binary classification task

The baseline method, NAML, is originally implemented as a $K + 1$ classification task [60]. However, in this thesis, it is implemented as a binary classification task. This is due to the difference in datasets. The original implementation uses a dataset that supplied information on which articles are presented to the user, but not read. These can be used as negative samples outright. Furthermore, in this case, it is reasonable to implement the task as classification and to use classification and ranking metrics, as they have several candidates and a single positive. To stay consistent with the original implementation of NAML, the task is defined as classification. However, the dataset used in this thesis does not contain information on the seen-but-not-read articles. Therefore, negative sampling is performed to get negative candidates. The assumption made in this sampling, that a user is more interested in the article read than a popularity-biased randomly sampled article, is more problematic than the assumption made by the original implementation, that article read by the user is more relevant to the user than an article the user was shown, but not read. Therefore, the implementation was set to a binary classification task to not overemphasize a possibly problematic assumption. The evaluation focused on classification metrics, although if the task was not structured as a slate-presentation to the user, classical recommender system metrics could be beneficial.

6.1.7 Target Values

In this experiment, the target value of relevance is determined by whether the article was clicked by the user. Although this is a commonly used implicit feedback in news recommenders, the Adressa dataset contains information about other indicators as well, such as the time spent on the article. This implicit feedback may be a more informative source of information on relevance and confidence of the relevance, such as a very high read time should indicate a higher relevance than an interaction over a short time. In addition to this, a very short read time (e.g., $t < 10s$) could be a good indicator of a non-relevant article, as the user has enough time to know the subject of the article and then decide against reading it. Furthermore, the continuous axis of time can facilitate the possibility of modelling relevancy as a continuous spectrum, such as relevancy is in the real-world. An example of this is two articles about football, which are both relevant to the user, but one is about their favorite team. Thus we have two articles, where one article is more relevant than the other. To the best of our knowledge, no state of the art in neural news recommendation employs the read time as a target. Gulla et al. [21] uses the read time to filter out "insignificant timed act", which are defined in the article as read times below 4.2s. Although such an implementation could be beneficial for the performance of a news recommender, this falls outside the scope of our thesis, and it is left for further studies.

6.1.8 Test result variation

As described in subsection 5.3.1, the results presented is the average of two iterations of the models. Some variation in the results between the two iterations was observed. The execution time of training one ablation of the model ranged between 36-60 hours each. The long execution time and a lack of available compute hindered the possibility of doing more iterations of the models. Available resources to allow several iterations of the models would enable the completion of statistical tests on the significance of our results. Therefore, a point of future work is to repeat the study over several more iterations to confirm the validity and significance of the results.

6.2 Contributions

This thesis provides several contributions to the field. The contributions which are specifically linked to the research questions of this thesis will be presented in the following chapter. In addition to these, we have made several contributions that are not tied to the research questions which we will present in this section.

A contribution to the field is a preprocessing routine for the Adressa Dataset. The dataset is messy and requires preprocessing. This contribution is beneficial for research within the same field as this thesis, as Adressa is to date the only news dataset with user position and news location data. The preprocessing routine handles the interactions and articles of the dataset, and works for both the larger 20M dataset and the 2M dataset.

The spatial extension to the NAML model is a contribution that enables the use of spatial data in the recommendation. The extension is a module that enables the examination we completed in this thesis. Furthermore, the extension is independent of the underlying model and can be applied to any other news recommender which predicts a click probability. Therefore, the module can be reused on several other models to view the performance effects of user position and news location data.

Several custom TensorFlow layers are designed and built for this thesis, which have general value. The *Masked Mean Pooling* layer has a large use case and provides functionality that is not natively available through the basic TensorFlow package. The *Attention* Layer which is used in the original implementation of NAML does not function properly with the current version of TensorFlow (2.0). A contribution is therefore a custom layer that provides the same functionality as that of the original implementation but ported to TensorFlow 2.0. This lowers the implementation difficulty of several State of the Art neural news recommender methods which use this module, such as NRMS [62], NAML [60] and LSTUR [5]. The slicer is a custom layer that takes in a Tensor, slices out a defined section of the Tensor, and passes it on. In this project, we have used this layer to divide the input into the defined chunks for each *view*. However, this has general-purpose utility.

The results of the thesis demonstrate the validity of a privacy-preserving, but high-performing news recommender. We have shown that the data on the current user position lowers the performance of the recommender, and have shown that the results are significant (p <1E-5) by a statistical test.

