Modeling Contextual Preferences of Hearing Aid Users

Korzepa, Maciej Jan

Publication date:
2020

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Modeling Contextual Preferences
of Hearing Aid Users

Maciej Korzepa

Kongens Lyngby 2020
Abstract

The current clinical approach for calibrating hearing aid (HA) settings to compensate for individual hearing loss employs amplification schemes which are established based on how an average hearing impaired listener perceives loudness or speech. This approach does not take into consideration the large variability across users nor the variety in auditory needs resulting from differences in listening environments and situations encountered throughout the day. While modern HAs provide a wide range of capabilities that can address these variations, the lack of resources in hearing health care prevents a successful personalization of the devices in clinical settings. A possible solution is to move the personalization process outside the clinics by employing algorithms to learn auditory preferences of HA users in real environments. This project explores how to model and learn the preferences of HA users dependent on the context.

Firstly, an in-the-wild user study was run to explore setting preferences of HA users in real listening environments. The results showed that HA users may have preferences for HA settings very different from the ones provided in standard clinical settings and that these preferences vary not only between but also within individual users depending on the context.

Secondly, a Gaussian process based approach is proposed to simulate contextual preferences of HA users in order to facilitate a systematic evaluation and agile development of HA personalization solutions. It incorporates aspects such as correlations of preferences across users, complexity of preferences, partial observability of context and user features inducing these preferences, and noisiness of perceptual user feedback. Simulation allows to circumvent the limitations of traditional user studies and enables a more scalable approach to run multiple large-scale experiments validating different aspects of preference learning solu-
tions, such as efficiency of learning in the presence of limited and noisy data, leveraging similarities across users or accounting for latent factors.

Thirdly, a novel probabilistic meta-learning method is proposed inspired by parallel research focusing on Bayesian neural networks and their connections with Gaussian processes. This meta-learning method is applied to the problem of contextual personalization of HA settings in a number of simulations with varying assumptions about the observability of user and context features. The results indicate potential both for using the probabilistic meta-learning framework to facilitate contextual personalization in the HA domain and for evaluating and comparing HA personalization solutions in simulated environments.

This project has provided a foundation for taking a more holistic approach to HA personalization by enabling development of scalable solutions that learn auditory preferences dependent on the context.
Den nuværende kliniske tilgang til kalibrering af høreapparat’s (HA) indstillinger, for at kompensere for individuelt høretab anvender et forstærkningsparadigme, der er baseret på hvordan en gennemsnitlig hørehemmet forstår tale i forhold til lydstyrke. Denne tilgang tager ikke højde for den store variabilitet på tværs af brugere eller variationen i auditive behov, som følge af forskelle i lyttemiljøer og situationer der opstår i løbet af dagen. Selvom moderne HA’er giver mange muligheder for finjustering i forhold til disse variationer, begrænser manglen på ressourcer i sundhedssektoren ofte i hvor høj grad det er muligt at personalisere enhederne i en høreklinik. En mulig løsning er at flytte personaliseringsprocessen uden for klinikkerne, ved at integrere algoritmer i smartphone apps der kan lære HA-brugeres audiologiske præferencer i virkelige lydmiljøer.

Dette PhD projekt fokuserer på hvordan man kan modellere og lære præferencer for HA indstillinger afhængigt af konteksten.

Som det første i projektet blev der gennemført en række kliniske eksperimenter med HA-brugere for at afprøve alternative HA indstillinger, for dermed at opsamle deres præferencer i ægte lyttemiljøer. Resultaterne viste, at HA-brugere kan have præferencer for HA-indstillinger der er meget forskellige fra dem, der typisk bruges som standard i en høreklinik, og at disse præferencer ikke kun varierer imellem brugere, men også individuelt afhænger af konteksten for den enkelte bruger.

Denæst blev der i projektet arbejdet med modellering af HA indstillinger, base-ret på Gaussiske processer for at simulere kontekstuelle præferencer for HA-brugere, med henblik på at forbedre systematisk evaluering og agil udvikling af HA-personaliseringsløsninger. Her indgår aspekter som korrelationer mellem
præferencer på tværs af brugere, kompleksitet af præferencer, i hvilken grad der er muligt at observere kontekst og underliggende baggrund for præferencer, samt den støj der generer i perceptuel brugerfeedback. Simulation giver mulighed for at omgå begrænsninger i traditionelle kliniske eksperimenter, for dermed at skalere og parallelt gennemføre flere undersøgelser. Dette kan bruges til at validate hvorvidt det er muligt at lære brugerpræferencer baseret på begrænset og støjfyldt data, ved at udnytte similaritet på tværs af brugere samt underliggende latente faktorer.


Dette projekt giver et fundament for en mere holistisk tilgang til HA-personalisering ved at muliggøre udvikling af skalerbare løsninger, der lærer auditive præferencer afhængigt af konteksten.
This thesis is presented in fulfillment of the requirements for acquiring a Ph.D. in Engineering, and was prepared at the Cognitive Systems section of DTU Compute, under the supervision of Professor Jan Larsen, Associate Professor Jakob Eg Larsen, Professor Morten Mørup and Industry Fellow Michael Kai Petersen.

The thesis deals with aspects of how modeling contextual preferences of hearing aid users can provide a more holistic and scalable approach to the personalization of hearing aids. The thesis includes 5 published papers, and 1 unpublished manuscript.

Lyngby, December 31st-2020

Maciej Korzepa
Scientific contributions

Peer-reviewed publications

A Maciej Jan Korzepa, Benjamin Johansen, Michael Kai Petersen, Jan Larsen, Jakob Eg Larsen, and Niels Henrik Pontoppidan. Learning preferences and soundscapes for augmented hearing. *Proceedings of Intelligent User Interfaces*, 2068, 2018. ISSN 16130073


Unpublished manuscripts


The above contributions are part of this thesis and are included in the corresponding appendices A to F.

Publications not included in the thesis


First and foremost, I would like to express my sincere gratitude to my external supervisor Michael Kai Petersen who opened the opportunity of this PhD project to me and encouraged me to start this journey. I would like to thank you for helping me navigate in the world of hearing aids, for giving me unlimited freedom to explore my research interests and most importantly for your endless patience and kindness, and believing in me at every stage of my PhD project. Without your tremendous support, I would not be able to finish this project.

I would also like to sincerely thank my university supervisors, Associate professor Jakob Eg Larsen and Professor Morten Mørup, who offered me enormous support that carried my through the last months of my project. Also, thank you, Morten, for always keeping your door open whenever I wanted to consult my ideas with your machine learning expertise. You could always understand them thoroughly even before I finished explaining them. I would also like to thank my original primary supervisor, the late Jan Larsen, for helping me in setting my first research directions.

Thank you my fellow colleagues in the Cognitive Section at DTU Compute. Special thanks go to Benjamin Johansen with whom I collaborated on numerous projects in the hearing aid domain in the early stages of my PhD project. I would also like to thank Professor Ole Winther for giving me the opportunity to assist in your Deep Learning courses. Thank you, Laurent Vermue and Petr Taborsky, for our collaboration on the Bayesian Cut project. I enjoyed working with you tremendously and learned so many things from you. I would also like to thank my colleagues in Oticon and Eriksholm Research Centre for collaboration in many of the projects revolving around hearing aids. I would also like to
I would also like to thank Emtiyaz Khan for accepting me for an internship at the Approximate Bayesian Inference team at the RIKEN center for Advanced Intelligence Project despite my limited prior experience in machine learning. Big thanks to all my colleagues from the ABI team for making my stay in Japan so enjoyable. I’m also extremely thankful to Alexander Immer, Ehsan Abedi and Matthias Bauer for our collaboration on the projects combining Bayesian neural networks and Gaussian processes. I enjoyed working with you hugely and learned a great deal from you.

Lastly, I would like to sincerely thank my family who has unconditionally supported me throughout my PhD project. Thank you, Mom, Dad, Helena and Monika for always putting faith in me. Without your support that I cannot express in words, I would not be able to reach the end of this project.
Acronyms

BNN  Bayesian neural network. 24, 25
GGN  generalized Gauss-Newton. 23, 25
GGP  generalized Gaussian process. 24, 27
GLM  generalized linear model. 24, 27
GP   Gaussian process. 30, 31, 33, 38, 48, 51, 53
HA   hearing aid. 3, 9, 11, 13, 15, 21, 27, 29, 31, 38, 39, 44, 48, 51, 54, 55, 57, 58
HCP  hearing care professional. 6, 7, 16
NN   neural network. 22, 23, 26, 27, 48
SNR  signal-to-noise ratio. 16, 18
SPL  sound pressure level. 16, 18
Contents

Abstract  i
Summary (Danish)  iii
Preface  v
Scientific contributions  vii
Acknowledgements  ix

1 Introduction  1
1.1 Hearing loss  2
  1.1.1 Prevalence of hearing loss  3
  1.1.2 Consequences of hearing loss  4
1.2 Hearing health care and its challenges  4
  1.2.1 How do hearing aids work?  5
  1.2.2 Low uptake of hearing aids  5
  1.2.3 Hearing aid fitting  6
  1.2.4 Fine-tuning  7
1.3 Paradigm shift in hearing health care  8
  1.3.1 User-driven personalization in hearing aids  8
  1.3.2 Smartphone as a platform for personalization  10
1.4 Research objectives and motivation  11

2 Summary of contributions  15
  2.1 Learning preferences and soundscapes for augmented hearing (Appendix A)  16
  2.2 Modeling user intents as context in smartphone-connected hearing aids (Appendix B)  18
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3 Simulation environment for guiding the design of contextual personalization systems in the context of hearing aids (Appendix C)</td>
<td>19</td>
</tr>
<tr>
<td>2.4 Approximate inference turns deep networks into Gaussian processes (Appendix D)</td>
<td>22</td>
</tr>
<tr>
<td>2.5 Improving predictions of Bayesian neural networks via local linearization (Appendix E)</td>
<td>24</td>
</tr>
<tr>
<td>2.6 Bayesian meta-learning using functional inference in linearized neural networks (Appendix F)</td>
<td>25</td>
</tr>
<tr>
<td>3 Learning contextual preferences via Bayesian meta-learning</td>
<td>29</td>
</tr>
<tr>
<td>3.1 Simulation setup</td>
<td>29</td>
</tr>
<tr>
<td>3.2 Method</td>
<td>35</td>
</tr>
<tr>
<td>3.3 Experiments</td>
<td>40</td>
</tr>
<tr>
<td>3.4 Discussion</td>
<td>47</td>
</tr>
<tr>
<td>4 Discussion</td>
<td>49</td>
</tr>
<tr>
<td>4.1 Exploring context and preferences</td>
<td>50</td>
</tr>
<tr>
<td>4.2 Simulating contextual preferences</td>
<td>51</td>
</tr>
<tr>
<td>4.3 Learning contextual preferences</td>
<td>52</td>
</tr>
<tr>
<td>4.4 Broader perspectives for hearing health care</td>
<td>54</td>
</tr>
<tr>
<td>5 Conclusion</td>
<td>57</td>
</tr>
<tr>
<td>A Learning preferences and soundscapes for augmented hearing</td>
<td>59</td>
</tr>
<tr>
<td>B Modeling user intents as context in smartphone-connected hearing aids</td>
<td>67</td>
</tr>
<tr>
<td>C Simulation environment for guiding the design of contextual personalization systems in the context of hearing aids</td>
<td>73</td>
</tr>
<tr>
<td>D Approximate inference turns deep networks into Gaussian processes</td>
<td>81</td>
</tr>
<tr>
<td>E Improving predictions of Bayesian neural networks via local linearization</td>
<td>101</td>
</tr>
<tr>
<td>F Bayesian meta-learning using functional inference in linearized neural networks</td>
<td>119</td>
</tr>
<tr>
<td>Bibliography</td>
<td>131</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Hearing is undoubtedly one of the core senses in humans. It facilitates efficient communication between people - not only hearing others but also expressing oneself as hearing play an important role in the development of language skills. The ability of hearing also constantly safeguards us as we within hundreds of milliseconds perceive a potential threat before it is visible to our eyes, such as the braking noise of an approaching car that may gives us an extra couple of seconds to jump off, the rattling sound of falling rocks allowing us to take cover or the rumbling sounds of thunder giving us enough time to retreat to safety from an exposed terrain. Hearing is also integral to all the things we enjoy the most, ranging from listening to our favourite songs on Spotify, the thrilling experience of being in the middle of a crowd, whether in live concerts or on a dance floor, being sucked into the sound track of a movie, to simply relaxing surrounded by the sounds of nature. Unfortunately, the precious ability of hearing is not something humans can take for granted. Hearing loss, a partial or total inability to hear, is a common disability that negatively impacts lives of a large fraction of human population.
1.1 Hearing loss

The absolute threshold of hearing, that is the minimum sound pressure level (SPL) that a young human with undamaged hearing can hear, is generally reported to be 0 decibels (dB). As long as the hearing threshold is 25dB or lower, hearing is considered normal\(^1\). If the hearing threshold is over 20dB, it starts being classified as a hearing loss. The severity of hearing loss can be graded as slight (hearing threshold between 26dB and 40dB), moderate (41-60dB), severe (61-80dB) and profound (above 81dB)\(^1\). Additionally, if the hearing threshold is greater than 40 dB in the better hearing ear in adults and 30 dB in children, the hearing loss is classified as disabling. Hearing loss can be caused by a number of factors such as aging, excessive exposure to noise, mechanical damage, heredity or ototoxicity (drug-induced).

The great majority of hearing loss cases originates from dysfunctions in the inner ear. The role of the inner ear is to transform the sound pressure patterns represented as mechanical vibrations in the fluid in the cochlea into electrical signals that can be then passed to the brain through the auditory nerve. The vibrations corresponding to the softer sounds need to be first amplified by the outer hair cells. Hearing loss has been traditionally viewed as a dysfunction of the outer hair cells that inhibits their ability to amplify the soft sounds. As a result, soft sounds no longer elicit any auditory nerve activity meaning that hearing impaired are not able to hear sounds below some threshold that is audible to people with normal hearing. At the same time, the sensitivity to loud sounds is affected to a much smaller degree and the loudest sounds that one can tolerate are usually similar in hearing impaired as in those with normal hearing. The hearing thresholds are also dependent on sound frequency - most commonly, the outer hair cells that are responsible for amplifying signals of higher frequency are affected to a greater extent.

Hearing loss is however a much more complex problem going beyond a sensory loss in the outer hair cells. Lesica \(^51\) highlights the role of non-linear signal processing occurring in the inner ear and describes hearing loss as a "profound distortion of neural activity patterns". The outer hair cells do not just amplify different frequencies but they also capture cross-frequency interactions. Consequently, a dysfunction in the outer hair cells may lead to distortion in neural encoding of sounds and it might be hard for the brain to robustly interpret a complex stimuli like speech, especially in the presence of noise. Lopez-Poveda \(^55\) suggests that degraded neural encoding of speech might arise from insufficient sampling of the sound waveform by the auditory nerve. Hearing loss may also extend beyond the inner ear and the auditory nerve. Kral et al. \(^47\) argues

\(^1\)According to WHO's grades of hearing impairment \(^71\)
that interindividual variability in the brain’s ability of adaptation to hearing impairments may play a role in how well one can process degraded signals.

The intricacies of human auditory system help to understand that sound perception is a complex matter and suggest that it is reasonable to expect a certain degree of variability that cannot be explained just by the loss of sensitivity to soft sounds. This variability has been demonstrated in different studies. Kil- lion [41] showed that the ability to understand speech in noise varied between hearing-impaired subjects with similar hearing thresholds by 15-20dB in signal-to-noise ratio (SNR). Such a large variation helps to explain why some [HA] users may fail to see the benefit of wearing [HA] when it comes to speech understanding in noisy environments. Oetting et al. [70] carried out a study that showed how greatly loudness perception may vary not only between but also within individual users. The authors evaluated the perceived loudness of different test stimuli in 15 hearing-impaired listeners aided with narrowband loudness compensation comparing to loudness perception in listeners with normal hearing. The stimuli differed in frequency bandwidth, sound pressure level and presentation (monaural vs binaural). It was shown that some listeners needed additional gain correction up to 30dB for certain stimuli to achieve normal loudness perception. Only 3 out of 15 listeners showed close to normal loudness perception across all listening conditions. The loudness perception was in general highly dependent on the characteristics of the stimuli which indicates that [HA] users need different amount of loudness compensation in different acoustic conditions.

1.1.1 Prevalence of hearing loss

It has been estimated that hearing loss affects 1.3 billion people worldwide putting it among most prevalent disabilities across human population [100]. The World Health Organization (WHO) estimates that 466 million people suffer from a disabling hearing loss and predicts that this number may reach 600 million in 2030 and exceed 900 million by 2050 [73]. In a review of age-related hearing loss in Europe, Roth et al. [81] established that around 30% of men and 20% of women have a hearing loss of 30 decibels (dB) or more by the age of 70 and by the age of 80 these numbers go up to 55% and 45% for men and women respectively. While hearing loss is often associated with getting older, the problem affects the young population to a great extent as well. Shargorodsky et al. [87] reported a significant increase of hearing loss among US adolescents from 14.9% in 1988-1994 to 19.5% in 2005-2006. The increased risk of acquired hearing loss was associated with the use of personal audio devices [13] and the WHO estimates that 1.1 billion teenagers and young adults are at risk of hearing loss due to recreational noise exposure [73]. Furthermore, it is estimated that 12%-15% of people with hearing thresholds within the limits of normal hearing are affected
by hidden hearing loss [91, 89]. These people report hearing difficulties, most commonly with understanding speech in noisy environments, yet the standard audiometric tests do not indicate a hearing impairment. All together, when considering both diagnosable and hidden hearing loss, recent studies on the US hearing healthcare market suggest that 25% of adults may have problems understanding speech in noise [94].

1.1.2 Consequences of hearing loss

Hearing loss, especially if left unaddressed, can lead to a broad range of consequences negatively impacting the quality of life of hearing impaired as well as other people in their surroundings. Reduced communication skills caused by the inability to understand others may result in social withdrawal, feeling of loneliness, lowered self-esteem and depression [25, 90, 52]. Hearing loss has been linked to a decline in cognitive function [8, 2] including increased risk of dementia [92, 53]. Furthermore, older adults with a moderate to severe hearing loss were found to have a significantly increased likelihood of reporting difficulties with Activities of Daily living compared to those with normal hearing [15, 26]. Untreated hearing loss may also have a profound impact on children’s development of spoken language [67] as well as literacy skills [9]. Apart from consequences on an individual level, hearing loss also brings a huge economical impact. The annual costs related to lowered quality of life and lost earnings caused by hearing loss were estimated to be around £30 billion [1] in the UK alone. Neitzel et al. [63] estimate that prevention of hearing loss related to excessive noise exposure in the US would bring the country an economic benefit of $58 to $152 billion annually.