6.3 Conclusion

In this thesis we have looked at how the news location interests of a user can be modelled, and whether it can mitigate the need for tracking the user's position. The state of the art in news recommenders has been examined, and a state of the art method has been extended to enable usage of user position data and news location data. The extended model has been tested in an experiment structured

as an ablation study.

For clarity, we restate the goal and research questions of the thesis:

Goal *The main goal of this thesis is to investigate to which degree one can create a news recommender system which uses textual content to build a user profile containing enough information on the users' location interests to mitigate the need to track the users position without a large loss in performance.*

Main Question *How can a news recommender use the inferred location interests of users to mitigate the need of using users' position?*

RQ 1: *What is the current state of the art in location-aware news recommendation?*

RQ 2: *How can user position and news location be modelled for news recommendation?*

RQ 3: *Is the user's position a necessary component for achieving high performance news recommendation?*

RQ1 was answered in Chapter 3. We found that the state of the art in location-aware news recommendation is broad and varied in its application of the user position and news location. However, the implementations of user position and news location is often complex and abstract from the view of the output. Therefore, we selected a state of the art personalized news recommender to implement and extend with user position and news location modules, to clearly examine the effect of the spatial data.

RQ2 was answered in Chapter 3 and Chapter 4 combined. In Chapter 3, we examined how the user position and news location have been modelled in the state of the art. In Chapter 4, we detailed the technical approach of extending the state of the Art personalized news recommender system with a spatial encoder to utilize user position and news location.

RQ3 was answered by the ablation study and its results. In the parameters of this thesis, we found that the user position is *not* a necessary component for achieving high-performance news recommendation. We found that a State of the Art recommender on the dataset performs better without the user position extension.

These findings and answers to our research questions combine to answer the Main Question of the thesis. The answer to our main question is then that the news recommender can create a user profile through the content of the article which is accurate enough to mitigate the need of using the user position. Furthermore, answering the Main Question has ensured that we achieved the goal of this thesis.

Although the results show the location interests have little improvement of the recommendation performance over the base state of the art recommender, the results are affected by the several factors discussed in Chapter 6. We believe that modelling the user's location interests would benefit the recommender performance, given a newspaper of larger user reach and news span. Furthermore, the user position assumption is that a user is interested in news occurring spatially close to the user, which requires the path of user position → news location interest → relevant news. The modelling of user location interests requires the path of click history → news location interest → relevant news. In this way, rather than relying on the user's positional data, the user can prove their interests through their click history. Hence, the assumption of "a user is interested in news which is about a location close to the user" is converted to a much more reasonable assumption of "a user which has previously been interested in news about a location, is interested in news about the same location". The last assumption is very similar to the general assumptions made by most news recommenders: "a user which has previously been interested in news about a topic, is interested in news about the same topic". In addition, this would be a non-intrusive approach to modelling the user's location interests, and would encapsulate a possible performance enhancement from user position modelling, without the need to intrude on the user's privacy. Therefore, the validity of a hypothesis that the location interests should perform comparatively in performance while preserving the privacy of the user, is valid, despite the lack of outperformance relative to the base news recommender.

In conclusion, the work in this thesis has demonstrated that a news recommender system that uses textual content can build a user profile containing enough information on the users' location interests to mitigate the need to track the users' position.

6.4 Future Work

To continue this research, we propose a few suggestions for further work. These suggestions are tied to improvements or alterations of the work done in this thesis.

6.4.1 News Location dataset

As described in subsection 6.1.1, the news locations algorithmically extracted from the news articles are a source for error. For future work within the impact of news locations in recommendation, a more robust and precise method for generating the news locations in the dataset should be developed. This could be done by the journalists as they create the news article, by outlining the relevant news location(s) of the article.

Furthermore, as discussed in subsection 5.3.2, the news location dataset of Adressa may be too local. The Adressa dataset is based on interactions on the Adresseavisen News portal[1], which is a regional newspaper. Therefore, both the user position and the news locations are heavily biased toward locations within the Trønderlag County area. Conducting similar research on a dataset from a less localized news source could prove informative to the validity of the findings for larger and more widespread news portals. Therefore, it would be beneficial for future work on the subject if a dataset was created by a national news portal, or by a larger media organization, such as Polaris Media, which runs 64 regional media houses, by combining user location interests across news portals.

6.4.2 Encoding of location interests based on distance measure

An interesting topic for further research is the implementation of the location encoder with a basis in the physical distance between locations. This could be implemented as a sort of topography map where the "height" of a location is the user's historical interest in a location, which would allow a sensible smoothing, so as to reflect that an interest in a particular location may indicate an interest in neighboring locations. Due to the state of the current open datasets, this would require an integration with a knowledge base, such as WikiData[2], to derive the physical locations from the textual names of the locations.

6.4.3 Combining the news and location preferences to provide context

subsection 6.1.3 describes the XOR-problem limitation of the score combiner which occurs because it receives only the score and not the topics, and can therefore not reason on the preferences in relation to positions and locations. A point of further work would be to implement the score combiner to take in the distribution of the scores over topics and locations, and to combine them in a manner that allows the model to reason over topics and positions individually. A possible method of this would be to concatenate the distributions and use a neural net to combine the inputs and to let the model learn the connections between specific locations and topics.

[1] `www.adressa.no`
[2] `www.wikidata.org`

Bibliography

[1] F. Abel, Q. Gao, G.-J. Houben, and K. Tao. Analyzing user modeling on twitter for personalized news recommendations. In *Proceedings of the 19th International Conference on User Modeling, Adaption, and Personalization*, UMAP'11, page 1–12, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 9783642223617.

[2] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005. doi: 10.1109/TKDE.2005.99.

[3] C. C. Aggarwal. *Recommender Systems: The Textbook*. Springer, 2016.

[4] X. Amatriain, J. M. Pujol, and N. Oliver. I like it... i like it not: Evaluating user ratings noise in recommender systems. In G.-J. Houben, G. McCalla, F. Pianesi, and M. Zancanaro, editors, *User Modeling, Adaptation, and Personalization*, pages 247–258, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-02247-0.

[5] M. An, F. Wu, C. Wu, K. Zhang, Z. Liu, and X. Xie. Neural news recommendation with long- and short-term user representations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 336–345, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1033. URL `https://aclanthology.org/P19-1033`.

[6] J. Bao, M. F. Mokbel, and C.-Y. Chow. Geofeed: A location aware news feed system. In *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*, ICDE '12, page 54–65, USA, 2012. IEEE Computer Society. ISBN 9780769547473. doi: 10.1109/ICDE.2012.97. URL `https://doi.org/10.1109/ICDE.2012.97`.

[7] J. Bennett and S. Lanning. The netflix prize. 2007.

[8] L. Biewald. Experiment tracking with weights and biases, 2020. URL `https://www.wandb.com/`. Software available from wandb.com.

[9] D. M. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4): 77–84, 2012.

[10] H. L. Borges and A. C. Lorena. A survey on recommender systems for news data. In *Smart Information and Knowledge Management*, pages 129–151. Springer, 2010.

[11] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. *arXiv preprint arXiv:1301.7363*, 2013.

[12] C. Chen, X. Meng, Z. Xu, and T. Lukasiewicz. Location-aware personalized news recommendation with deep semantic analysis. *IEEE Access*, 5:1624–1638, 2017.

[13] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches, 2014. URL `https://arxiv.org/abs/1409.1259`.

[14] G. de Souza Pereira Moreira. Chameleon: a deep learning meta-architecture for news recommender systems. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 578–583, 2018.

[15] G. de Souza Pereira Moreira, F. Ferreira, and A. M. da Cunha. News session-based recommendations using deep neural networks. In *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*, DLRS 2018, page 15–23, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450366175. doi: 10.1145/3270323.3270328. URL `https://doi.org/10.1145/3270323.3270328`.

[16] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer. The yahoo! music dataset and kdd-cup'11. In G. Dror, Y. Koren, and M. Weimer, editors, *Proceedings of KDD Cup 2011*, volume 18 of *Proceedings of Machine Learning Research*, pages 3–18. PMLR, 21 Aug 2012. URL `https://proceedings.mlr.press/v18/dror12a.html`.

[17] P. M. Gabriel de Souza, D. Jannach, and A. M. Da Cunha. Contextual hybrid session-based news recommendation with recurrent neural networks. *IEEE Access*, 7:169185–169203, 2019.

[18] E. Gabrilovich and S. Markovitch. Wikipedia-based semantic interpretation for natural language processing. *Journal of Artificial Intelligence Research*, 34:443–498, 2009.

[19] Y. Goldberg. Neural network methods for natural language processing. *Synthesis lectures on human language technologies*, 10(1):1–309, 2017.

[20] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[21] J. A. Gulla, A. D. Fidjestøl, X. Su, and H. N. C. Martínez. Implicit user profiling in news recommender systems. In *WEBIST (1)*, pages 185–192, 2014.