1.2 Hearing health care and its challenges

While the majority of hearing loss cases cannot be cured as the damage in the inner ear is currently irreversible, in most cases they can be addressed by fitting hearing aids (HAs). HAs are electronic devices whose role is provide a processed (e.g. amplified) sound to the ears of hearing impaired listeners to compensate for their hearing loss.
1.2 Hearing health care and its challenges

1.2.1 How do hearing aids work?

Modern hearing aids offer a range of signal processing capabilities aiming to improve the perception of sounds and intelligibility of speech. The most important aspect of every hearing aid is amplification. As the hearing thresholds in hearing impaired are highly dependent on the frequency of sound, hearing aids provide a different amount of gain to different frequency bands. Additionally, as auditory sensitivity loss is much less severe for loud sounds, hearing aids reduce the amount of amplification with the increasing level of sound. This is referred to as compression as the wide dynamic range of original sounds needs to be compressed into a much narrower range that can be perceived by hearing impaired as having an adequate balance of intensity. One of challenges of applying compression in hearing aids is it should be dynamically adapted as the dynamic range of surrounding sounds depends on the acoustic scene and dynamics of signals within the scene. Additionally, hearing aids are typically equipped with directional microphone arrays allowing for amplifying and attenuating sounds depending on the direction. This can be used to focus the amplification on the frontal direction where typically sources of speech that hearing aid users attempt to attend to are located. Improving speech intelligibility as well as listening comfort can be also achieved by noise and ambient sound attenuation algorithms that aim to selectively amplify the target signal (e.g. speech) from noise and amplify the separated signal and thus increase the signal-to-noise ratio (SNR).

1.2.2 Low uptake of hearing aids

A systematic review has shown a wide range of beneficial effects wearing hearing aids may have on hearing impaired such as improving the ability to participate in everyday situations and improving general health-related quality of life including physical, social, emotional and mental well-being. Despite these potential benefits, most people affected by hearing loss do not wear hearing aids. In lower income countries, this may be in big part attributed to low availability of hearing care in developing countries. According to WHO, the number of audiologists in most of low-income countries is below one per million people. However, even in high-income countries with well-developed hearing health care, only 10%-20% of the population with hearing loss wears hearing aids. Such low uptake of the devices is caused by a number of factors. The high costs of the modern hearing aids can be a major financial burden in countries offering no refunds of costs of acquisition of these medical devices. With the medical evaluation, initial fitting and follow-up fine-tuning appointments typically priced in, a pair of hearing aids from most popular brands costs around $5000. Furthermore, being reluctant to hearing loss treatment is also linked to social and
psychological reasons such as the stigma associated with wearing HAs. There are also other reasons that suggest that the technological aspects of the devices or the current audiological approach might be insufficient. Many people fitted with HAs do not use them. McCormack and Fortnum established that one of the most common reasons for HAs ending up in a drawer is the lack or poor benefit of wearing them. Most commonly, problems with speech intelligibility in noisy environments are reported - users can hear but not understand what is being said. Another problem users commonly experience relates loudness perception - many users complain about sounds being too loud or too soft. While modern HAs are technologically advanced and offer extensive control of how the sound is processed, the current audiological workflow provides very limited opportunities for utilizing the technological sophistication of HAs.

1.2.3 Hearing aid fitting

The process of fitting HAs is carried out by a hearing care professional in an audiological clinic. HAs ship with an API i.e. fitting software provided by manufactures to enable HCPs to quickly program the digital signal processing software in the devices according to the measured hearing loss. Due to the lack of hearing care professionals, and limited time available in a clinic for running a hearing test and programming devices, the fitting interfaces come with predefined default settings that are typically dispensed to the majority of the patients. Manufacturers essentially compete on delivering the most accessible fitting interfaces for HCPs, as simple programming of the devices is a part of the sales process, which may influence what brand among competing HAs the HCP prefers to sell.

The first step of the fitting process is to examine the degree of hearing loss. Hearing loss measurements are typically limited to pure-tone audiogram that measures hearing thresholds for perceiving sine tones at low volume at a fixed set of frequencies in the range between 250Hz and 8kHz. Capturing the sensitivity to soft sounds, these frequency-specific thresholds are translated into a gain-frequency response which defines the amplification provided by the speaker units in the HAs. The HPC does it using the proprietary fitting software provided by the manufacturer, which combines the gain-frequency response with default specific settings controlling functionalities such directionality or attenuation of ambient sounds. The gain-frequency response is calculated according to special prescription formulas, called fitting rationales such as Desired Sensation Level (DSL), National Acoustic Laboratories non-linear version 2 (NAL-NL2), Cambridge Method for Loudness Equalization 2 - High Frequency
1.2 Hearing health care and its challenges

CAM2 or some proprietary fitting rationale by HA manufacturer which adjust the frequency-gain response shape and the compression ratios according to the HA user’s audiogram. Some newer fitting rationales, e.g. NAL-NL2, may also take additional factors into consideration such as wearer’s experience with hearing aids, age, gender. However, even a modern rationale like NAL-N2 simply represents an average amplification paradigm which is not adapted according to individual perception of sounds such as perceived loudness which may vary greatly across as well as within individual users. It also does not adjust amplification according to the listening situation but is optimized to simply boost the speech spectrum (100-2000Hz). This average amplification of the rationale is based on maximizing understanding of speech in noise while not exceeding comfortable loudness in around 200 users [37].

1.2.4 Fine-tuning

After the initial fitting is completed, the patient is asked to use the HA for a few weeks and return to the clinic for a follow-up visit during which the HAs might be fine-tuned. Fine-tuning is traditionally ‘complaint-driven’, i.e. patients provide descriptions of auditory problems they experienced in their daily life while wearing HA, e.g. ‘too loud’, ‘too shrill’ or ‘cannot hear whispers’, which are then translated by the HPC into HA setting adjustments [64, 34]. There are a number of obstacles associated with fine-tuning in a clinic that may make it hard to achieve a positive outcome. First of all, fine-tuning is dependent on the HA user’s recall. Most likely, the user will not remember all the hearing difficulties. Likewise, the user may not remember some relevant details of a problematic listening situation weeks after it happened. For example, the user may remember he had a difficulty hearing speech across the table, but not remember if there were more speakers speaking simultaneously or if there were any background noises. Secondly, as HA have a number of adjustable parameters, the HCP needs to know which (or what combination) of them should be adjusted to achieve the desired effect [64, 93]. Additionally, fine-tuning a certain setting to alleviate hearing difficulties experienced by the user in one situation might cause hearing difficulties in situations that the users had not had problems with before. Last but not least, fine-tuned settings cannot be easily evaluated by the user as it is in most cases impossible to reproduce in the clinic listening situations similar to ones the user had problems in [19]. Consequently, the user may need to repeatedly revisit the clinic if the fine-tuning failed to bring the desired improvement.

These numerous obstacles might explain why clinical fine-tuning has not been shown to have a positive impact on HA satisfaction [14, 85]. Cunningham et al. [14] compared two groups of nine first-time HA users where both groups
received counselling at 30-day intervals for 5 months but only one of them had their HA fitting fine-tuned based on the encountered problems they reported. While the final fine-tuned settings differed from the initial fitting by up to 10dB, no significant differences in HA satisfaction, perceived sound quality and HA usage time were found between the group with fine-tuned settings and the control group. Similarly, speech-in-noise understanding did not show significant differences between the two groups except one out of all 24 measured statistics which was considered by the authors as a spurious finding. In a similar study, Saunders et al. [85] also did not show significant differences in HA satisfaction and benefits between a group of 20 users who had their HA settings fine-tuned and a group of 40 users who kept using their initial fitting, yet the group with fine-tuned settings used their HAs significantly more.

1.3 Paradigm shift in hearing health care

Hearing loss is a complex condition going far beyond a simple loss of sensitivity to soft sounds as approached in audiological health care. Inter- and intra-individual differences in sound perception may result from a wide range of impairments in the auditory system including outer and inner hair cells, auditory nerve and its synapses and even the auditory cortices in the brain. These differences, combined with the increasing prevalence of hearing loss, pose a major challenge for hearing health care which lacks workforce, time and even technological resources. Fitting and fine-tuning process carried out by a HCP in a clinic cannot adequately address listening problems experienced by HA users in specific situations encountered in their daily life.

1.3.1 User-driven personalization in hearing aids

A potential way to address the problems of lacking resources in hearing health care and insufficient personalization to individual auditory needs is to move the fine-tuning process outside the clinics by empowering users to drive the personalization of their HA in the real listening environments they experience. Such paradigm shift can be facilitated by HAs automatically learning preferences of a HA user for different HA settings in different listening environments. Such situation-dependent preferences, that we will refer to as contextual preferences, could be inferred either from user adjustments to HA settings or user perceptual feedback to the settings recommended by a learning system. Subsequently, if a learning system has enough data about the preference of a user in a given environment, it could automatically adjust the settings in that environment. Over
time, users would be expected to obtain personalized settings in an increasing
number of listening environments. In HA research, a hearing aid with such
capabilities has been commonly referred to as a trainable HA.

The first prototype of a trainable HA was developed by Zakis (2003) [102]. The
prototype consisted of HAs connected to a body-worn processor unit that al-
lowed to modify the HA gain settings (overall and frequency-specific) and vote
when the preferred setting was reached. When a vote was submitted, the pro-
cessor recorded the corresponding SNR across three frequency channels and the
overall SNR. Based on the recorded preferred gain levels and the correspond-
ing SNR measurements, the algorithm learned to modify the gain, compression
and noise reduction across three frequency channels. First, 18 hearing-impaired
participants were invited for a four-week training trial in which they were in-
structed to use the processor controls to modify and vote for settings when they
are in a new listening environment. In a following comparison trial, they were
asked to compare, in their daily environments, two settings and vote for the
preferred one. One of the settings was the default setting resulting from a fit-
ting rationale, while the other one was generated by the processor based on the
preferences of users learned through the training trial. Out of 13 participants
who participated in the comparing trial, 9 participants obtained a significant
preference for their trained setting.

Before the first commercial trainable HAs appeared on the market, the potential
benefits trainable HAs might bring to hearing-impaired listeners as well as clin-
icians were discussed by Dillon et al. [18]. The researchers suggested that users
of trainable HAs are expected to experience benefits related to improved lis-
tening in different auditory environments, fewer visits to a clinic for fine-tuning
and increased ownership of the fitting. For clinicians, moving the responsibility
of fine-tuning on HA users would result in saved time which could be used to
provide more counselling, spend more time with more complex patients or fit
additional clients.

The first commercial HA with a trainable feature appeared in 2006 and allowed
to train only the overall starting gain for each of the four available programs
[10]. Over following years, the number of HA manufactures releasing trainable
HAs increased. Most commonly, these solutions focus on learning overall gain,
but some more advanced ones allow for learning frequency-specific gain, noise
reduction or compression [98]. Some solutions may also have a basic context
awareness, training the preferred settings for a few types of acoustic environ-
ments as classified by HAs (e.g. quiet, noise, speech in quiet, speech in noise,
car noise, music [97]). Training traditionally relies on user adjustments to HAs
using HA buttons or through a separate remote control [97].

While commercial trainable hearing aids have existed since 2006, the extent of
their use in clinical practices had remained unknown for many years. Only in 2016, Walravens et al. [98] investigated provision of trainable HAs in Australia. In the survey among clinicians, 53% reported that they activated trainable features, however, the percentage of clinicians who did it always was only 10%. 34% of clinicians were classified as passive non-providers as they could not order trainable HAs, did not know if they can order them, or did not know whether a trainable feature was activated. The remaining 13% actively decided not to activate trainable features. Interestingly, most clinicians who provided trainable hearing aids indicated that this practice did not, or only marginally, changed their fitting and follow-up procedures. These results show that provision of trainable HAs is very limited and has had little impact on clinical workflows.

1.3.2 Smartphone as a platform for personalization

The emergence of smartphone-connected HAs in the recent years has opened a broad range of opportunities to completely rethink personalization in HAs. First and foremost, the connectivity with smartphone provides access to superior computing power and storage capacity. This allows to run advanced machine learning models that could not be run on HA due to very limited resources. Such models could be used not only to learn contextual preferences of a user but also extend context-awareness by performing a much more advanced acoustic scene analysis than the one performed by HAs, or to run advanced signal processing such as signal separation. Moreover, Internet connectivity of a smartphone enables to exchange the data or models between end user devices and services running in the cloud. This gives an opportunity to leverage the contextual preference data collected from millions of users to improve personalization for individual users as well as learn insights about how HA users use their devices in real environments and what needs they have on a scale that had not been feasible before.

Furthermore, smartphones open the door to a wide range of new interfaces such as smartphone apps that could facilitate much more flexible and user-friendly interactions between HA users and HA personalization systems replacing old interfaces with rigid controls such as HA buttons or dedicated remotes. Furthermore, extending interfaces to smartphone-connected smartwatches or smartbands would facilitate interactions without the need of taking the phone out of the pocket. HAs can be also considered as hearables that can receive a stream from a smartphone app and send back a user response or query recorded by on-device microphones. Such voice interactions could be facilitated by extending existing virtual assistants such as Siri or Google Assistant.

Recently, a step in the direction of smartphone-based HA personalization sys-
tems was made by Widex [4]. Using a HA-connected smartphone app, users can find their preferred setting by evaluating a sequence of A/B setting comparisons. Settings in the comparisons are proposed by a machine learning system that takes into account the user’s responses to previous comparisons in a sequence as well as preferred settings collected from other users. The optimized setting can be then saved by a user as a new program.

Rather than inferring preferences from passively observing user adjustments to HA settings, smartphone-based interactive interfaces like the one above allow to take a more proactive approach to HA personalization. This approach may not only help users to explore the complex space of HA settings leveraging preference data collected from other users, but also can increase user engagement providing active assistance and interaction rather than leaving users alone in the personalization process. Furthermore, in this way, machine learning can be used not only for modeling contextual preferences, but also for driving the collection of new data points to facilitate a faster, more efficient personalization process.

1.4 Research objectives and motivation

Personalization of HA settings based on learning contextual auditory preferences of HA users in real environments is a multi-layered challenge. This thesis aims to address different aspects of this challenge in the following three ways:

1. Explore the impact of real-world context on auditory preferences of HA users.

The current audiological approach in hearing healthcare focuses mainly on the problem of speech understanding in noisy environments in clinical settings. Users, however, use their HAs in a wide range of real-world environments and situations, and only few studies have been focused on how the context impacts the users’ perception of sound. To facilitate development of intelligent adaptive solutions that would personalize HA settings according to what helps user most in specific situations, an important first step is to gain a better understanding of how real-world context may affect users’ auditory needs and preferences.

2. Simulate auditory preferences based on observable and latent user and context features.

Building robust machine-learning solutions for contextual personalization of HA settings requires the ability and flexibility to evaluate different aspects of these solutions to know how they perform and what needs to be
improved. As there are no domain datasets that could be used for this purpose, the only alternative is currently carrying out user studies. However, user studies are typically costly, limited in size and scope, and require lots of time to prepare and run, which makes it impossible to quickly iterate and improve different aspects of the developed methods in an efficient way. Furthermore, the inaccessibility to the ground-truth of preferences of real users renders a systematic evaluation and comparison of the developed methods challenging.

Simulating auditory preferences and interactions between users and preference learning systems may provide a basis for a more efficient and scalable development of contextual personalization systems for HA users. While simulated preferences are bound to provide only a very crude, inaccurate representation of real preferences, simulations with varying assumptions about the diversity of preferences within and across users as well as the observability of features inducing these preferences can help to quantify strengths and weaknesses of different methods, and improve their robustness before testing and deploying the most promising solutions in real environments.

3. Develop machine learning methods that can efficiently learn contextual preferences from limited amounts of noisy user feedback.

Learning contextual preferences from perceptual feedback of HA users is undoubtedly a challenging task. Perceptual user feedback is inherently noisy - factors such as cognitive capabilities, hastiness or tiredness are bound to affect the consistency of user responses. It is therefore important that the method for learning preferences can quantify the noise in user feedback and reduce the risk of overfitting to spurious patterns. The frequency and the amount of feedback an average user is able (and willing) to provide is also limited. It is thus vital that the method can not only efficiently use the limited data but that it also provides tools for collecting new data points in an informed way promoting a more efficient learning.

Similar to how people share preferences for specific kinds of movies or music, we expect that HA users are not completely different but exhibit some similarities in their contextual auditory preferences. Learning these similarities and how different user characteristics can be linked to them may have a great impact on how quickly the method would learn preferences of individual users.

Moreover, it is impossible to know, before enough real-world data is collected, how well the observed user characteristics and context features explain contextual auditory preferences of HA users. In fact, considering how complex and underexplored the impacts of user characteristics (such as the ones related to the auditory system) on how users perceive sounds appear to be, it is highly unlikely that it will be possible to identify (and
1.4 Research objectives and motivation

acquire) user features that would allow to accurately predict preferences for less engaged users or new users before they provide enough auditory feedback. Likewise, given how diverse and intricate the environments and situations users encounter may be, we might never be able to identify and collect contextual characteristics that would allow to perfectly discern between situations in which a user has different auditory preferences. For this reason, it is desirable that the method should be robust to incomplete information about context and user, and also be able to infer the latent structure based on the observed data.

The more detailed context data and the larger quantities of it we collect from users, the more accurately we would be able to capture their contextual preferences and adapt HA's settings to them. However, context data is of a very personal nature (especially if detailed and abundant) and sharing it would pose a serious threat to users’ privacy. In order to protect users’ privacy, we might therefore also need to consider federated learning approaches that allow to learn patterns across many users from data stored on end-user devices without the need of sharing the data to a central server.

Outline of the thesis

The rest of this thesis is structured as follows. Chapter 2 summarizes the main contributions from the articles accompanying this thesis. Chapter 3 shows a practical application of the developed methods in the problem of contextual personalization of HA settings. Chapter 4 discusses the limitations of the conducted research, suggests future directions and provides broader perspectives for how contextual personalization can help to redefine hearing health care. Finally, Chapter 5 concludes this thesis.
Summary of contributions

This chapter summarizes the articles that this thesis is based on and that are attached in Appendices A to F. Section 2.1 outlines the main findings from a user study that focused on exploring contextual preferences of HA users in real environments. Section 2.2 outlines a position paper that discussed how different types of context could help to distinguish between situations in which HA users may have different auditory needs. Section 2.3 summarizes the research on simulating contextual preferences of HA which aimed to establish a new method for evaluation of contextual personalization systems in the HA domain. Section 2.4 introduces parallel research carried out during a research stay at the Approximate Bayesian Inference team in RIKEN. This research focused on investigating connections between approximate Bayesian inference in neural networks and inference in Gaussian process models. Section 2.5 outlines the paper providing a different perspective to and generalizing the method described in Section 2.4. Finally, Section 2.6 summarizes an unpublished manuscript that introduced a novel probabilistic meta-learning method inspired by previous research described in Sections 2.4 and 2.5. This section also explains how the proposed method could be used as a framework for learning contextual preferences of HA users.
2.1 Learning preferences and soundscapes for augmented hearing (Appendix A)

As described in the introduction, the standard audiological approach for fitting HAs is based on measuring pure tone audiogram that quantifies the loss of sensitivity to sounds of different frequencies, and translating it into amplification characteristics of HAs using generic or proprietary fitting rationales. These rationales are commonly established with the goal of improving speech intelligibility in noisy environments and/or normalizing loudness perception in an average hearing-impaired listener with a specific pure-tone audiogram [37, 60]. While being simple to implement in audiological workflows, this approach fails to account for the large differences in sound perception between listeners with similar audiograms as well as within individual listeners in different auditory environments and situations they encounter in their daily life. This lack of personalization to individual needs leads to reduced perceived benefit of wearing HAs and in many cases leaving them unused [31, 59]. Even though HAs allow for very advanced fine-tuning, this process takes place in the clinic and relies on the HA user’s recall of the difficult listening situations weeks after they happened as well as the HCP knowing which of many HA settings to tune [64, 34]. Research so far has shown very limited benefit of this approach [14, 85]. A possible solution would be to move from fine-tuning performed by an HCP in a clinic to user-driven fine-tuning in the real world.