[22] J. A. Gulla, L. Zhang, P. Liu, Ö. Özgöbek, and X. Su. The adressa dataset for news recommendation. In *Proceedings of the international conference on web intelligence*, pages 1042–1048, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349512. doi: 10.1145/3106426.3109436. URL `https://doi.org/10.1145/3106426.3109436`.

[23] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4), dec 2015. ISSN 2160-6455. doi: 10.1145/2827872. URL `https://doi.org/10.1145/2827872`.

[24] M. Hossin and M. N. Sulaiman. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2):1, 2015.

[25] S. Høst. Avisåret 2020. papiraviser og betalte nettaviser. 2021.

[26] L. Hu, S. Xu, C. Li, C. Yang, C. Shi, N. Duan, X. Xie, and M. Zhou. Graph neural news recommendation with unsupervised preference disentanglement. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4255–4264, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.392. URL `https://aclanthology.org/2020.acl-main.392`.

[27] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE international conference on data mining*, pages 263–272. Ieee, 2008.

[28] ixnay. Recurrent neural network unfold, 2017. URL `https://commons.wikimedia.org/wiki/File:Recurrent_neural_network_unfold.svg#file`.

[29] G. Jawaheer, M. Szomszor, and P. Kostkova. Comparison of implicit and explicit feedback from an online music recommendation service. In *proceedings of the 1st international workshop on information heterogeneity and fusion in recommender systems*, pages 47–51, 2010.

[30] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015. doi: 10.1126/science.aaa8415. URL `https://www.science.org/doi/abs/10.1126/science.aaa8415`.

[31] M. Karimi, D. Jannach, and M. Jugovac. News recommender systems–survey and roads ahead. *Information Processing & Management*, 54(6):1203–1227, 2018.

[32] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[33] A. Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.

[34] M. Landsforening. *Mediehus 21/2*. Lesertall. 2022. URL `https://www.mediebedriftene.no/tall-og-fakta/lesertall/`.

[35] L. Li, D. Wang, T. Li, D. Knox, and B. Padmanabhan. Scene: A scalable two-stage personalized news recommendation system. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, page 125–134, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450307574. doi: 10.1145/2009916.2009937. URL `https://doi.org/10.1145/2009916.2009937`.

[36] L. Li, L. Zheng, and T. Li. Logo: a long-short user interest integration in personalized news recommendation. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 317–320, 2011.

[37] L. Li, L. Zheng, F. Yang, and T. Li. Modeling and broadening temporal user interest in personalized news recommendation. *Expert Systems with Applications*, 41(7):3168–3177, 2014.

[38] J. Lian, F. Zhang, X. Xie, and G. Sun. Towards better representation learning for personalized news recommendation: a multi-channel deep fusion approach. In *IJCAI*, pages 3805–3811, 2018.

[39] E. Loper and S. Bird. Nltk: The natural language toolkit. *CoRR*, cs.CL/0205028, 2002. URL `http://dblp.uni-trier.de/db/journals/corr/corr0205.html#cs-CL-0205028`.

[40] R. Meyes, M. Lu, C. W. de Puiseau, and T. Meisen. Ablation studies in artificial neural networks. *arXiv preprint arXiv:1901.08644*, 2019.

[41] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space, 2013. URL `https://arxiv.org/abs/1301.3781`.

[42] T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997. ISBN 978-0-07-042807-2.

[43] Y. Noh, Y.-H. Oh, and S.-B. Park. A location-based personalized news recommendation. In *2014 International Conference on Big Data and Smart Computing (BIGCOMP)*, pages 99–104, 2014. doi: 10.1109/BIGCOMP.2014.6741416.

[44] S. Okura, Y. Tagami, S. Ono, and A. Tajima. Embedding-based news recommendation for millions of users. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1933–1942, 2017.

[45] M. Ovsjanikov and Y. Chen. Topic modeling for personalized recommendation of volatile items. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 483–498. Springer, 2010.

[46] D. Parra and X. Amatriain. Walk the talk. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 255–268. Springer, 2011.

[47] S. Raza and C. Ding. News recommender system: a review of recent progress, challenges, and opportunities. *Artificial Intelligence Review*, pages 1–52, 2021.

[48] S. Raza, S. R. Bashir, D. D. Liu, and U. Naseem. Balanced news neural network for a news recommender system. In *2021 International Conference on Data Mining Workshops (ICDMW)*, pages 65–74. IEEE, 2021.

[49] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.

[50] R. Rehurek and P. Sojka. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks*. Citeseer, 2010.

[51] S. J. Russell and P. Norvig. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[52] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

[53] M. Steyvers and T. Griffiths. Probabilistic topic models. *Handbook of latent semantic analysis*, 427(7):424–440, 2007.

[54] M. Tavakolifard, J. A. Gulla, K. C. Almeroth, J. E. Ingvaldesn, G. Nygreen, and E. Berg. Tailored news in the palm of your hand: A multi-perspective transparent approach to news recommendation. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13 Companion, page 305–308, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450320382. doi: 10.1145/2487788.2487930. URL `https://doi.org/10.1145/2487788.2487930`.

[55] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2017. URL `https://arxiv.org/abs/1706.03762`.

[56] H. Wang, F. Zhang, X. Xie, and M. Guo. Dkn: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 world wide web conference*, pages 1835–1844, 2018.

[57] J. Wang, L. Yu, W. Zhang, Y. Gong, Y. Xu, B. Wang, P. Zhang, and D. Zhang. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 515–524, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350228. doi: 10.1145/3077136.3080786. URL `https://doi.org/10.1145/3077136.3080786`.

[58] H. Wen, L. Yang, and D. Estrin. Leveraging post-click feedback for content recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 278–286, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362436. doi: 10.1145/3298689.3347037. URL `https://doi.org/10.1145/3298689.3347037`.

[59] C. Wu, F. Wu, J. Liu, S. Wu, Y. Huang, and X. Xie. Detecting tweets mentioning drug name and adverse drug reaction with hierarchical tweet representation and multi-head self-attention. In *Proceedings of the 2018 EMNLP Workshop SMM4H: The 3rd Social Media Mining for Health Applications Workshop & Shared Task*, pages 34–37, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5909. URL `https://aclanthology.org/W18-5909`.

[60] C. Wu, F. Wu, M. An, J. Huang, Y. Huang, and X. Xie. Neural news recommendation with attentive multi-view learning. *arXiv preprint arXiv:1907.05576*, 2019.

[61] C. Wu, F. Wu, M. An, J. Huang, Y. Huang, and X. Xie. Npa: neural news recommendation with personalized attention. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2576–2584, 2019.

[62] C. Wu, F. Wu, S. Ge, T. Qi, Y. Huang, and X. Xie. Neural news recommendation with multi-head self-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6389–6394, 2019.

[63] C. Wu, F. Wu, Y. Huang, and X. Xie. Personalized news recommendation: A survey. *arXiv preprint arXiv:2106.08934*, 2021.

[64] C. Wu, F. Wu, T. Qi, and Y. Huang. Empowering news recommendation with pre-trained language models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1652–1656, 2021.

[65] F. Wu, Y. Qiao, J.-H. Chen, C. Wu, T. Qi, J. Lian, D. Liu, X. Xie, J. Gao, W. Wu, and M. Zhou. MIND: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.331. URL `https://aclanthology.org/2020.acl-main.331`.

[66] W. Xu, C.-Y. Chow, M. L. Yiu, Q. Li, and C. K. Poon. Mobifeed: A location-aware news feed system for mobile users. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '12, page 538–541, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450316910. doi: 10.1145/2424321.2424409. URL `https://doi.org/10.1145/2424321.2424409`.

[67] S. Zhai, K.-h. Chang, R. Zhang, and Z. M. Zhang. Deepintent: Learning attentions for online advertising with recurrent neural networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 1295–1304, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939759. URL `https://doi.org/10.1145/2939672.2939759`.

[68] L. Zhang and P. Liu. Addressa dataset description. 2018. URL `https://reclab.idi.ntnu.no/dataset/cxdataset_v5.pdf`.

[69] L. Zhang, P. Liu, and J. A. Gulla. Dynamic attention-integrated neural network for session-based news recommendation. *Machine Learning*, 108(10): 1851–1875, 2019.

[70] P. Zhang and Z. liu. Gateformer: Speeding up news feed recommendation with input gated transformers, 2022. URL `https://arxiv.org/abs/2201.04406`.

[71] W. Zhang, T. Chen, J. Wang, and Y. Yu. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, page 785–788, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450320344. doi: 10.1145/2484028.2484126. URL `https://doi.org/10.1145/2484028.2484126`.

[72] G. Zheng, F. Zhang, Z. Zheng, Y. Xiang, N. J. Yuan, X. Xie, and Z. Li. Drn: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, page 167–176, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee. ISBN 9781450356398. doi: 10.1145/3178876.3185994. URL `https://doi.org/10.1145/3178876.3185994`.

Adrian Wærøe Langseth

Use of Spatial Information in News Recommenders

NTNU
Kunnskap for en bedre verden