In this paper, we present the results of a pilot study in which we recorded 1) how two hearing-impaired subjects over 6-7 weeks chose between four contrasting programs in real world situations and 2) acoustic context measured continuously by the HA consisting of SPL, SNR, modulation characteristics and classification of the environment as 'quiet', 'speech', 'speech in noise', 'noise'. The four programs reflected different fine-tuning capabilities of the HAs related to soft sound compression, high frequency gain and attenuation of ambient sounds. The soft sound amplification can be adjusted to reflect individual differences in loudness perception [49, 70]. Enhancing soft sounds increases the intensity and completeness of a sound scene but it also means that there are more sounds a HA user needs to be able to cope with. High frequency gain influences the perceived brightness and may provide additional spatial cues due to interaural level differences making it easier for the user to localize sounds and apply selective attention to e.g. concentrate on following a voice in background noise [5]. Modifying the degree of attenuation of ambient sounds may reflect the interindividual variability in speech intelligibility in noisy environments [11] by improving SNR of speech signals, though at the cost of removing auditory objects present in the soundscape.
The results showed that the two subjects were exposed to very different acoustic environments with subject 1 spending considerably more time in speech scenarios. The subjects also showed a large variation in the sound environments experienced throughout the day. Both subjects spent most of their HA usage time in programs offering less attenuation of ambient sounds than normally provided by the default fitting in the clinic. This preference was visible even in complex environments suggesting that the subjects were able to cope by selectively directing their attention to specific sources. Subject 1 occasionally used a program with high attenuation of ambient sounds but only in the most complex environments. In contrast, subject 2 appeared able to cope even in the noisiest situations by relying on a program with no attenuation but boosting brightness and soft sounds. This was most evident at lunch time when the subject frequently selected this program. Subject 1 preferred a similar setting in quiet environments to increase the intensity of the nature sounds during walks in the evening as reported in a follow-up interview. It is also of particular interest that subject 2 spent almost no time in the program corresponding to the default clinical fitting.

The findings indicate that hearing-impaired listeners may have preferences highly different from the default fitting commonly provided in a clinic and that these preferences might be very dependent on the context. This illustrates the need for contextual personalization which is not possible in traditional clinical workflow due to lack of hearing health care resources. It requires that the user is able to compare and evaluate different HA settings in real listening situations rather than trying to recall the problems experienced in past weeks during a follow-up clinical visit. In this study we observed relatively few program changes compared to how often the context changed. This suggests that in order to be able to learn contextual preferences of hearing aid users faster, it might require a more proactive interface for prompting users to engage. The programs provided in this study represented only a few discrete points in a much larger fitting space. For a user to explore this fitting space efficiently, it might be beneficial to implement methods that suggest specific settings based on past interactions of that user as well as preferences learned from other similar users.
2.2 Modeling user intents as context in smartphone-connected hearing aids (Appendix B)

Fitting rationales that provide the foundation for translating hearing loss into amplification characteristics of HA are optimized for speech understanding at comfortable loudness [37]. However, HA users are exposed to constantly varying listening scenarios in which they may have highly contrasting preferences [45] as summarized in Section 2.1. These preferences may be associated with different ways of compensating for a hearing loss needed to satisfy a particular auditory intent, e.g. interacting in a conversation, listening to music or ignoring surrounding sounds to focus on reading. The ability to capture different aspects of context is a prerequisite for learning such situation-dependent preferences and subsequently adapt the HA settings automatically. Most modern HAs can be connected with smartphones which provides new opportunities for collecting and processing the context data not only from HAs but also from smartphones, wearables as well as cloud services.

In this position paper, we discuss what kind of context data might be helpful to distinguish between different situations in which HA users might need different HA settings to satisfy their auditory intents. Different listening scenarios could be described by combining information about:

- **acoustic scene** - soundscape continuously captured by the built-in microphones in the HA can be processed at different levels:
  - basic characteristics of sound environment such as SPL, SNR or modulation which are used by HA for controlling internal signal processing
  - acoustic scene classification providing a high level understanding of the environment such as 'nature', 'traffic', 'cafe', 'speech', 'music'
  - voice characteristics that may vary greatly in pitch, timbre, pace, modulation, articulation or be related to a specific speaker

- **location** - GPS-based data accessed from a smartphone which can be processed as:
  - frequently visited locations that can be inferred from clustering of geospatial coordinates or manually tagged by users
  - venue classification using public APIs such as Google Places or Foursquare (e.g. Arts & Entertainment → Performing Arts Venue → Theater)
2.3 Simulation environment for guiding the design of contextual personalization systems in the context of hearing aids (Appendix C)

- **motion** - accelerometer and GPS based data that can be used for predicting the current activity of a user such as driving, running, stationary

- **time** - e.g. time of day (specific hour or morning/afternoon/evening), weekday/weekend that may be associated with specific activities or even cognitive fatigue

Based on a user study with 10 hearing-impaired participants carried out over 2 months, we provide a number of examples of real listening scenarios which could be identified by collecting these different types of context information.

Context-awareness is the basis for learning situation-dependent preferences which might be enabled by either observing user-made adjustments to the HA settings or collecting explicit user feedback for HA setting recommendations e.g. in A/B comparisons in different situations represented by the momentary context data. However, even though extensive context-awareness may help to reliably discriminate between situations in which HA users prefer different settings, learning generalizable patterns in a high-dimensional space from limited data is a major challenge. This might be complicated even further by missing data e.g. due to privacy concerns or unavailability of sensors. We might need to consider identifying what types of context information help to explain most of the variation in user preferences to facilitate more robust learning.

2.3 Simulation environment for guiding the design of contextual personalization systems in the context of hearing aids (Appendix C)

The richness of context information and the diversity of adjustable HA parameters provide a basis for learning fine-grained preferences of individual users. This however also poses a challenge as limited preference data most users may feasibly provide would occupy the complex space of contexts and settings very sparsely which may render efficient and robust learning of contextual preferences impossible. To address the problem of sparsity in contextual preference data, we may consider leveraging correlations in user preferences as some studies suggest that there is a certain degree of similarity in preferences across HA users. For example, in a study conducted by Oetting et al. [70] where gain was adjusted monaurally for narrowband signals, 4 out of 15 hearing-impaired listeners showed a need for reducing gain when perceiving broadband signals binaurally while exhibiting close-to-normal perceived loudness for signals that were either narrowband binaural or broadband monaural. Most likely, such correlations
Summary of contributions

in hearing loss compensation needs are not coincidental, but rather caused by some underlying factors that are shared across a group of listeners. In many cases, individual preferences and needs cannot be explained by simple hearing loss profiling such as a pure-tone audiogram, but they may be explained by a more complex profile describing the hearing impairment and sound perception, or even other factors such as cognitive capabilities. For example, Neher et al. [62] showed a significant correlation between poor working memory in hearing-impaired listeners and a preference for strong noise reduction. Capturing [HA] user characteristics that help in predicting user preferences would play an important role in addressing the ‘cold-start’ problem i.e. how quickly or with how little data a [HA] setting personalization system could learn the contextual preferences of new users and provide perceivable benefits to them. However, many relevant characteristics that induce specific preferences may be unknown or may be unmeasurable in standard audiological workflows due to the lack of time or technological resources. While such characteristics cannot be observed, certain algorithms may infer the latent structure in the data (i.e. structure that cannot be explained by the observed variables) e.g. by means of latent variable modeling [16], and benefit from it.

Developing multi-faceted [HA] personalization systems that can learn efficiently from limited data taking into consideration aspects such as diverse context, preference correlations across users and limited observability of relevant context or user characteristics requires a framework in which one can evaluate different approaches and algorithms and iteratively improve them. A traditional approach to evaluate methods for [HA] setting personalization is through user studies [101, 65, 99]. User studies may not however be a viable approach here as they often have a limited number of participants and may not provide a sufficient basis for evaluating multiple aspects of the designed solution. It also takes weeks or even months to design, arrange and run a user study which does not allow for an agile approach to iteratively improve and reevaluate the developed system. Additionally, at times such as the COVID-19 pandemic, it may not even be feasible to carry out user studies due to safety restrictions.

In this paper, inspired by the recent work on simulation-driven design and evaluation of recommender systems [80, 32], we propose to use simulation environments to develop and evaluate [HA] setting context-aware personalization algorithms in order to circumvent the practical limitations associated with user studies. The goal of this approach is not to create a "perfect" simulation environment but rather one that mirrors specific aspects of contextual preferences and user behavior found in real world to serve as a controlled environment for developing, evaluating and comparing systems and algorithms for [HA] personalization. By carrying out simulations under varying assumptions about aspects such as complexity of contextual preferences of individual users, correlations in preferences across different users, partial observability of features characterizing
users and context or noisiness of user responses, we could aim to iteratively improve the performance and robustness of different personalization methods in simulated conditions before deploying and evaluating the most promising solutions in the real world.

The primary focus of this paper is on generating plausible contextual preference models for users. To that end, we assume that contextual preferences can be uniquely determined given context that is characterized by both observable and latent context features, i.e. features that may or may not be observed by a learning agent. We also assume that contextual preferences have a certain similarity structure across users which is dependent on observable and latent user features. As the first step, we sample preferred settings for a set of users over a range of contexts from a Gaussian Process prior whose covariance encodes our assumptions about the preferences such as their complexity, similarity structure across as well as sensitivity to observable and latent components of user and context features. These assumptions can be flexibly controlled by modifying the hyperparameters of the GP kernels operating on context and user features. In the second step, we use the sampled preferred settings of a user to train a GP regression model that can be used to evaluate the latent preference of that user for any context and any settings within the assumed context and settings space. This latent preference model becomes then the basis for simulating noisy user responses to contextual setting recommendations of a learning system (i.e. agent). Finally, based on a simple example of a simulation environment, we demonstrate how a GP-based agent supported by Bayesian optimization can balance exploration and exploitation to gradually learn the simulated contextual preferences of a user through a sequence of interactions with that user in varying contexts.

Designing an agent to perform well in a simulation environment does not guarantee its performance when deployed in the real world. It is very likely that when an agent developed in a simulation environment is eventually tested on real users, the acquired real data may reveal that some of the assumptions implemented in the simulation environment or the agent are significantly mismatched with the characteristics of the real-world environment and appear to negatively impact the learning capabilities of the agent. In such a case, one can remodel these assumptions according to the new knowledge coming from the real data and improve the agent based on new simulations with revised assumptions to make sure that the new version of the agent is better suited to deal with particular characteristics of real environments. This approach can create a virtuous circle in which real world data collected by the solutions deployed in the real world helps to improve the simulation environment which in turn helps to develop solutions that are more effective and robust in real conditions.
Approximate inference turns deep networks into Gaussian processes (Appendix D)

Neural networks (NNs) and Gaussian processes (GPs) are powerful machine learning models that have complementary strengths and weaknesses. NNs can model very complex functions by means of a complex, yet flexible parametric representation \[^{66}\]. They are very effective in handling diverse and complex data types such as images or text and can easily scale to millions of data points achieving state-of-the-art performance on many real-world problems. However, the flexibility of NNs comes at a price. NNs are prone to overfitting, especially in problems with limited data, and typically require expensive fine-tuning of numerous hyperparameters involving training dozens or hundreds of models with different configurations to find one with optimal performance. The complex parametric structure of NNs also makes interpretability of the predictions challenging. Additionally, standard NNs do not account for uncertainty in parameters and tend yield overconfident predictions.

In contrast to NNs, GPs are non-parametric models that model functions directly by placing a prior on them \[^{76}\]. Being a Bayesian method, GPs offer a principled and efficient approach to tuning hyperparameters by means of optimizing the marginal likelihood which helps to train well-generalizing models even with very limited data. Predictions in a GP model have high-quality uncertainty estimates and can be expressed as a linear combination of the training observations which may facilitate a simple and intuitive interpretation. However, to enjoy these benefits GPs require a careful design of the kernel which may be challenging in many applications, especially those involving high-dimensional data. Additionally, exact inference in GP models comes with a huge computational and requires approximations to scale the method to large datasets \[^{54}\].

The problem of overconfidence in NNs can be potentially addressed by making NNs Bayesian. By imposing a prior distribution on NN's parameters and applying Bayesian inference for a given dataset, a posterior distribution is obtained which quantifies the uncertainty about NN parameters which in turn allows to quantify the uncertainty of the predictions. However, exact Bayesian inference in NNs is in practice intractable and approximate inference schemes have to be used. The Laplace \[^{56,79}\] and variational approximations \[^{6,39,103}\] are two common methods to approximate the intractable posterior distribution of a NN. The Laplace approximation fits a Gaussian distribution centered at the posterior's mode (MAP) which can be obtained by standard gradient-based optimization methods used in NNs. In contrast, rather than fitting a Gaussian distribution post-training, the Gaussian variational approximation is obtained by applying variational inference which updates the parameters of a Gaussian
distribution in an online fashion by optimizing a lower bound to the marginal likelihood. However, to evaluate the covariance of the approximate posterior, the Hessian of the loss function is required which in neural networks is not guaranteed to be positive-semidefinite and thus may not lead to a valid covariance matrix. A common way to address this issue is to approximate the Hessian with the generalized Gauss-Newton (GGN) matrix [68, 57] which is guaranteed to be positive-semidefinite and can be computed only using first-order derivatives. Furthermore, computation of a full covariance matrix is typically infeasible due to its prohibitive size and computational costs. In practice, diagonal or other structured approximations to the Gaussian posterior are employed to reduce the computational and space complexity of inference [6, 103, 79].

Numerous efforts have been made to relate NNs and GPs. Originally, Neal [61] showed that the outputs of an infinitely-wide single layer feed-forward NN with parameters sampled from an i.i.d. prior follow a Gaussian Process. Subsequently, the equivalence between these two models was shown for deeper NNs and more complex architectures, and expressions for the corresponding GP kernels were derived [29, 50, 69, 24]. All these works considered infinitely-wide NNs with parameters sampled from a prior distribution.

In this paper, we establish the relationship between approximate inference in a finite-width Bayesian neural network and inference in a GP model. First, we show that the Laplace posterior with GGN approximation is equivalent to the posterior of a linear regression model obtained for a transformed dataset. Then, using the equivalence between the parameter-space and function-space view, we derive the corresponding GP model whose predictive distribution is equal to the predictive distribution of the linear model. Further, we show an analogical relationship for the updates of a variational Gaussian posterior with GGN approximation.

We demonstrate the benefits of the proposed approach on a number of problems. First, we visualize the predictive uncertainty of the GP arising from a NN trained on a simple regression task. The uncertainty increases wherever the data is not observed which is a desirable property in many applications such as active learning, Bayesian optimization or outlier detection. Secondly, we visualize the kernels and predictive distributions of GPs obtained from NNs trained on CIFAR-10 and MNIST datasets. Inspecting the kernels helps us to understand the interactions between different classes and data examples and explain the predictions of a NN. Finally, on two regression datasets (synthetic and real-world), we demonstrate how we can tune the hyperparameters of a NN using the marginal likelihood of the derived GP model to find well-generalizing models without using cross-validation.
Approximate posterior inference in Bayesian neural networks (BNNs) typically requires making approximations to the Hessian of the loss function of the BNN. A common choice is the GGN approximation introduced in the previous section. To make predictions in a BNN, one commonly combines an approximate posterior with the original BNN likelihood using Monte-Carlo sampling, i.e. the predictions of the BNN model are obtained for a number of parameter samples drawn from the approximate posterior. This approach may lead however to underfitting which may be particularly severe when full posterior covariance is used [79].

In this paper we argue that the GGN approximation should be viewed as a implicit modification of the underlying probabilistic model and should be considered separately from further approximate inference techniques. Applying the GGN approximation works as a local linearization and thus gives rise to a generalized linear model (GLM), or equivalently, a generalized Gaussian process (GGP) model. Subsequently, performing approximate inference in a BNN with GGN approximation results in an approximate posterior that is consistent with the GLM but not the original BNN model. Consequently, when making predictions with this approximate posterior, one should use the likelihood of the GLM rather than that of the original BNN model. This formulation generalized previous work by Khan et al. [40] (Section 2.4) and Foong et al. [22] to non-Gaussian likelihoods and explains underfitting observed when combining the Laplace and GGN approximations with a BNN likelihood [79].

Inference in the linearized model resulting from applying the GGN approximation can be performed in closed-form for Gaussian likelihoods while for non-Gaussian likelihoods any approximate inference technique (e.g. Laplace or variational inference) can be used. Depending on the size of the BNN model and the size of the dataset, one may choose to perform inference either in a GLM model with computational cost being cubic in the number of BNN parameters, or in a GGP model with computational cost being cubic in the number of data points. Additional approximations might be used to reduce the computational (and storage) complexity even further. Inference in a GLM model may use structured approximations of the covariance matrix of the parameters, e.g. diagonal or Kronecker-factored [58, 79]. For a GGP model, one can use a scalable GP inference method, e.g. variational inference with inducing points [30].

We demonstrate the benefits of the proposed GLM/GGP predictive in a number of experiments on synthetic and real datasets. As the proposed predictive
for regression problems is equivalent with the one derived by [40] and [22], we focus on classification problems. First, we compare the BNN and GLM/GGP predictives on a 2d toy classification dataset. The GLM/GGP predictive alleviates the underfitting problem present in the BNN predictive. It also shows uncertainties growing away from the data which is not the case when predicting with the deterministic MAP estimate that was used to construct the Laplace approximation used in both BNN and GLM/GGP predictives. We also show that the predictive uncertainty meaningfully decomposes into an aleatoric component that explains the uncertainty about predictions due to the overlap between different classes visible at the decision boundaries, and an epistemic component that reflects the model’s inability to make confident predictions due to limited data in certain regions of the predictive space. Further, we demonstrate the advantage of the GLM predictive over the BNN predictive on a number of UCI binary and multiclass classification datasets. We also show that in most cases variational inference in the GLM model yield better results than the Laplace approximation.

When applying the GGN and Laplace approximation in a BNN one assumes the parameters’ estimate resulting from gradient-based optimization to be the mode of the true posterior (i.e. gradient of the objective being zero) and fits a Gaussian distribution at that estimate [79] [22]. In practice however, the gradient is typically not exactly zero and consequently the obtained estimate is not a mode of the posterior, neither that of the BNN, nor that of the corresponding GLM/GGP. As the GLM/GGP objective is convex and thus easier to optimize, the estimate of the mode at which Laplace approximation is applied can be refined. This procedure may improve the predictive performance which we demonstrate on the CIFAR-10 dataset.

2.6 Bayesian meta-learning using functional inference in linearized neural networks (Appendix F)

Learning from few examples is a skill that humans excel at. It is believed it arises from building up prior knowledge and being able to utilize it adequately to solve new unseen problems [48]. Meta-learning is a subfield of machine learning that focuses on building algorithms that can mimic this human skill. A typical meta-learning problem is formed by a large collection of independent yet in some way related tasks where each task has very little data. The goal of a meta-learning algorithm is to ‘learn how to learn’ to solve these tasks efficiently and apply the learned knowledge to solve new unseen tasks. Due to the flexibility to address a
wide range of complex real-world problems, NNs have become a very common modelling tool in meta-learning in the recent years [84, 95, 78, 88, 21].

A method that gained particular popularity is model-agnostic meta-learning (MAML) [21]. MAML aims to learn initialization parameters of a model (most commonly a NN) such that they can be efficiently adapted to solve a given task by taking a small number of gradient steps on the loss function of that task. However, while being universal and effective, MAML only provides deterministic predictions and consequently may not be a suitable candidate for many practical applications in which it is important to account for uncertainty and ambiguity in possible solutions of a given task. A theoretically compelling solution to this problem is to reframe this approach using Bayesian modeling framework. Instead of learning a deterministic initialization, one would learn a prior for the parameters of a NN and instead of adapting to individual tasks using optimization, one would perform posterior inference in the NN using the learned prior.

Unfortunately, Bayesian inference, which is already very difficult in NNs, becomes even more challenging in the context of meta-learning. Instead of performing inference once on a single dataset, one needs to perform it thousands or millions of times on a large number of small datasets. To make this feasible, different approximate schemes have been proposed to greatly reduce the complexity of individual inferences. Ravi and Beatson [77] learned parameters of a diagonal Gaussian posterior approximation using optimization similarly as in MAML but on a variational objective. Gordon et al. [27] restricted probabilistic inference only to the last linear layer predicting the parameters of a diagonal Gaussian posterior approximation using an amortization network. Instead of resorting to very rough approximations of inference procedure, one could alternatively consider choosing a model which naturally facilitates more efficient inference such as a GP. Inference in GPs is performed in function-space which offers a major computational advantage due to miniscule size of task datasets commonly used in meta-learning. However, due to the challenge of construction of GP priors that would be well-suited for solving complex tasks such as natural image classification, GPs have received very limited attention in the domain of meta-learning [23].

In this paper, we propose a new approach to meta-learning that combines the flexibility of parametric structure of NNs with the efficiency of Bayesian inference in GPs. We take inspiration from our previous work [33] (Section 2.5) and linearize a NN to convert into a GLM/GGP model; however here, instead of the mode of the posterior we use the mean of the prior as the linearization point. We aim to learn the mean and (diagonal) covariance of a Gaussian prior such that inference in the GLM model with this prior for individual tasks provides good generalization. We learn this prior by optimizing one of two objectives,
the marginal likelihood or the posterior predictive likelihood, in the correspond-
ing GGP model to leverage the miniscule size of individual datasets and avoid resoring to rough approximations of the parametric posterior. Adaptation to individual tasks is performed by posterior predictive inference resulting in a distribution over prediction. Inference with Gaussian likelihoods admits closed-form solutions to the marginal likelihood and posterior predictive distribution, while in the case of non-Gaussian likelihoods an approximate inference scheme (such as the Laplace approximation) needs to be used.

We demonstrate the properties of the method on a set of synthetic regression and binary classification tasks. We learn a diagonal Gaussian prior for a NN-based GLM model using tasks representing sine waves with different amplitudes and phase shifts, or linear functions with different slopes and offsets. We show that sampling functions from the GLM model using the prior trained on linear function tasks yields samples that look completely linear and cover a range of offsets and slopes resembling the true distribution of these tasks. The high quality of the prior is reflected in the excellent generalization and uncertainty estimates that we demonstrate by visualizing posterior predictive distributions for a set of sampled tasks. Meta-learning on sine function tasks shows to be a slightly more complex problem as the function samples drawn from the prior, while maintaining the right periodicity and representing a range of phase shifts and amplitudes, in many cases fail to maintain a constant amplitude. Nonetheless, the trained prior enables very good generalization and reasonable uncertainty quantification. We also show that increasing the number of data points used for task adaptation leads to consistent contraction of the posterior for both training objectives. Similarly, we demonstrate the quality of the trained prior as well as and posterior inferences for a binary classification task distribution. Due to the employed approximations, the quality appears to be lower than for regression problems. We also show that the posterior predictive likelihood objective might be a better choice in classification as adding more data points during adaptation leads to a more consistent contraction of the posterior than in the case of the marginal likelihood objective. We leave it to future work to compare our method with other Bayesian approaches to meta-learning and extend it more complex problems such as image classification.

Meta-learning could be a powerful tool to address the problem of contextual personalization in the HA domain as it can be viewed as a natural framework for performing personalization. Additionally, the probabilistic formulation of the proposed meta-learning approach offers benefits that may be particularly important for facilitating efficient personalization based on limited preference data. A HA user can be considered as an individual task with a dataset formed by the preference data (e.g. user’s perceptual feedback to specific settings recommended in different contexts) that has been acquired so far from that user. Task datasets of all users would share some similarities reflecting correlations in
Summary of contributions

user preferences, yet they would also have some individual variability. A meta-model with a parametric prior would aim to represent prior knowledge about the contextual preferences of users, e.g., how different settings, contexts, and user features relate to users’ perceptual feedback. Training the prior would aim to encapsulate the similarities in user preferences and serve as a good starting point for personalization to individual differences. Adaptation, facilitated by posterior inference, would then update the prior based on the preference data of a given user to account for the individual differences in the contextual preferences of that user. The posterior predictive distribution over the preferences of a user would enable a learning agent to use methods such as Bayesian optimization to efficiently trade-off the exploitation of the setting preferences learned so far with the exploration of new settings that the user might like. Similarly, the agent could leverage the knowledge about what is known and what is unknown to determine in which contexts that settings can be automatically adapted and in which contexts the user should be prompted to provide more feedback. Meta-learning can be also combined with federated learning to facilitate training of a shared model incorporating correlations in the data across many users while keeping the sensitive training data on the end users’ devices. This could help to address the privacy concerns of users over the extensive context data that may be collected from them. In the next chapter, we demonstrate the discussed application of the proposed meta-learning method to the problem of contextual personalization of hearing aid settings in simulated environments.
In this chapter we demonstrate how to learn simulated contextual preferences of HA users using the Bayesian meta-learning framework proposed in Korzepa [43] and summarized in Section 2.6. In Section 3.1 we define the problem using a simple simulation environment based on the simulation framework proposed in Korzepa et al. [44]. In Section 3.2 we formalize how the meta-learning framework combined with Bayesian optimization are used to address the defined problem. In Section 3.3 we present the results of the simulations carried out in the developed environment. In Section 3.4 we discuss the findings from this experiment.

3.1 Simulation setup

We assume a population of $N = 700$ users that splits into a set $\mathcal{U}_{\text{old}}$ consisting of 500 'old' users who have already interacted with a HA personalization agent and a set $\mathcal{U}_{\text{new}}$ consisting of 200 new users who have not interacted with the agent yet. The goal is to provide contextual personalization of HA settings for
this group of new users.

We model this problem in a simulation environment similar to the one proposed in Korzepa et al. [44]. The main difference lies in how we approach observability of user and context features. In Korzepa et al. [44], we explicitly defined observable and hidden features and controlled the extent of their impact on preferences by modifying individual length scales corresponding to observed and hidden components in the GP kernels. This approach does not allow to keep the same form and complexity of simulated preferences when different assumptions about observability are to be tested. Here, in order to reuse the same preference models across simulations with different observability assumptions, instead of modifying the length scales, we modify the number of features that are observed by a personalization agent. In the following sections we describe how we generate a simulation environment for this problem.

Sampling users

Each user $j$ is characterized by a two-dimensional user vector $u_j = (u_1, u_2)$. We sample 150 user vectors from each of three diagonal Gaussian distributions with means $[-2.5, -1.5]$, $[2.5, -1.5]$, $[0, 2.5]$, and unit variances. Further, we sample 169 users uniformly from $[-5, 5]$ range in both dimensions. For visualization purposes, the vectors for the remaining 81 users lie on an integer grid $\{-4, -3, \ldots , 4\}$ in both dimensions. The split in training and test users is performed randomly. To introduce correlations between preferences arising from similarities in the characteristics of these users, we use a radial basis function (RBF) kernel:

$$
\kappa_u(u, u') = \exp \left( -\frac{||u - u'||^2}{2\lambda_u} \right)
$$

(3.1)

where $\lambda_u$ is the length scale parameter that we set to 2.5. In Figure 3.1 we show the sampled user vectors and the kernel matrix resulting from the kernel $\kappa_u$.

Defining context and HA setting features

Similarly as in the demonstration in Korzepa et al. [44], we assume two-dimensional context $c = (c_1, c_2) \in C$, where $C = [-2, 2]^2$. To induce a certain degree of smoothness in preferences of individual users, we use another RBF kernel $\kappa_c(c, c')$ with length scale $\lambda_c = 1$. Further, for the sake of simplicity, we assume a one-dimensional HA setting $s \in S$, where $S = [-5, 5]$. 
3.1 Simulation setup

(a) user vectors  
(b) user kernel matrix

Figure 3.1: (a) User vectors located on a grid (●), sampled uniformly (●) and sampled from Gaussian distributions (●), and (b) the corresponding kernel matrix.

Sampling preferred settings

To generate user preference models using GP regression, we first construct user-specific datasets that describe user preferred settings in a range of contexts. For each user $j$, we generate a dataset $D_j = \{(c_i, \hat{s}_{ji}, f_H)\}^{M}_{i=1}$, where $\hat{s}_{ji}$ is the setting for which user $j$ achieves a high preference value $f_H = 2$. For visualization purposes, we choose the set of $M$ contexts $\{c_1, \ldots, c_M\}$ to represent points on an equispaced $L \times L$ grid with ranges $[-2, 2]$ in both dimensions. We set $L$ to 50. To obtain settings $\hat{s}_{ji}$, we draw a single sample from a multi-task GP prior:

$$(\hat{s}_{ji})_{j=1 \ldots N, i=1 \ldots M} \sim \mathcal{N}(0, K_c \otimes K_u),$$

(3.2)

where $(K_c)_{pq} = \kappa_c(c_p, c_q)$ and $(K_u)_{pq} = \kappa_u(u_p, u_q)$.

As the full covariance matrix resulting from the Kronecker product in (3.2) is too large to store and draw samples exactly, we apply low-rank root decomposition (with rank 500) as implemented in the GPyTorch library. The sampled settings are shifted such that their mean is at 0 so that $s = 0$ reflects the default setting provided by a standard fitting rationale that fits best an average user. Further, the settings are scaled such that 10% of them exceed the edge value of -5 or 5 that can be set in the assumed HA. This is done to reflect the possibility that the available range of HA settings might not be sufficient to fully adapt to all contextual preferences of all HA users. In Figure 3.2 we visualize the transformed sampled settings for the set of 81 users with user vectors lying on
Figure 3.2: A sample of preferred settings for 81 users with vectors located on a grid. Each heatmap corresponds to the same context grid defined on $c_1, c_2 \in [-2, 2]$. 
Figure 3.3: Latent preference for 9 users with $u_2 = -4$ and different values of $u_1$. The preferences are shown for $c_1 \in [-2, 2]$, while $c_2$ is kept fixed to 0. The dashed line indicates the maximum and minimum values of setting $s$ that can set in the assumed HA.

the grid. It can be seen how the preferred setting are correlated for the users with similar user characteristics $u_1$ and $u_2$.

Constructing preference models

To be able to evaluate preference for any setting $s^*$ in any context $c^*$ for each user $j$, we generate preference functions $f_j$ using GP regression on the sampled datasets $D_j$. To that end, we put a prior $GP(0, \kappa^{pref})$ on $f_j$ with the kernel $\kappa^{pref}$ defined as:

$$ \kappa^{pref}([c, s], [c', s']) = \kappa^c_{\text{pref}}(c, c') \cdot \kappa^s_{\text{pref}}(s, s'), $$

(3.3)

where $\kappa^c_{\text{pref}}$ is an RBF kernel with lengthscale $\lambda^c_{\text{pref}} = 0.15$ and $\kappa^s_{\text{pref}}$ is a rational quadratic kernel with lengthscale $\lambda^s_{\text{pref}} = 0.4$ and scale mixture parameter $\alpha = 0.5$.

As we assume that user’s latent preference $f$ for any setting is fixed given complete context (i.e. both observed and hidden context components), we are only concerned with the predictive mean of the trained GP model which can be expressed as:

$$ f_j(c^*, s^*) = \mathbf{k}^{pref}_{\text{GP}} \mathbf{K}^{-1}_{\text{pref}} \mathbf{f}_H, $$

(3.4)

where $(\mathbf{k}^{pref})_p = \kappa^{pref}([c^*, s^*], [c_p, \hat{s}_{jp}])$, $(\mathbf{K}^{pref})_{pq} = \kappa^{pref}([c_q, \hat{s}_{jq}], [c_q, \hat{s}_{jq}])$ and $\mathbf{f}_H = [f_H, \ldots, f_H] \in \mathbb{R}^M$.

In Figure 3.3, we visualize latent preference for setting $s$ for 9 users 'located' on the grid with $u_2 = -4$ and for fixed $c_2 = 0$. 

Finally, in order to facilitate interactions between users and a setting recommendation agent, we define a response model from which noisy feedback of user $j$ for setting $s$ in context $c$ will be drawn. We assume that users provide ratings $y \in \mathcal{Y} = [0, 1]$ for one setting at a time and independently from previous interactions with the agent. A natural choice for the response model is then a Beta distribution:

$$y|c, s, j \sim \text{Beta}(\alpha(c, s, j), \beta(c, s, j)),$$

with parameters:

$$\alpha(c, s, j) = \nu \cdot \mu(f_j(c, s)) \quad \text{and} \quad \beta(c, s, j) = \nu \cdot (1 - \mu(f_j(c, s)))$$

where $\mu : \mathbb{R} \to (0, 1)$ is a function mapping the latent preference $f$ to the mean of the response, and $\nu$ controls the variance of the response. We define $\mu(f) = \Phi(f)$, where $\Phi$ is the cumulative distribution function of the standard normal distribution. We also set $\nu = 10$ for all users. In Figure 3.4 we visualize sample response distributions for different values of latent preference $f$.

### Sampling historical interactions

Having the preference and response models, we generate historical interactions for 500 existing users. To that end, for each user $j \in \mathcal{U}_{\text{old}}$ we independently sample a dataset $\mathcal{D}_j$ of $K = 50$ interactions, $\mathcal{D}_j = \{(c_i, s_i, y_i)\}_{i=1}^K$, with $c_i \sim \mathcal{U}(C)$, $s_i \sim \mathcal{U}(S)$ and $y_i$ sampled from the response model defined in (3.5). This sampling procedure corresponds to the agent having recommended only random settings to the existing users. The new users $\mathcal{U}_{\text{new}}$ start with no past interactions. Our goal is to train a recommendation agent based on the historical data of existing users to provide efficient personalization for the new users.
3.2 Method

In this section we revisit the Bayesian meta-learning framework proposed in Korzepa [43], putting it in the context of learning contextual preferences across many users. We also outline how we use Bayesian optimization to guide the personalization for new users leveraging the probabilistic formulation of the proposed meta-learning method.

We assume that a learning agent can only access the observed features in the simulation environment and define \( u^o \) and \( c^o \) as the observed components of the corresponding user and context vectors, \( u \) and \( c \). The size of \( u^o \) and \( c^o \) vectors will vary depending on the taken assumptions about observability of \( u \) and \( c \) feature vectors.

Model definition

Following Korzepa [43], we define the model that maps a given setting \( s \) and the observed features \( u^o_j \) and \( c^o \) to a predicted response \( y \) of user \( j \) to be the linearization (i.e. first-order Taylor expansion) of a neural network \( h(u^o, c^o, s; \theta_j) \) with parameters \( \theta_j \in \mathbb{R}^p \) at the mean of the prior \( p_{\mu, \Sigma}(\theta_j) = \mathcal{N}(\theta_j | \mu, \Sigma) \), i.e. \( \theta_j = \mu \):

\[
h_{\mu}^{lin}(u^o, c^o, s; \theta_j) = h(u^o, c^o, s; \mu) + J_{\mu}(u^o, c^o, s)(\theta_j - \mu), \tag{3.7}
\]

where \( J_{\mu}(u^o, c^o, s) = \nabla_\theta h(u^o, c^o, s; \theta_j) |_{\theta_j = \mu} \).

The prior parameters \( \mu \) and \( \Sigma \) are shared for all user-specific parameters \( \theta_j \). Our goal is to learn these shared parameters such that the prior combined with the linearized model provides a good personalization performance. Similarly as in Korzepa [43], we restrict \( \Sigma \) to a diagonal structure to limit the computational and space complexity.

In order to apply the Bayesian modelling framework, we need to specify a likelihood function \( p_{\gamma}(y | g^{-1}(h_{\mu}^{lin}(u^o, c^o, s; \theta_j))) \), where \( \gamma \) expresses the set of likelihood hyperparameters (assumed to be shared across all users) and \( g^{-1} \) is the inverse link function mapping the latent prediction \( h_{\mu}^{lin} \) to the mean of the predicted response \( y \). As we assumed that users respond to setting recommendations with a response \( y \in [0, 1] \), a natural choice for the likelihood is Beta distribution. We use the same parametrization as in (3.6) implying that \( \gamma = \{\nu\} \) and \( g^{-1}(\cdot) = \Phi(\cdot) \). We will suppress the inverse link \( g^{-1} \) function in the likelihood notation for the sake of clarity.
This choice of a non-Gaussian likelihood, however, renders the exact Bayesian inference intractable and thus requires using posterior approximations. If the quality of such approximations is not sufficiently good, the meta-training and adaptation performance may be negatively impacted. Also, calculating the approximate posterior may be computationally much more demanding than the computation of the exact posterior with the Gaussian likelihood. For these reasons, we will also perform meta-learning using the Gaussian likelihood. While this likelihood is mismatched with the user response data which may in many ways negatively impact the personalization performance, the lack of approximations may provide some gains in the performance (apart from the clear computational benefits). We parametrize the likelihood with variance $\sigma_y^2$ shared across all users and set the inverse link $g^{-1}$ to the identity function.

**Training objective**

Following Korzepa [43], we define a hierarchical Bayesian model with shared prior parameters $\mu, \Sigma$ and likelihood parameters $\gamma$ that are estimated using maximum likelihood estimation (MLE). Based on the results from Korzepa [43], we choose to optimize the posterior predictive likelihood objective rather than the marginal likelihood. MLE estimation of the shared parameters allows for splitting the logarithm of the objective into a sum of user-specific terms which facilitates stochastic gradient optimization using mini-batching. To evaluate the posterior predictive likelihood, for each user $j \in U_{\text{old}}$ we define a train and test dataset, $D_{tr}^j$ and $D_{te}^j$, such that $D_{tr}^j \cup D_{te}^j = D_j$. We also define $D_{tr}^{old} = \bigcup_{j \in U_{\text{old}}} D_{tr}^j$ and $D_{te}^{old} = \bigcup_{j \in U_{\text{old}}} D_{te}^j$, and $M_{tr}$ and $M_{te}$ to be the sets of interaction indices that belong to the train and test datasets respectively. As the interactions in $D_j$ were sampled randomly and independently for all users, the exact split into train and test interactions other than split proportion will lead to the same results on average; we can also use the same interaction indices for each user. The objective can be then formulated as:

$$p_{\mu, \Sigma}(D_{te}^{old} | D_{tr}^{old}) = \prod_{j \in U_{\text{old}}} p(D_{te}^j | \theta_j)p_{\mu, \Sigma}(\theta_j | D_{tr}^j)d\theta_j$$

$$= \prod_{j \in U_{\text{old}}} \left( \prod_{i \in M_{tr}} p_{\gamma}(y_{ji} | h_{\mu}^{lin}(u_{ji}^o, c_{ji}^o, s_{ji}; \theta_j)) \right) p_{\mu, \Sigma}(\theta_j | D_{tr}^j)d\theta_j,$$

(3.8)

where $y_{ji}$, $c_{ji}^o$ and $s_{ji}$ correspond to the response, observed context and recommended setting in the $i$-th interaction in $D_j$, and $p_{\mu, \Sigma}(\theta_j | D_{tr}^j)$ is the posterior
distribution for $\theta_j$ that can be expressed using Bayes theorem as:

$$p_{\mu, \Sigma}(\theta_j|D_{tr}^j) = \frac{p_{\mu, \Sigma}(\theta_j)p(D_{tr}^j|\theta_j)}{p(D_{tr}^j)} \quad (3.9)$$

$$= \frac{p_{\mu, \Sigma}(\theta_j) \prod_{i \in M^*} p_\gamma \left( y_{ji}|h_{\mu}^{\text{lin}}(u_{ji}^o, c_{ji}^o, s_{ji}; \theta_j) \right)}{\int p_{\mu, \Sigma}(\theta_j) \left( \prod_{i \in M^*} p_\gamma \left( y_{ji}|h_{\mu}^{\text{lin}}(u_{ji}^o, c_{ji}^o, s_{ji}; \theta_j) \right) \right) d\theta_j}$$

For non-Gaussian likelihoods such as the Beta likelihood, this posterior is intractable due to the form of the normalization constant in the denominator. An approximate posterior $q_{\mu, \Sigma}(\theta_j|D_{tr}^j)$ can be computed using e.g. the Laplace approximation as explained in Korzepa [43].

**Predictions**

To predict the response $y$ of user $j$ for a given setting $s$ in observed context $c^o$ given the past interactions $D_j$, we marginalize the parameters $\theta_j$ in the likelihood using the posterior from (3.9):

$$p(y|D_j, u_j^o, c^o, s) = \int p(y|h_{\mu}^{\text{lin}}(u_{ji}^o, c_{ji}^o, s; \theta_j)) p_{\mu, \Sigma}(\theta_j|D_j) d\theta_j \quad (3.10)$$

This gives us a probability distribution of different outcomes $y$ which can be utilized in methods such as Bayesian optimization to efficiently explore settings based on the model’s confidence about a given outcome as will be discussed later.

For non-Gaussian likelihoods such as the Beta likelihood, (3.10) cannot be evaluated analytically. A common solution is to replace the true posterior $p_{\mu, \Sigma}(\theta_j|D_j)$ with the approximate posterior $q_{\mu, \Sigma}(\theta_j|D_j)$ and approximate the integral using Monte-Carlo (MC) sampling.

**Conversion to function space**

Instead of performing Bayesian inference in parametric space, following Korzepa [43] where we exploited the duality of weight and function space formulations
in linear models, we perform inference in function space. This involves using a GP prior of the following form:

\[
h_{\mu}^{\text{lin}}(u^\circ_j, c^\circ_{ji}, s_{ji}) \sim \mathcal{GP}\left(h(u^\circ_j, c^\circ_{ji}, s_{ji}; \mu), J_\mu(u^\circ, c^\circ, s)\Sigma J_\mu(u^\circ, c^\circ, s)^T\right)
\] (3.11)

Inference in function space brings major computational benefits as datasets \(D^\text{tr}_j\) are very small as their are limited by the number of interactions between the user and the agent (leading to \(O(|M^\text{tr}|^3)\) time complexity and \(O(|M^\text{tr}|^2)\) space complexity). This allows to consider full posterior correlations of the parameters \(\theta_j\) even if the size of the original neural network, \(P\), is very large. Performing inference in parametric space would lead to \(O(P^3)\) time complexity and \(O(P^2)\) space complexity which would either require using very small models or resorting to approximations of the posterior with simpler covariance structure.

Contextual Bayesian optimization

When recommending a setting \(s\) to a HA user \(j\) in observed context \(c^\circ\), a personalization agent needs to choose the value of \(s\). To select this value, it uses some acquisition function. Two simple but very different choices are a random acquisition function, i.e. \(s \sim U(S)\), or a function selecting the setting that maximizes the posterior predictive mean, i.e. \(s = \arg \max_{s \in S} \mathbb{E}[y|D_j, u^\circ_j, c^\circ, s]\). The latter choice allows to fully exploit the knowledge learned by the agent and may provide high quality recommendations if the knowledge represented in the posterior is accurate and sufficient. However, if there is lots of ambiguity due to little amount of data or if the model captured some spurious patterns (e.g. due to overfitting to the noise in the data), this approach may lead to repeatedly recommending settings that are far from optimal and it might not be possible to improve the model as the agent will not explore different settings. While this approach might not allow to consistently improve personalization performance in the long run due to lack of exploration, it may offer good performance relatively quickly due to exploitation. We will refer to this acquisition function as ‘greedy’.

The former choice, i.e. the random acquisition function, works in the exactly opposite way. It focuses purely on exploration ignoring exploitation altogether. While this approach may be prohibitively slow to reach good personalization performance (especially when the setting space \(S\) is complex), it is resistant to model misspecification or spurious patterns as it does not use the knowledge captured in the model when choosing which setting to recommend.

A simple way to make a trade-off between the exploration of the unexplored ar-
3.2 Method

...as the setting space and the exploitation of the patterns learned by the agent is to use a so-called $\epsilon$-greedy (epsilon-greedy) approach which switches from the greedy to random acquisition function with $\epsilon$ probability. The $\epsilon$-greedy approach fails, however, to perform exploration in a principled way as it ignores what the model knows and what it does not know. This may lead to a particularly low exploration efficiency if $S$ is complex.

A solution to this problem is offered by Bayesian optimization. Bayesian optimization aims to make a trade-off between exploration and exploitation in a principled way leveraging the knowledge of what is known and what is unknown by the model. Bayesian optimization can use different acquisition functions such as the probability of improvement (PI), the expected improvement (EI) or the upper confidence bound (UCB). Here we focus on the EI acquisition function.

We consider a user $j$ with a dataset $D^i_j$ at interaction $t$ consisting of past interactions of that user. We aim to find setting $s$ for recommendation in a newly observed context $c^o_t$. After putting this into setting contextual personalization context, the EI acquisition function is defined as

$$EI^t_j(s) = \mathbb{E}_{p^t_{j}(y)} \left[ \max(y, 0) \right], \quad (3.12)$$

where $p^t_{j}(y) = p(y|D^t_j, u^o_j, c^o_t, s)$ is the posterior predictive distribution for the response of user $j$ to setting $s$ recommendation at interaction $t$. When $p^t_{j}(y)$ is a Gaussian distribution, the expectation in (3.12) has an analytical solution which can be expressed as

$$EI^t_j(s) = (m^t_j(s) - m^+) \Phi \left( \frac{m^t_j(s) - m^+}{\sigma^t_j(s)} \right) + \sigma^t_j(s) \phi \left( \frac{m^t_j(s) - m^+}{\sigma^t_j(s)} \right), \quad (3.13)$$

where $m^t_j(s)$ and $\sigma^t_j(s)$ are respectively the mean and the standard deviation of the posterior predictive distribution $p^t_{j}(y)$, $\Phi$ and $\phi$ are respectively the cumulative density and the probability density functions of the standard normal distribution, and $m^+$ is calculated as:

$$m^+ = \max \{ m^t_j(s) \}_{(c^o_t, s) \in D^t_j}. \quad (3.14)$$

The first summation term in (3.13) corresponds to exploitation and the second terms corresponds to exploration. Equipped with the EI acquisition function, a single round of Bayesian optimization for user $j$ at interaction $t$ in observed context $c^o_t$ can be written as:

$$\text{...}$$
1. Calculate the posterior predictive distribution based on $D_{jt}^t$.

2. Recommend setting $s_t = \arg\max_{s \in S} EI_{jt}^t(s)$ to the user.

3. Observe the user’s noisy response $y_t$ to the recommended setting.

4. Update the dataset with past interactions with the new interaction $(c_t^o, s_t, y_t)$, i.e. $D_{jt}^{t+1} = D_{jt}^t \cup \{(c_t^o, s_t, y_t)\}$.

The maximization of the acquisition function in step 1 can be performed by evaluating the function on a grid of settings $s$ sufficiently covering the setting space $S$. However, if $S$ is complex, grid-based optimization might be prohibitively expensive. A standard solution is to use gradient-based optimization. Furthermore, it is common to run such optimization with multiple random initializations to increase the chances of finding the global (or a good local) maximum.

For Gaussian likelihoods, the posterior predictive distribution $p_{jt}^*(y)$ is Gaussian and the procedure described above can be applied directly. However, for non-Gaussian likelihoods, $p_{jt}^*(y)$ is not a Gaussian distribution. Bayesian optimization can instead be performed using the approximate Gaussian posterior for latent prediction $h_{jt}^\text{lin}$ before the inverse link $g^{-1}$, i.e. $q(h_{jt}^\text{lin}|D_{jt}^t, u_j^o, c_j^o, s)$ (the functional equivalent of the approximate parametric posterior $q_{\mu,\Sigma}(\theta_j|D_{jt}^t)$).

## 3.3 Experiments

In this section, we describe the experimental procedure, the agents used, the assumed simulation scenarios, the measured performance metrics and the obtained results.

### Meta-training details

We model $h(u^o, c^o, s; \theta_j)$ with a neural network consisting of 2 hidden layers with 50 units, $\text{tanh}$ activation function and a single output corresponding to the predicted response $y$. We run simulations with both Gaussian and Beta likelihoods.

We train each model for 100 epochs where each epoch is defined as updating the training objective with the data of each user $j \in U_{\text{old}}$ exactly once. We use mini-batches of datasets from 16 users. To obtain the posterior predictive distribution for the Beta likelihood for objective calculation or predictions, we
apply the Laplace approximation as described in Korzepa [43]. To evaluate the posterior predictive likelihood objective, we split the historical interactions of users from $U_{old}$ into train and test sets, $M^{tr}$ and $M^{te}$, such that $|M^{tr}| = 30$ and $|M^{te}| = 20$. To evaluate the objective for the Beta likelihood, following Korzepa [43] we use 5 MC samples. We optimize the shared prior parameters $\mu$ and $\Sigma$ and the likelihood parameters $\sigma_y$ (for the Gaussian likelihood) and $\nu$ (for the Beta likelihood) using Adam optimizer with learning rate $1e^{-3}$ for $\mu$ and $5e^{-3}$ for the remaining parameters. The initialization for $\mu$ is random, following the default initialization scheme in the PyTorch library. All variances in the diagonal prior covariance $\Sigma$ are initialized to 0.5.

We evaluate the performance on 200 new users from $U_{new}$. For each user we generate a sequence of $T = 200$ random contexts sampled uniformly from the context space, i.e. for each $i$-th interaction of $j$-th user we have $c_{ji} \sim U(C)$. These contexts will stay constant throughout the whole experiment. Additionally, after the meta-training is finished, we keep the final learned prior fixed and do not update it with the new data acquired during interactions with new users. Consequently, personalization for a new user is only dependent on the past interactions of that user with the agent as well as the prior learned from the data of old users, meaning that the data acquired from a new user does not have an impact on personalization provided to other users.

Agents

In the evaluation phase, we compare 5 agents that differ in terms of how they select settings for recommendation:

- The **EI** agent recommends settings according to the EI acquisition function defined in (3.13).
- The **greedy** agent recommends a setting that maximizes the current posterior predictive mean. This corresponds to pure exploitation of the learned patterns ignoring exploration altogether.
- The **random** agent recommends a random setting $s$ sampled uniformly from $S$.
- The **prior** agent recommends settings maximizing the prior predictive mean. It roughly corresponds to recommending settings using a model trained on the combined datasets of old users without personalizing to individual users. As the prior mean does not change between interactions, this agent always recommends the same setting in a given observed context.
and will have constant performance across all interactions for a given user. The performance may however vary between different trained models.

- The **fixed** agent always recommends \( s = 0 \) which corresponds to the default setting provided in clinical workflows that on average may work well but may be far from optimal for many users and contexts. The performance of this agent depends only on the user preference model and will stay constant across all simulations as all of them assume the same set of users.

We assume that the EI, greedy and random agents can learn by updating the beliefs (i.e. the posterior predictive distribution) about user preferences after each interaction. The prior and fixed agents are considered as static baselines (although one could also calculate the posterior based on the interactions of these agents with the users).

**Simulation scenarios**

We consider three simulation scenarios with different assumptions about observability of user and context features:

- **Scenario A** assumes full observability of both user and context features. This implies that each user vector \( u^o \) consists for \( u_1 \) and \( u_2 \) features and each context vector \( c^o \) consists of \( c_1 \) and \( c_2 \) features.

- **Scenario B** assumes full observability of context features and no observability of user features. This means that no information about user characteristics is revealed to the agents.

- **Scenario C** assumes partial observability of context features and user features. Each user vector \( u^o \) consists only of \( u_1 \) feature and each context vector \( c^o \) consists only of \( c_1 \) feature.

We expect that scenario A will be simplest for the agents (other than the fixed one) which may reach a very good performance even without interacting with the new users due to full observability of the features determining similarity between preferences of different users. Scenario B is more difficult as the agents cannot personalize the recommendations from the start as no knowledge about individual users is exposed to the agents. However, with a sufficient number of interactions, the agents may accurately learn contextual preferences of the users as no information about relevant context is hidden to the agents. Scenario C
might be the hardest task for the agents. Even though the agents have access to partial information about users, the partial observability of context makes it impossible to always recommend an optimal setting. Instead, the agents may only learn to recommend settings that on average work best in a given observed context.

Performance metrics

We measure the personalization performance of the agents using the following two metrics:

- The **mean cumulative regret** metric measures the cumulative regret (i.e. the difference between the optimal preference for a given context and the observed user feedback) of past interactions at interaction \( t \) and is averaged across all users in \( \mathcal{U}_{\text{new}} \). It can be expressed as:

\[
r_t = \frac{1}{|\mathcal{U}_{\text{new}}|} \sum_{j \in \mathcal{U}_{\text{new}}} \sum_{t=1}^{t} \mu(f_{ji}^*) - y_{ji},
\]

where \( f_{ji}^* = \max_{s \in S} f_j(c_{ji}, s) \) is the latent preference of user \( j \) for the optimal setting in context \( c_{ji} \) and \( y_{ji} \) is the response of the user to the setting that was recommended. Agents that perform lots of exploration are expected to score worse on this metric than agents that focus on exploitation. The lower value of this metric, the better.

- The **mean exploitation regret** metric measures the mean regret at interaction \( t \) for a set of contexts \( \mathcal{G} \) for settings selected according to the posterior predictive mean acquisition function (i.e. fully exploiting the learned patterns) averaged across all users in \( \mathcal{U}_{\text{new}} \). The set \( \mathcal{G} \) is defined as a \( 10 \times 10 \) equispaced covering the context space \( \mathcal{C} \). This metric can be expressed as:

\[
l_t = \frac{1}{|\mathcal{U}_{\text{new}}|} \sum_{j \in \mathcal{U}_{\text{new}}} \frac{1}{|\mathcal{G}|} \sum_{c \in \mathcal{G}} \mu(f_{jc}^*) - \mu(f_j(c, \hat{s}_{jc})),
\]

where \( f_{jc}^* = \max_{s \in S} f_j(c_{jc}, s) \) is the latent preference of user \( j \) for the optimal setting in context \( c \), and \( \hat{s}_{jc} \) has a different definition depending on the agent. For the EI, greedy and random agents, \( \hat{s}_{jc} = \arg \max_{s \in S} \mathbb{E}[y|D_{jt}, u_j^o, c^o, s] \) is the setting maximizing the posterior predictive mean function. For the prior agent, \( \hat{s}_{jc} = \arg \max_{s \in S} \mathbb{E}[y|u_j^o, c^o, s] \) is the setting maximizing the prior predictive mean function. For the fixed agent, \( \hat{s}_{jc} = 0 \).
As for the prior and fixed agents this metric does not depend on the past interactions $D_t^j$, it will have constant performance across all interactions. The lower value of this metric, the better.

We calculate the mean and the standard deviation of these metrics from 5 random initializations of the prior parameters $\mu$. We will also refer to these metrics simply as the *cumulative regret* and the *exploitation regret*.

**Results: scenario A**

In this scenario we investigate how the agents perform when all relevant features determining contextual preferences of HA users can be observed. We present the performance as measured by the defined metrics for both likelihoods in Figure 3.5. As expected, the performance of the learning agents and the prior agent is much better than the one of the fixed agent. The full observability of user features

![Graphs showing performance metrics for different likelihoods](image)

**Figure 3.5:** Personalization performance as measured by the mean cumulative regret (as defined in (3.15)) and the mean exploitation regret (in defined in (3.16)) in a simulation scenario with full observability of user and context features.
and context features allows to train a very strong shared prior (in particular its mean function) that can provide very good personalization to users even if they have not provided any data yet. The quality of the prior mean is better for the Gaussian likelihood as it achieved 0.024 lower exploitation regret than the Beta likelihood. However, it can be seen that the Beta likelihood provides much better predictive distributions for applying Bayesian optimization as the exploitation regret improves much faster than in the case of the Gaussian likelihood. In fact, for the Gaussian likelihood, this regret is higher for the agent with the EI acquisition function than for the agent with the random one.

When looking into the cumulative regret, we can see that for the Beta likelihood, the EI agent performs lots of exploration as it incurs higher regret than the greedy or prior agents. This exploration, however, allows to provide better personalization performance in the long run as measured by the exploitation regret. For the Gaussian likelihood, the cumulative regret of the EI agent grows at roughly the same pace as the one of the greedy agent which suggests that using the Gaussian likelihood in this problem results in a considerable underestimation of the predictive variance which in turn almost completely prevents exploration. The random agent naturally incurs the highest cumulative regret as it rarely recommends settings that users prefer. Even though this extensive exploration allows to improve the personalization performance considerably when switched to exploitation, this agent is not applicable in practice as it would bring no benefit to users (or could even annoy them) as long as the exploration is ongoing. Finally, the small difference in the exploitation regret between the EI agent and the random agent for the Beta likelihood may be the result of underestimation of the predictive variance leading to insufficient exploration or its overestimation leading to excessive exploration.

Results: scenario B

In this scenario we investigate how the agents perform when they cannot access any relevant user characteristics but can fully observe the relevant context. In Figure 3.6 we present the results analogical to the ones presented in the previous scenario. As the model does not account for the latent user features, the prior mean can only model the preferences for a given context averaged across all users that were participating in the meta-training phase. Consequently, when looking at the Beta likelihood, the exploitation regret for the prior agent is only 0.013 lower than for the fixed agent while this difference was as much as 0.082 in scenario A in which the user features were fully observable. While the prior cannot provide good personalization for the new users from the start, it offers lots of space for improving personalization by interacting with them. Similarly as in scenario B, the performance of the EI agent is much better for
the Beta likelihood than the Gaussian one. The exploitation regret for the EI agent improves by as much as 0.057 after 200 interactions. The cumulative regret indicates similar behaviour of the EI agent for the two likelihoods as was observed in scenario A.

Results: scenario C

In this scenario we investigate how the agents perform when they can only partially access the relevant user and context features. We present the results in Figure 3.7. It can be seen that obscuring one of the context features has a big negative impact on the performance of all agents other than the fixed agent. The initial exploitation regret for these agents for the Beta likelihood is higher than for the fixed agent. In combination with the large variance between different initialization, this may suggest that some of the trained models did not properly converge within the scheduled 100 epochs. When looking at the
3.4 Discussion

In this chapter we demonstrated the proposed probabilistic meta-learning method in a number of simulations with different observability assumptions. We carried out simulations for the Beta and the Gaussian likelihoods and compared different acquisition functions for selecting settings for recommendations.

Even though the initial performance (before interacting with new users) was slightly worse for the Beta likelihood than for the Gaussian one, it was the former that provided a much stronger basis for improving personalization by further interactions with Bayesian optimization, most likely due to a much better

![Figure 3.7:](image)

EI agent it can be seen that while it has the exploitation regret for the Beta likelihood 0.020 higher than for the Gaussian one, this gap is completely closed after 200 interactions. This again indicates that the Beta likelihood provides a much better basis for Bayesian optimization in this problem.

3.4 Discussion

EI agent it can be seen that while it has the exploitation regret for the Beta likelihood 0.020 higher than for the Gaussian one, this gap is completely closed after 200 interactions. This again indicates that the Beta likelihood provides a much better basis for Bayesian optimization in this problem.
uncertainty quantification. Nonetheless, the difference between the random and the EI acquisition functions was relatively small. It is likely that this was the result of excessive exploration in the EI agent due to overestimation of the predictive variance in the posterior distribution approximated using the Laplace approximation as discussed in Immer et al. [33]. Nonetheless, if the complexity of the setting space $S$ was bigger, the difference between the EI and the random agents would be most likely much larger.

Furthermore, it was very difficult for the agents to learn in the presence of latent context. As the agents could not access the hidden context feature $c_2$, they could at best learn to recommend the setting that on average works best in a given observed context. When looking at the simulated preferred settings in Figure 3.2, it can be seen that they greatly differ across the hidden context feature $c_2$ which explains why this scenario was so challenging for the agents. This could be potentially addressed by explicitly conditioning the model on auxiliary latent features and learning them from the data. Similarly, one could model latent user features to facilitate capturing more complex preferences in the prior. These directions will be further discussed in Section 4.3.

The obtained results indicate potential for applying the proposed probabilistic meta-learning method for personalizing HA settings. Not only it could provide a way to capture complex preferences leveraging the flexibility of NNs, but it could also provide GP-like uncertainty needed for trading-off exploration and exploitation. Furthermore, as this method uses NN-like gradient-based updates to learn across users, it can potentially scale to thousands or even millions of users and may also facilitate privacy-protecting learning thanks to the applicability of federated learning methods. On the other hand, the GP formulation used in inference within individual users can facilitate performing sequences of setting recommendations at very short intervals within a single session of interactions between a user and the agent.

The conducted experiment also demonstrates the potential for using simulation environments for a systematic and scalable evaluation of HA setting personalization solutions as well as guiding their development. The carried out simulations allowed to compare different personalization agents and investigate their performance in a number of conditions with varying observability of user and context features.
In this thesis we set out with three objectives that addressed different aspects of understanding and modeling contextual preferences of HA users:

- Explore the impact of real-world context on auditory preferences of HA users.
- Simulate auditory preferences based on observable and latent user and context features.
- Develop machine learning methods that can efficiently learn contextual preferences from limited amounts of noisy user feedback.

In Sections 4.1 to 4.3 we summarize the key findings and contributions made with respect to these objectives, discuss the limitations of the conducted research and provide an outlook for future directions. In Section 4.4 we provide broader perspectives for hearing health care that the contributions made in this thesis may open.
4.1 Exploring context and preferences

We conducted a user study in which hearing-impaired participants interacted with their HAs in daily-life environments [45]. The study showed that HA users may have preferences for HA settings that are very different from the ones provided in standard clinical settings and that these preferences may vary not only between but also within individual users depending on the context.

The main limitation of the conducted study was its small scale as it involved only two participants. While it was evident from the obtained data that there are both inter- and intra-individual variations in contextual preferences of HA users, a larger study would need to be run to quantify the degree of these variations in a broader population of HA users. Furthermore, in this study the participants could only choose between four fixed programs due to technical constraints of the HAs used in the study. This means that they were not able to fully explore the space of the HA settings and there could be settings that they would prefer over the ones provided in the study. Additionally, every time the HAs were turned on, they started in a specific default program which creates a bias in the usage statistics of the provided programs. Finally, the two participants did not receive the same set of programs which makes a direct comparison difficult.

In Korzepa et al. [46] we argued which types of context can be potentially helpful in discriminating between situations in which listening needs of a HA user might differ, and thus facilitate better personalization to individual auditory preferences. It remains to be verified how useful the discussed contexts would be in practical settings. Moreover, the list of considered context types is not exhaustive and there might be many other sources of contextual information that are relevant e.g. biometric data such as heart rate which was shown to correlate with the difficulty of the listening environments and may indicate increased listening effort [12].

We believe that an important direction for future research is to empirically identify the types of context that are most relevant for personalization of HA settings. A potential approach would be to run a larger study that focuses on collecting information about individual situations in which HA users have a need to adjust the HA settings e.g. by prompting user to provide a description of the situation and the experienced difficulties or tracking a wide range of contexts and finding which ones most frequently co-occur with HA setting adjustments.
4.2 Simulating contextual preferences

In Korzepa et al. [44] we proposed a GP-based approach to simulate contextual preferences of HA users incorporating aspects such as correlations of preferences across users, complexity of preferences, partial observability of context and user features, and noisiness of perceptual feedback. This approach allows to circumvent the limitations of traditional user studies and enables a more scalable approach to run multiple large-scale experiments validating different aspects of preference learning solutions such as speed of learning in the presence of limited data, accountability for latent factors or leveraging similarities across users. We also demonstrated how the proposed simulation framework can be used to evaluate and compare different agents for HA setting personalization in a number of simulations with different assumptions about the observability of user and context features.

One limitation of the proposed approach relates to drawing samples from a Gaussian distribution with a Kronecker-factored covariance whose Kronecker factors correspond to context and user covariances. This implies that preferences for each user will follow the same assumptions about complexity and sensitivity to observed and hidden context features. In practice, however, these aspects are likely to vary across HA users, e.g. some users may have very diverse preferences dependent on the context while some others may prefer roughly the same settings independent of the situation they are in. Also, when simulating preferences for a large number of users and contexts, the covariance matrix is too large to draw samples exactly and it needs to be approximated with a simpler one, e.g. using a low-rank decomposition. Due to such approximation, the samples may not accurately reflect the assumptions introduced in the structure of the original covariance matrix.

Furthermore, in the proposed simulation framework we considered context defined in low-dimensional continuous space. This could for example correspond to low dimensional representation of an acoustic scene. In practical settings we might also have categorical context features arising from classification of acoustic scene, activity, etc. A potential approach to model them in a simulation framework would be to discretize continuous features.

In our research we focused on simulating contextual preferences of HA users to facilitate systematic and scalable evaluation of contextual personalization algorithms for HA users. A natural direction for future work is to extend the simulation environment to model more aspects characteristic for the HA domain that could play an important role in the development of well-performing HA personalization systems. For example, HA users may have a different level of engagement dependent on factors such as personality, interest in new technolo-
gies or obtrusiveness of an agent they interact with. Implementing such aspects in the simulation environment could help to develop agents that can decide in which situations it is most beneficial to interact with the user to learn their preferences while keeping them engaged. Similarly, we could consider modeling the inherent noise in user responses in more detail. It has been shown that the ability to consistently choose the preferred setting is dependent not only on the perceptual differences in the evaluated settings but also varies across HA users and listening conditions they are in \cite{38, 99}. Incorporating these aspects in the simulation could help to develop algorithms that can quantify the context- and user-dependent noise present in perceptual feedback users provide and adapt the granularity of recommended settings or even the type of response a user is expected to provide (e.g. binary vs continuous).

In our simulation environment we considered generic context and user features without specifying any model for generating them. In future work we could develop a generative model for sampling users that mirror some characteristics and similarities observed in real HA users e.g. based on audiological user profiles \cite{82, 83}. Similarly, it could be beneficial to have a structured approach to generate user-dependent context trajectories to mimic the characteristics of real-world context as well as transitions between different listening situations. This could be modelled e.g. using a Bayesian network trained on longitudinal context data collected in real user studies \cite{11}.

### 4.3 Learning contextual preferences

In our research we focused on Bayesian modelling in neural networks exploring their connections with Gaussian processes. In particular, we developed a novel probabilistic meta-learning method approach that provides a framework for learning contextual preferences of HA users from limited amounts of user feedback. We demonstrated the use of this meta-learning framework combined with Bayesian optimization in a number of simulations with different assumptions about the observability of user and context features.

The simulation environment we considered had a small, one-dimensional setting space. This allowed us to evaluate different acquisition functions on a grid of settings and select the setting that maximizes such a function. This approach is not however a viable option when the dimensionality of the setting space is large. A common solution in such scenario is to apply gradient-based optimization to find the setting that maximizes a given acquisition function. Furthermore, in order to increase the chances of finding the global (or a good local) maximum, the optimization can be run from different initializations.
Following Korzepa \cite{43}, we employed the Laplace approximation to facilitate Bayesian inference with non-Gaussian likelihoods in the proposed meta-learning method. This choice of approximation appeared to overestimate the variance of the predictive distribution, most likely leading to excessive exploration. This observation is in line with our previous research \cite{33}. A potential solution would be to replace the Laplace approximation with variational inference which may provide better predictive performance and better quantification of the uncertainty \cite{33}.

In the conducted simulations, we trained the prior once based on the data from old users and performed personalization for the new users always using the same prior. This however does not allow to improve the personalization provided to a new user based on the data acquired from other new users. In practice, it would be reasonable to update the prior at some intervals when interacting with new users to better utilize the growing data (which would be particularly important in the early stage of running a personalization system when the acquired data is limited).

In the proposed method we did not explicitly account for latent user or context features. Modeling latent user features could help to capture in the learned prior a diversity of different preferences in a given context rather than averaging them across users. This approach would allow to infer latent user features when some data for a new user is observed and thus exploit the correlations with other users having similar values of the latent features. Modeling latent context features could potentially help to recommend multiple settings that a user might prefer in a given observed context. Some approaches for modeling latent features were discussed in Damianou et al. \cite{16} in the context of GPs.

In the simulations, we focused on evaluating the proposed meta-learning method in a simulation environment but did not compare to other methods that could be applied to this problem. Probably the most natural candidate would be multi-task GPs \cite{7}. This method would combine all data from all users and perform GP inference with a kernel that is a Hadamard product of a kernel operating on context and settings features and a kernel operating on user-specific features. Such an approach would have however a few drawbacks. Firstly, it would not scale as hyperparameter updates and evaluation of the predictive distribution would involve $O(M^3 N^3)$ time and $O(M^2 N^2)$ time complexity where $N$ is the number of training users and $M$ is the average number of interactions per user. Secondly, this approach requires access to all data which would not allow for privacy-preserving learning. The first limitation could be potentially alleviated using scalable GP approximations with inducing points \cite{54}. Another potential approach is to use a hierarchical GP model with GP priors with a mean parametrized by a shared NN and a standard kernel such as an RBF kernel, and jointly learn the NN’s parameters and the kernel hyperparameters.
Finally, it also remains to be seen how the proposed method would perform in real-world settings.

### 4.4 Broader perspectives for hearing health care

Instead of viewing HAs as stand-alone devices as it has been traditionally done in the HA industry, they could be considered as a part of an ecosystem allowing HA manufacturers to leverage the value of data crowd-sourced from millions of devices. This value could go far beyond improving personalization for individual users based on shared preference patterns. Through mining the contextual preference data collected from a large number of users, HA manufacturers could learn to what extent their devices are able to provide support to users in different situations. Knowing in which situations the capabilities of the devices are not sufficient to satisfy users’ expectations could help to set out directions for future improvements in the design of the devices.

Contextual preference data collected from thousands or millions of users could also provide valuable audiological insights about how different groups of users perceive sounds and what kind of hearing loss compensation is needed to provide them benefit. Furthermore, if within a certain group of users there are some clear preference patterns that do not correlate with any observed user characteristics, an interesting opportunity would be to recruit some of these users for a user study which would attempt to identify if there are any underlying characteristics that these users share that could explain the observed correlation in preferences. Studies like this could potentially provide new insights about hearing loss as well as identify new patient profiling methods that could be implemented in clinical workflows to provide a basis for faster personalization.

Considering smartphone as an integral extension of a HA and its functionalities could even potentially redefine the HA industry from being hardware-oriented to providing value through software run on smartphone. While it takes around three to four years between major releases of new hardware in HA industry, releasing new software-driven capabilities or updating the existing ones could be performed on a much more regular basis. The value could be determined by what actually makes a difference to the user, for example, in terms of extent of contextual personalization, degree of dynamic adaptability or even additional speech and sound enhancements powered by deep learning models run on a smartphone.

Furthermore, smartphone-based ecosystem could provide a research and testing platform enabling HA manufacturers to test different methods or answer specific
research questions by inviting groups of users to try out modified versions of
the software or new experimental features. Such a platform would allow to
run user studies at a large scale, low cost and whenever needed which could
greatly contribute to the development of [HA] solutions that perform well in real
settings and provide most value to [HA] users. For example, [HA] manufacturers
could validate in real settings and compare different promising solutions (e.g.
personalization algorithms developed in a simulation environment or new deep
learning methods for speech or sound enhancement) or explore different ways
of interacting with users to find what kind of interfaces are most convenient or
engaging for what kind of users.
Conclusion

In this project, we addressed the problem of contextual personalization of HA settings in real listening environments.

First, we conducted an in-the-wild user study in which we observed how two hearing-impaired participants switched between four contrasting HA programs in real-world environments. We found out that the participants consistently selected and used programs that considerably differed from the default settings that are commonly provided in clinical workflows. Furthermore, the participants selected different programs depending on the context. These results confirmed the need for contextual personalization in the HA domain.

Secondly, we proposed a simulation-driven approach for facilitating a systematic evaluation and agile development of HA contextual personalization systems. In particular, we developed a Gaussian process based method for simulating contextual preferences of HA users incorporating aspects such as correlations of preferences across users, complexity of preferences, partial observability of context and user features inducing these preferences, and noisiness of perceptual user feedback.

Thirdly, inspired by parallel research that explored connections between Bayesian neural networks and Gaussian processes, we proposed a novel probabilistic meta-learning method and demonstrated how it can serve as a framework for contex-
tual personalization of HA settings. The results indicated potential both for using the probabilistic meta-learning framework to facilitate contextual personalization in the HA domain and for evaluating and comparing HA personalization solutions in simulated environments.

We hope that the proposed methods will encourage further research into contextual personalization of HA settings.
Learning preferences and soundscapes for augmented hearing

Maciej Jan Korzepa, Benjamin Johansen, Michael Kai Petersen, Jan Larsen, Jakob Eg Larsen, and Niels Henrik Pontoppidan. Learning preferences and soundscapes for augmented hearing. *Proceedings of Intelligent User Interfaces*, 2068, 2018. ISSN 16130073
Learning preferences and soundscapes for augmented hearing

Maciej Jan Korzepa  
Technical University of Denmark  
Lyngby, Denmark  
mjko@dtu.dk

Benjamin Johansen  
Technical University of Denmark  
Lyngby, Denmark  
benjoh@dtu.dk

Michael Kai Petersen  
Eriksholm Research Center  
Snekkersten, Denmark  
mkpe@eriksholm.com

Jan Larsen  
Technical University of Denmark  
Lyngby, Denmark  
janla@dtu.dk

Jakob Eg Larsen  
Technical University of Denmark  
Lyngby, Denmark  
jaeg@dtu.com

Niels Henrik Pontoppidan  
Eriksholm Research Center  
Snekkersten, Denmark  
npon@eriksholm.com

ABSTRACT
Despite the technological advancement of modern hearing aids (HA), many users abandon their devices due to lack of personalization. This is caused by the limited hearing health care resources resulting in users getting only a default ‘one size fits all’ setting. However, the emergence of smartphone-connected HA enables the devices to learn behavioral patterns inferred from user interactions and corresponding soundscape. Such data could enable adaptation of settings to individual user needs dependent on the acoustic environments. In our pilot study, we look into how two test subjects adjust their HA settings, and identify main behavioral patterns that help to explain their needs and preferences in different auditory conditions. Subsequently, we sketch out possibilities and challenges of learning contextual preferences of HA users. Finally, we consider how to encompass these aspects in the design of intelligent interfaces enabling smartphone-connected HA to continuously adapt their settings to context-dependent user needs.

ACM Classification Keywords
H.5.2 Information Interfaces and Presentation (e.g. HCI): User Interfaces—User-centered design; K.8.m Personal Computing: Miscellaneous

Author Keywords
personalization; augmented hearing; intelligent interfaces

INTRODUCTION
Even though hearing loss is one of the leading lifestyle causes of dementia [11], up to one quarter of users fitted with hearing aids (HA) have been reported not to use them [5]. One of the reasons behind the prevalence of non-use of fitted HA is identified by McCormack et al. [12] as users feeling that they do not get sufficient benefits of HA. However, in the light of technological advancement of HA as well as the abundance of research indicating clear benefits of HA usage, we rather seek the source of the problem in the lack of personalization in the current clinical approach. The increasing number of hearing-impaired people [6] and lack of hearing health care resources often results in users getting a ‘one size fits all’ setting and thus not exploiting the full potential of modern HA.

Furthermore, the current clinical approach to measure hearing loss is based on pure tone audiogram (PTA). PTA captures the audible hearing thresholds in frequency bands usually from 250 Hz to 10 kHz. However, PTA does not fully explain a hearing loss. Killion et al. showed that the ability to understand speech in noise may vary by up to 15 dB difference in Signal-to-Noise ratio (SNR) for users with a similar hearing loss [8]. Likewise, users differ in terms of how they perceive loudness. Le Goff showed that speech at 50dB can be interpreted either as moderately soft or slightly loud [9]. This means that some users may perceive soft sounds as noise which they would rather attenuate than amplify. These aspects are rarely taken into account in current clinical workflows.

Earlier research by Dillon et al. [3] indicated potential benefits of customization both within and outside the clinic including fewer visits to clinics, a greater choice of acoustic features for fitting and end users’ feeling of ownership. Previous studies that focused on customizing the settings of devices based on perceptual user feedback [13] or using interactive tabletops in the fitting session [2] indicate that users prefer such customization. Aldaz et al. [1] used reinforcement learning to personalize HA settings based on auditory and geospatial context by prompting users to perform momentary A/B listening tests. However, only with the recent introduction of smartphone connected HA like the Oticon Opn [15], it has become possible to go beyond ecological momentary assessment by continuously tracking the users’ interactions with the HA and thereby learn individual coping strategies from data [7]. Such inferred behavioral patterns may provide a foundation for
correlating user preferences with the corresponding auditory environment and potentially enable continuous adaptation of HA settings to the context.

When interpreting user preferences, one needs to consider how the brain interprets speech. Auditory streams are bottom-up processes fused into auditory objects, based on spatial cues related to binaural intensity and time difference [4, 10, 14, 16]. However, separating competing voices is a top-down process, applying selective attention to amplify one talker and attenuate others. HA may mimic this top-down process by either 1) increasing the brightness to enhance spatial cues facilitating focusing on specific sounds or 2) improve the signal to noise ratio by attenuating ambient sounds to facilitate better separation of voices. Incorporating these aspects into our experimental design, we hypothesize we could learn top-down preferences for brightness or noise reduction based on HA program and volume adjustments combined with bottom-up sampling of how HA perceive the auditory environment in terms of sound pressure level, modulation and signal to noise ratio. This allows us to assess in which listening scenarios the user relies on enhanced spatial cues provided by omnidirectionality with more high frequency gain to separate sounds and in which environments the user instead reduces background noise to selectively allocate attention to specific sounds.

In our pilot study, we give two subjects HA programmed with four contrasting programs in terms of brightness and noise reduction, and register how they interact with programs and volume over a period of 6-7 weeks. The purpose of this work is to:

- show how the subjects interact with HA settings in real environments without any intervention,
- discover basic contextual preferences for the subjects,
- identify possibilities and challenges of learning contextual preferences of HA users,
- suggest application of intelligent user interfaces that would continuously support users in optimizing their HA not only by learning and adjusting to individual preferences but also exploiting crowd-sourced patterns.

**METHOD**

**Participants**

Two male participants (from a screened population provided by Eriksholm Research Centre) volunteered for the study (Table 1). The participants suffer from a symmetrical hearing loss, ranging from moderate to moderate-severe as described by the WHO [17]. All test subject signed an informed consent before the beginning of the experiment.

<table>
<thead>
<tr>
<th>Subject</th>
<th>Program</th>
<th>Mode</th>
<th>Brightness</th>
<th>Soft Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>P1</td>
<td>omnidirectional</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>P2</td>
<td>omnidirectional</td>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td></td>
<td>P3</td>
<td>low noise reduction</td>
<td>+2</td>
<td>+2</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>high noise reduction</td>
<td>-2</td>
<td>-2</td>
</tr>
</tbody>
</table>

Table 1: Demographic information related to the subjects.

**Apparatus**

The subjects were fitted with a pair of research prototype HA EVOTION extending Oticon Opn. The subjects used Android 6.0 or iOS 10, connected via Bluetooth. Data was logged using the nRF connect app and shared via Google Drive.

**Procedure**

Based on the individual hearing loss, the subjects were fitted with 4 programs as shown in Table 2. For all programs, HA volume could be adjusted to one of the levels from $-8 \cdots +4$, where 0 is the default volume. The subjects were instructed to explore different settings using HA buttons over a period of 6-7 weeks. In the experimental setup, the HA always start up in the default program and volume. The default program for subject 1 was P2 in the first five weeks which was then switched to P1 for the last two weeks at the subject’s request. Subject 2 used P2 as the default program.

**Soundscape data**

To create an interpretable representation of the auditory features defining the context, we applied k-means clustering to the acoustic context data collected from HA. The values comprise auditory features defining how the HA perceive the acoustic environment:

- **sound pressure level** measure of estimated loudness,
- **noise floor** tracking the lower bound of the signal,
- **modulation envelope** tracking the peaks in the signal,
- **modulation index** estimated as difference between modulation envelope and noise floor,
- **signal to noise ratio** estimated as difference between sound pressure level and noise floor.

The above parameters are captured as a snapshot across multiple frequency bands once per minute. Additionally, the HA perform a rough classification of the auditory environment and represent it as a categorical variable with one of the following values: 'quiet', 'noise', 'speech in quiet', and 'speech in noise'. These labels are used as ground truth for evaluating the performance of the clustering by means of normalized mutual information (NMI) score. The optimal number of clusters was estimated to be 4 with $NMI = 0.35$.

![Figure 1: Applying k-means algorithm to the soundscape data captured from the HA results in four clusters which estimate the acoustic context as C1 'quiet', C2 'speech in noise', C3 'clear speech' or C4 'normal speech'.](image-url)
The resulting four soundscape clusters were labeled according to the proportion of samples with different ground-truth labels within each cluster (Figure 1) while ambiguities were solved by examination of the cluster centroids. The first cluster mainly captured the ‘quiet’ class which is also validated by the cluster centroid having very low values of sound pressure level and noise floor. Thus, the environments assigned to this cluster will be represented as ‘quiet’. The second cluster captured both ‘speech in noise’ and ‘noise’ classes which suggests that the numerical representations of these environments are similar. For simplicity, we label them as ‘speech in noise’. The third and fourth cluster both captured mainly ‘speech in quiet’ with a small addition of other classes. As the third cluster captured samples with much higher sound pressure level and signal to noise ratio, it will be labeled as ‘clear speech’, while the fourth cluster with attributes of the samples closer to mean will be represented as ‘normal speech’.

RESULTS
We refer to the user’s selected volume and program choice as user preferences, and to the corresponding auditory environment as the context. Juxtaposing user preferences and the context allows us to learn which HA settings are selected in specific listening scenarios. To facilitate interpretation we assign each cluster a color from white to green gradient, in which increasing darkness correspond to increased noise in the context (quiet → clean speech → normal speech → speech in noise). Likewise, we assign each program a color from yellow to red gradient. Lighter colors define programs with an omnidirectional focus and added brightness. Darker colors indicate increasing attenuation of noise. This coloring scheme will apply throughout the paper.

Contextual user preferences
Figure 2 shows the user preference and context changes for both subjects, plotted across the hours of the day over the weeks constituting the full experimental period. Subject 1 most frequently selects programs which provide an omni-
Subject 2 mainly selects two programs; P1 offering an omnidirectional focus with added brightness and P2 (default) providing slight attenuation of ambient sounds. Compared to subject 1, this user spends more time in 'quiet' context. Comparing weekdays to weekends, the latter seem to contain a larger contribution of 'normal speech' and 'speech in noise' auditory environments.

Figure 3, illustrates subjects’ average usage of their HA and which programs are used most throughout the day. Days without any HA usage are excluded from the average. The HA usage for subject 1 steadily increases in the morning and early afternoon and peaks at around 4pm. P1 and P2 are the most used programs throughout the day. Interestingly, in the evening, P3 is used more frequently reaching similar usage level as P1 and P2 between 11pm and midnight. P4 is used very rarely yet consistently throughout the day. The HA usage of test subject 2 is shifted towards the morning with peak activity around 2pm. The default P2 is the most commonly used program throughout the whole day. However, during afternoon, P1 seems to be chosen more often.

Figure 4 shows which contexts the subjects use their HA at different times of the day. The HA usage for subject 1 is dominated by speech-related contexts most of the day. Only after 5pm, the context has more ‘quiet’ and ‘clear speech’ and less ‘speech in noise’ contribution. From 9pm, the ‘quiet’ context rapidly overtakes context containing speech. Subject 2 appears to be exposed to different contextual patterns. Normal and noisy speech contexts seem to be dominated by ‘quiet’ soundscapes in the morning. Subsequently, their contributions increase and peak around 7pm. Afterwards, the ‘quiet’

Subject 1 switches to P4 mainly in 'speech in noise' context (twice as often as in 'normal speech'). The fact that 'speech in noise' is a less common environment than 'normal speech' strengthens this behavioral pattern. This suggests that subject 1 seems to cope by suppressing noise in challenging listening scenarios. Examples of this behavioral pattern are illustrated in Figure 5. Likewise, a clear behavioral pattern can be seen for subject 2. P1 is the preferred program in 'speech in noise' environments. Considering that P1 offers maximum brightness and omnidirectionality with reduced attenuation and noise reduction, this behavioral pattern suggests the user compensates by enhancing high frequency gain as a coping strategy in complex auditory environments (examples in Figure 6).

Table 3: Counts of changes to a given program in different contexts over day for subject 1 (top) and 2 (bottom).

<table>
<thead>
<tr>
<th>Context</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUIET</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>CLEAN SPEECH</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>NORMAL SPEECH</td>
<td>10</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>SPEECH IN NOISE</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 4 shows the number of volume changes for subject 2 (subject 1 rarely changes volume). All increases beyond
the default volume level (0) were made in 'speech in noise' context. On the other hand, changes to the default volume were evenly distributed across all contexts. This suggests that increasing the volume is another coping strategy for subject 2 in more challenging listening scenarios.

Table 4: Counts of changes to a given volume in different contexts for Subject 2.

<table>
<thead>
<tr>
<th>Context</th>
<th>Subject 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>QUIET</td>
<td>2</td>
</tr>
<tr>
<td>CLEAN SPEECH</td>
<td>2</td>
</tr>
<tr>
<td>NORMAL SPEECH</td>
<td>2</td>
</tr>
<tr>
<td>SPEECH IN NOISE</td>
<td>12</td>
</tr>
</tbody>
</table>

Figure 7 shows a behavioral pattern that might be more difficult to interpret based on the auditory context alone. Occasionally, subject 1 selects P3 in a 'quiet' environment late in the evening. The test subject subsequently reported that these situations occur when going out for a walk and wanting to be immersed in subtle sounds such as rustling leaves or the surf of the ocean. The preference for P3 thus implies both increasing the intensity of soft sounds as well as the perceived brightness.

**DISCUSSION**

**Inferring user needs from interaction data**

Empowering users to switch between alternative settings on internet connected HA’s, while simultaneously capturing their auditory context allows us to infer how users cope in real life listening scenarios. To the best of our knowledge, this has not been reported before.

Learning the mapping between preferences and context is a non-trivial task, as the chosen settings might not be the optimal ones in the context they appear in. For example, looking into the soundscape data, it is clear that the environment soundscape frequently changes without the user responding with an adjustment of the settings. Conversely, the auditory environment may remain stable whereas the user changes settings. We need to take into consideration not only the auditory environment but also the user’s cognitive state due to fatigue or intents related to a specific task. Essentially, the user cannot be expected to exhibit clear preferences or consistent coping strategies at all times. We hypothesize that many reasons could explain why the user does not select an alternative program although the context changes:

- being too busy to search for the optimal settings,
- too high effort is required to change programs manually,
- accepting the current program as sufficient for the task at hand,
- cognitive fatigue caused by constantly adapting to different programs.

Similarly, we observe situations in which user changes settings even though the auditory environment remain stable, which could be caused by:

- the user trying out the benefits of different settings,
- cognitive fatigue due to prolonged exposure to challenging soundscapes
- the auditory environment not being classified correctly

In our pilot study, the context classification was limited to the auditory features which are used for HA signal processing. However, smartphone connectivity offers almost unlimited possibilities of acquisition of contextual data. Applying machine learning methods such as deep learning might facilitate higher level classification of auditory environments. Different types of listening scenarios might be classified as ‘speech in noise’ when limited to parameters such as signal to noise ratio or modulation index. In fact, these could encompass very different listening scenarios such as an office or a party where the user’s intents would presumably not be the same. Here the acoustic scene classification could be supported by motion data, geotagging or activities inferred from the user’s calendar to provide a more accurate understanding of needs and intents.

Nevertheless, in some situations as illustrated in Figure 6, the behavioral patterns seem very consistent; the user preferences appear to change simultaneously with the context, remain unchanged as long as the context remains stable, and change back when the context changes again. Identifying such behaviors could allow to reliably detect user preferences with
limited amount of user interaction data. Furthermore, time as a parameter also highlights patterns as illustrated in Figure 6 related to activities around lunch time, or late in the evening (Figure 7), as well as the contrasting behavior in weekends versus specific weekdays.

Even though our study was limited to only two users, we identified evident differences in the HA usage patterns. Subject 1 tends to use the HA mostly in environments involving speech, whereas subject 2 spends substantial amount of time in quiet non-speech environments. This might translate into different expectations among HA users. Furthermore, our analysis suggests that users apply unique coping strategies in different listening scenarios, particularly for complex ‘speech in noise’ environments. Subject 1 relies on suppression of background noise to increase the signal to noise ratio in challenging scenarios. Subject 2 responds to speech in noise in a completely different way - he chooses maximum omnidirectionality with added brightness and increased volume to enhance spatial cues to separate sounds. These preferences are not limited to challenging environments but extends to the ambience and overall quality of sound, as subject 1 reported that he enhances brightness and amplification of quiet sounds to feel immersed in the subtle sounds of nature. We find this of particular importance as it indicates that users expect their HA not only to improve speech intelligibility, but in a broader sense to provide aspects of augmented hearing which might even go beyond what is experienced by normal hearing people.

Translating user needs into augmented hearing interfaces

We propose that learning and addressing user needs could be conceptualized as an adaptive augmented hearing interface that incorporates a simplified model reflecting the bottom-up and top-down processes in the auditory system. We believe that such an intelligent auditory interface should:

- continuously learn and adapt to user preferences,
- relieve users of manually adjusting the settings by taking over control whenever possible,
- recommend coping strategies inferred from the preferences of other users,
- actively assist users in finding the optimal settings based on crowdsourced data,
- engage the user to be an active part in their hearing care.

Such an interface would infer top-down preferences based on the bottom-up defined context and continuously adapt the HA settings accordingly. This would offer immense value to users by providing the optimal settings at the right time, dependent on the dynamically changing context. However, the system should not be limited to passively inferring intents, but rather incorporate a feedback loop providing user input. We see a tremendous potential in conversational audio interfaces as HAs resemble miniature wearable smartspeakers which would allow the user to directly interact with the device, e.g. by means of a chatbot or voice AI. First of all, such an interface might resolve ambiguities in order to interpret behavioral patterns. In a situation when user manually changes the settings in a way that is not recognized by the learned model, the system could ask for a reason in order to update its beliefs. Ideally, questions would be formulated in a way allowing the system to directly learn and update the underlying parameters. This could be accomplished by validating specific hypotheses that refer to the momentary context as well as the characteristics captured in the HA user model, incorporating needs, behavior and intents; e.g. ‘Did you choose this program because the environment got noisy / you are tired / you are in a train?’

Secondly, a voice interface could recommend new settings based on collaborative filtering methods. Users typically stick to their preferences and may be reluctant to explore available alternatives although they might provide additional value. Similarly, in the case of HA users, preferred settings might not necessarily be the optimal ones. Applying clustering analysis based on behavioral patterns, we could encourage users to explore the available settings space by proposing preferences inferred on the basis of ‘users like me, in soundscapes like this’. For instance, the interface could say: ‘Many users which share your preferences seem to benefit from these settings in a similar context - would you like to try them out?’ This would encourage users to continuously exploit the potential of their HA to the fullest. Additionally, behavioral patterns shared among users, related to demographics (e.g. age, gender) and audiology (e.g. audiogram) data, could alleviate the cold start problem in this recommender system, thus enabling personalization to kick in earlier even when little or even no HA usage data is available.

Lastly, users should be able to communicate their intents, as the preferences inferred by the system might differ from the actual ones. In such scenarios, users could express their intents along certain rules easily interpreted by the system (e.g. ‘I need more brightness.’) or indicate the problem in the given situation (e.g. ‘The wind noise bothers me.’). Naturally, translating the user’s descriptive feedback into new settings is more challenging, but could potentially offer huge value by relieving users of the need to understand how multiple underlying audiological parameters influence the perceived outcome.

Combining learned preferences and soundscapes into intelligent augmented hearing interfaces would be a radical paradigm shift in hearing health care. Instead of a single default setting, users may navigate a multidimensional continuum of settings. The system could be optimized in real-time by combining learned preferences with crowdsourced behavioral patterns. With growing numbers of people suffering from hearing loss we need to make users an active part of hearing health care. Conversational augmented hearing interfaces may not only provide a scalable sustainable solution but also actively engage users and thereby improve their quality of life.

ACKNOWLEDGEMENTS

This work is supported by the Technical University of Denmark and the Oticon Foundation. Oticon EVOTION HAs are partly funded by European Union’s Horizon 2020 research and innovation programme under Grant Agreement 727521 EVO-TION. We would like to thank Eriksholm Research Centre and Oticon A/S for providing hardware, access to test subjects, clinical approval and clinical resources.
REFERENCES
7. Benjamin Johansen, Yannis Paul Raymond Flet-Berliac, Maciej Jan Korzepa, Per Sandholm, Niels Henrik Pontoppidan, Michael Kai Petersen, and Jakob Eg Larsen. 2017. Hearables in Hearing Care: Discovering Usage Patterns Through IoT Devices. Springer International Publishing, Cham, 39–49. DOI: http://dx.doi.org/10.1007/978-3-319-58700-4_4
Appendix B

Modeling user intents as context in smartphone-connected hearing aids

Modeling User Intents as Context in Smartphone-connected Hearing Aids

Maciej Jan Korzepa
Technical University of Denmark
Kongens Lyngby, Denmark
mjko@dtu.dk

Benjamin Johansen
Technical University of Denmark
Kongens Lyngby, Denmark
benjoh@dtu.dk

Jan Larsen
Technical University of Denmark
Kongens Lyngby, Denmark
janla@dtu.dk

Jakob Eg Larsen
Technical University of Denmark
Kongens Lyngby, Denmark
jaeg@dtu.dk

Michael Kai Petersen
Eriksholm Research Center
Snekkersten, Denmark
mkpe@eriksholm.com

Niels Henrik Pontoppidan∗
Eriksholm Research Center
Snekkersten, Denmark
npon@eriksholm.com

ABSTRACT
Despite the technological advancement of modern hearing aids, many users leave their devices unused due to little perceived benefit. This problem arises from the limitations of the current fitting procedure that rarely takes into account 1) the perceptual differences between users not explained by measurable hearing loss characteristics and 2) the variation in context-specific preferences within individuals. However, the recent emergence of smartphone-connected hearing aids opens the door to a new level of context awareness that can facilitate dynamic adaptation of settings to users’ changing needs. In this position paper, we discuss how user auditory intents could be modeled as context collected via mobile devices and suggest what kinds of contextual information are relevant when learning situation-specific intents and the corresponding preferences of hearing impaired users. Finally, we illustrate our ideas with several examples of real-life situations experienced by subjects from our study.

KEYWORDS
context awareness; user adaptation; augmented hearing

1 INTRODUCTION
In recent years, hearing aids (HA) have undergone great technological advancements transforming these once bulky, analog devices into powerful, yet discrete wearables. However, despite this substantial change, a considerable fraction of the hearing impaired population fitted with HA does not wear them [20]. One of the most commonly reported reasons for non-use of HA is that they bring users little benefit. However, as at the same time, numerous studies prove the effectiveness of modern HA [8], we rather seek the source of the problem in the limitations of the current fitting procedure.

The current audiological approach bases on prescriptive formulas that determine the frequency-gain curve for a user with specific audiogram i.e. hearing loss characteristics measured as audible hearing thresholds at different frequencies. There are, however, several issues with this approach which lead to suboptimal settings that often do not provide satisfactory level of help to users. First of all, it is well established that hearing loss is not just weakening of neural activity, but also its serious distortion [18] and thus hearing impairment cannot be fully characterized by an audiogram. Killion et al. [15] demonstrated that the ability to understand speech may differ by up to 15-20dB difference in signal-to-noise ratio for subjects with nearly identical audiogram. Similarly, the perceived loudness of soft sounds can vary greatly as shown by LeGoff et al. [17]. Even though modern HA are equipped with advanced signal processing algorithms that go beyond simple amplification, they are rarely taken into account and, without proper control, may even work against wearers. For instance, noise reduction can introduce distortions of spatial cues that might be crucial for some users to distinguish between different auditory streams [18]. All these variations in users’ cognitive processing capabilities help to explain why the standard ‘one size fits all’ approach fails, and indicate that more personalization is needed when fitting HA.

Yet, it has been also established that there are large variations in setting preferences not only between different users, but also within individuals. Keidser et al. [14] demonstrated that the preferred frequency gain characteristics are highly dependent on the auditory environment the user is in. Likewise, Johansen et al. [13] showed that individual users, when given a set of settings varying in terms of omnidirectionality, brightness and noise reduction and freedom to change between them in real, non-clinical environments, exhibit consistent usage patterns of multiple, often very contrasting, settings. These results indicate that user preferences are dependent not only on users’ cognition but also on the environments and situations they experience every day. As a result, it is not even ‘one size

ACM Reference Format:

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

UMAP’18 Adjunct, July 8–11, 2018, Singapore, Singapore
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-5784-5/18/07…$15.00
https://doi.org/10.1145/3213586.3226211

∗Niels Pontoppidan is partly funded by European Union’s Horizon 2020 research and innovation program under Grant Agreement 727521 EVOTION.
fits one’ approach that should be aimed for, but rather continuous adaptation to individual users’ perception in dynamically changing contexts.

Contextual personalization of HA has been researched for many years. Over a decade ago, Dillon et al. [6] presented a concept of HA with trainable frequency-gain curve based on acoustic measurements of environment and ecological momentary assessment, and discussed the potential benefits of such solution. In the following years, the introduction of body-worn gateway devices made it possible to connect the HA with smartphones and prototype intelligent HA systems. Aldaz et al. [4] developed a prototype that used reinforcement learning to discover user preferences based on auditory and geospatial context by prompting users to perform momentary A/B listening tests. However, only with the emergence of the current state-of-the art HA such as Oticon Opn [21], it has become possible to go beyond ecological momentary assessment by continuously tracking both users’ interactions with their HA together with auditory context perceived by these devices. For more than a month, Korzepa et al. [16] continuously observed how users switch between different HA settings in different auditory contexts, discussed possibilities and challenges of learning contextual preferences directly from continuous data and suggested the application of conversational AI interfaces and collaborative filtering in this process.

The advancements in the connectivity of hearings aids, accessibility to various sources of contextual data as well as rapidly increasing adoption of smartphones among the elderly open up new possibilities for building context-aware and user-adapted solutions for hearing healthcare. To that end, one of the key elements is the ability to distinguish between different situations hearing-impaired users experience in their daily lives and understand the difficulties they face and the intentions they have in these situations. The purpose of this position paper is to present our views on how to facilitate such context awareness in HA. In Section 2, we explain how user auditory intents can be modeled as context and propose different types of contextual information that are relevant to hearing impaired users. In Section 3, we demonstrate how context can represent the underlying intents based on a few examples of real-life situations experienced by the subjects of our study.

2 USER AUDITORY INTENTS AND CONTEXT

User intent is a common term used in web search domain and refers to the information a user is looking for with a specific goal such as learning/doing something or going somewhere. Web browser and its search engine constitute an interface that attempts to identify user preferences itself is beyond the scope of this paper as we focus here on discussing the potential of different context sources that might facilitate accurate modelling of user intents. We present them in the following sections.

2.1 Auditory context

Acoustic scene might be the richest source of information that can help to determine user intents. Numerous auditory streams mix together and form so-called soundscape - sound understood as environment that is perceived by humans. Soundscape can consist of e.g. nature sounds, human speech, music or appliance noise. They all might provide useful insights into what users do and what their auditory intent is, and can be represented in a number of ways. Basic information about the soundscape can be extracted from HA as they measure various sound characteristics in different frequency bands and use them as control parameters for signal processing algorithms. [16] represented the auditory context by clustering records based on sound pressure level, noise level, signal-to-noise ratio and modulation characteristics. The authors also used HA’ in-built sound classification that labeled soundscapes as quiet, noise, speech in quiet or speech in noise. These characteristics allowed the authors to identify primary patterns related to soundscape.

Another information that can be learned from soundscape is its higher level representation that is connected to a physical location or specific sources of sound. This can be achieved by means of acoustic scene classification (ASC) which has been widely researched for the past two decades. The methods of ASC primarily use probabilistic models (e.g. Hidden Markov Model [7]) or neural networks (e.g. convolutional neural networks [24]) usually with the input in the form of Mel-frequency cepstral coefficients (MFCC)
calculated from short audio frames. ASC has the potential to distinguish between various environments users spend their time at every day such as supermarket, street, bus, restaurant, party, forest or seashore. Such acoustic environment awareness allows to infer much more about user intent. For example, it is very likely that in a restaurant or at a party, the user wants to understand speech despite the surrounding noise while in a forest or at a seashore, the user might rather want to focus on the sounds of nature.

Yet another highly informative component of soundscape is connected with what HA are primarily optimized for - enhancing human voices. The characteristics of human speech such as pitch, timbre or pace vary greatly between speakers dependent on their gender, age, language, possible disorders and many other factors. Additionally, signal processing in HA influences speech differently due to its varied characteristics and, as a result, users' perception of different speakers is challenged [9]. Voice- and speaker-awareness in HA would allow to learn user preferences and personalize settings with respect to voice characteristics. Simple distinction between male and female combined with basic hand-crafted features representing pitch and timbre would be already very informative but one could go even further. Speaker embeddings have been recently widely used for tasks such as speaker recognition or diarization with state-of-the-art results achieved by deep learning methods (e.g. [11, 19]). Speaker embeddings are real vector representations of different speakers that encode distinctive characteristics of their voices. Using such embeddings, HA could not only learn which types of voices need what processing to optimize user's perception without being constrained to a set of arbitrary features that might miss some important characteristics, but also they could help to selectively amplify specific, for example familiar, voices in order to solve the cocktail party problem [10].

2.2 Location

Similar to how acoustic scene analysis gives insight into the user's location, geolocation data can provide information about the acoustic environment and the corresponding user intents. In this way, these two context sources can complement each other by providing information if one is lacking. When both are available, they can be used as labeled data to adapt and optimize ASC model to user-specific environments. Location plays also another important role - it might be mapped to an activity which in turn could be interpreted as specific intents.

The type of location may often be obtained via a public API such as Google Places API [3] or Foursquare Places API [1] based on geographic coordinates. We see particularly big potential in the latter one which, in many countries, has very detailed venue maps and supports adding new, private or public, places. Additionally, venue category structure is very fine and hierarchical (e.g. Arts & Entertainment → Performing Arts Venue → Theater) which can facilitate learning user preferences on different levels of granularity.

Most commonly visited locations can be also learned based on clustering of geospatial coordinates (e.g. by HDBSCAN [12]). This approach might prove helpful especially when there is no access to venue type information. However, as locations expressed as a cluster of coordinates do not carry any semantic information, it is not possible to benefit from learned preferences without knowing to what degree the new locations resemble the familiar ones.

2.3 Time and motion

Some user intents might be related to the way users move. For example, when biking, the user might prefer to reduce traffic noise to focus on driving. Motion or activity can be easily predicted using one of many public APIs such as Google's Activity Recognition API [2]. Moreover, motion can be used to track location changes more robustly.

Time is another factor that can support modelling user intents as it often carries information about repeating activities that user is involved in. As shown by Johansen et al. [13], user preferences can greatly change throughout the day and week (especially weekdays vs weekends). Naturally, time context carries lots of uncertainty as what a user does at a specific time may vary greatly, but it might often prove very informative when coupled with other context sources. Time can be also potentially considered as a measure of mental fatigue. Listening in challenging environments requires higher listening effort and increases mental fatigue, especially for hearing-impaired people [5]. Tracking time of day and time spent in challenging listening conditions could conceivably allow to adjust HA settings (e.g. increase noise reduction) according to the estimated level of mental fatigue of a user.

3 DISCUSSION

Relating user intents to a single source of context is naturally prone to errors that might arise not only from limited accuracy of context classification but also from excessive generality. For instance, in a noisy environment with speech, the user might want to understand a specific speaker or zone out from the noisy surroundings. To model user intents, different context sources need to be used in a complementary way. An example from a not yet published study, where we collected data over 2 months capturing user's interactions with the HA and the corresponding context from 10 users, may illustrate both the different components defining the context as well as how they relate to the actual user intents which we learned through interviews with the test persons.

Figure 1 shows how a subject changed his HA program preference and the corresponding context classification for acoustic environment, location, type of location and activity over a period of 12 hours. Between 1pm and 5pm, the subject attended a bridge card game competition. In order to focus on the game, he aimed to attenuate the ambience and chose the program reducing noise (’P4’). Interestingly, the HA does not detect much noise, but mainly speech. In this case, the environment classification alone would not be sufficient to capture user intents and the corresponding program preference. Nor would it explain the preference for a brighter, more intense sound later in the evening, caused by a wish to enhance a TV soundtrack. However, adding the location and time might here generate recognizable patterns.

Another subject reported the benefit of a program offering maximum brightness and amplification of soft sounds during walking
his dog in the evening when his intention is to enjoy the subtle sounds of nature. In this case, motion combined with acoustic scene and possibly time would be needed to capture and act upon this user's preference. Yet another subject indicated the benefits of using a highly omnidirectional program with some added brightness when his intention is to understand other speakers during lunch in a noisy corporate canteen. In this case, location, acoustic scene and time could be combined to define the user intent. The last example is a subject who generally prefers an omnidirectional, bright setting enhancing the gain in mid and high frequencies, but complains that some female voices get too shrill in that program. Tracking the speakers' voice characteristics might facilitate adjusting the brightness to optimize the user's listening comfort.

The quoted examples serve as yet another proof that hearing impaired users have greatly varying intents and setting preferences that go beyond the need for speech understanding. Understanding intents and personalizing settings with respect to them requires redefining the concept of context awareness in hearing aids. Basic distinction between quiet/noisy and speech/non-speech environments is simply not sufficient to discern between many situations in which user auditory intents differ. However, the new generation of smartphone-connected hearing aids opens the door to infer behavioral patterns from multiple kinds of context that can be obtained through ubiquitous mobile sensors, powerful deep learning techniques and widely available cloud APIs. Combining various soundscape characteristics, location, motion, time and potentially other contextual features not considered in this paper would be a major step towards a whole new level of user-adaption that could unlock the full potential of modern hearing aids.

REFERENCES


Appendix C

Simulation environment for guiding the design of contextual personalization systems in the context of hearing aids

Simulation Environment for Guiding the Design of Contextual Personalization Systems in the Context of Hearing Aids

Maciej Korzepa  
Kongens Lyngby, Denmark  
mjko@dtu.dk  

Michael Kai Petersen  
Eriksholm Research Centre, Demant  
Snekkersten, Denmark  

Jakob Eg Larsen  
Technical University of Denmark  
Kongens Lyngby, Denmark  
mortenm@gmai.com

ABSTRACT

Adjusting the settings of hearing aids in a clinic is challenging as the measured thresholds of audibility do not reflect many aspects of cognitive perception or the resulting differences in auditory preferences across different contexts. Online personalization systems have a potential to solve this problem, yet the lack of contextual user preference data constitutes a major obstacle in designing and implementing them. To address this challenge, we propose a simulation-based framework to inform and accelerate the development process of online contextual personalization systems in the context of hearing aids. We discuss how to model hearing aid users and context allowing partial observability, and propose how to generate plausible preference models using Gaussian Processes incorporating assumptions about the environment in a controlled way. Finally, on a simple example we demonstrate how an uncertainty-driven agent can efficiently learn from noisy user responses within the proposed framework. We believe that such simulated environments are vital for successful development of complex context-aware online recommender systems.

CCS CONCEPTS

- Computing methodologies → Simulation environments; Gaussian processes;  
- Human-centered computing → Contextual design; User models;  
- Information systems → Recommender systems.

KEYWORDS

contextual personalization; simulation environments; hearing aids

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:


1 INTRODUCTION

The increasing prevalence of hearing loss combined with limited clinical resources lead to users being dispensed hearing aids (HA) fitted purely based on audiograms which only measure the loss of audibility at different frequencies. However, as hearing loss is a complex distortion in auditory nerve activity patterns rather than just a loss of sensitivity [13], audiogram-based fitting cannot address differences among the hearing-impaired such as in speech understanding in noisy environments [14]. Moreover, apart from large differences between individuals, there is often lots of variation in preferences within individuals. For example, binaural loudness perception may vary up to 30dB depending on the characteristics of the soundscape [16]. Similarly, depending on auditory intents in various daily situations and environments, users show preferences for very contrasting settings [11, 12]. These findings indicate that extensive contextual personalization is essential for the hearing impaired to minimize the risk of experiencing serious consequences of hearing loss related to e.g. well-being and cognitive load [2].

Multiple attempts have been made at optimizing hearing aid settings based on user feedback addressing different aspects such as fine-tuning frequency gain curve in static environments using Bayesian optimization [15], context-aware online preference learning using reinforcement learning (RL) [1] and learning shared preferences with hierarchical Bayesian models[5]. While all these aspects appear to be very important in designing systems for personalization of HA settings, there is, as of now, no system that incorporates all of them. When designing such a system, it is tempting to take a more holistic perspective and think of HA as an interactive, context-aware, online recommender system [17] that should continually adapt not only to users’ setting preferences, but also to different characteristics of how users interact with the system, such as level of engagement, frequency of interactions or consistency of feedback, which can vary tremendously across users. A major obstacle in developing such a system is the lack of data that would guide its iterative design and evaluation. User studies are typically very time-consuming, costly, limited in size and carried out when the designer believes the prototype (which is developed without adequate data) is ready for testing. This renders the development of a complex personalization system very difficult.

A promising direction to facilitate development and evaluation of recommender systems in domains without adequate historical data is designing a simulated environment that incorporates relevant characteristics of the real one. Le et al. [10] recently proposed RecSim, a framework for constructing such dynamic environments that allows for evaluation of RL-based recommendation agents. Using this framework, the goal of the modeler is to define the building blocks specifying in a probabilistic way what characteristics users
consist of, how these characteristics contribute to liking specific recommended items and how they evolve over time. Importantly, the goal is not to create a completely realistic environment, but rather a tool that will allow to test agent’s learning capabilities with respect to various assumptions the designer wants to incorporate about the environment such as partial observability of user or context features, noisiness of responses or complexity of preferences.

In this article, we extend this framework and adapt it to the HA domain. First, we introduce user-dependent context that, together with user characteristics, conditions user preferences. The context assumes partial observability which is crucial to evaluate how the inability to observe all the relevant context can potentially impede the agent’s learning performance. Secondly, we suggest how to model context and user features in the HA domain using commonly available data. Finally, we present a Gaussian Process based model for sampling plausible contextual preferences whose complexity, dependence on observable and hidden features as well as degree of correlation between different users can be easily controlled with a set of hyperparameters. Even though the proposed approach is presented in the context of the HA domain, the generative preference model works with arbitrary context and user features and thus can be easily adapted to other domains. While modelling agents is not the focus of this paper, we also implement a simple agent and perform a simple simulation to demonstrate how agents can interact with the proposed environment and learn.

2 SIMULATION ENVIRONMENT

Hearings aids represent a domain in which users might have completely different preferences depending on the context they are in. Settings that work well for a user in one situation might totally fail in another one. While introducing context-awareness in a personalization system allows to distinguish between various situations and potentially offer optimal settings at the right moment, it also adds an extra layer of complexity and challenges to the problem. What if the observed context does not reflect user preferences well? Can the agent actually learn contextual preferences in an online setting? Can the agent benefit from correlations between contextual preferences of multiple users? What if users’ contextual preferences are more complex than one expects? It is crucial for the designer of a recommendation agent to have some insight into its flexibility and performance under a range of different assumptions about the environment.

To facilitate informed development of agents, we borrow and adapt the RecSim simulation framework proposed by Ie et al. [10]. We present the modified diagram of the data flow in the framework in Fig. 1. A single cycle of contextual recommendation starts by sampling a user from a set of users generated for the simulation. The user is represented by a set of observable and hidden features according to the user model. Next, the context, also consisting of observable and hidden features, is sampled according to the user’s context model (which might have a temporal structure). The agent observes the observable components of user and context features and based on them generates a recommendation with a number of settings within a given setting space. Then, the user generates a noisy response to the recommendation based on the internal contextual preference model. The agent observes the user’s response and updates its internal recommendation model. Additionally, the state of the user changes according to the user’s transition model (e.g., engagement might increase or decrease).

In this paper, we focus on the construction of user and context features in the HA domain and propose a model for generating contextual preference models in a controlled way. We start with a high-level overview of the proposed approach in section 2.1. In sections 2.2 and 2.3 we discuss how context and user features can be modeled in the HA domain, and define mechanisms that will allow control over generated preferences. The details of how we generate contextual preferences are specified in section 2.4. Section 2.5 relates user preferences to an actual response to agent’s recommendations. Finally, in section 3 we present a simple simulation to demonstrate how agents interact with the environment and conclude with a discussion in section 4.

2.1 Overview

We start by presenting an outline of the approach we propose to generate user preference models. We define a HA setting vector \( s \in \mathbb{R}^k \) (e.g., low-dimensional reparametrization of added gain at different frequencies) that we can control, and context vector \( c \in \mathbb{R}^d \) that fully explains user auditory needs at given instance of time. Our goal is to create for each user a function \( (c, s) \mapsto f(c, s) \in \mathbb{R} \) that maps context and settings to a latent preference value which indicates the degree to which the user likes settings \( s \) in context \( c \). Additionally, we want to be able to induce correlations between functions for similar users and be able to evaluate the function efficiently for any \( c \) and \( s \). To model these functions we employ Gaussian Processes (GPs), a nonparametric class of models that enables straightforward control over complexity of the functions and correlations between them through kernels [18]. We first use a GP prior to sample preferred settings over a specified range of contexts to construct preference datasets that incorporate assumptions about correlations of user preferences and their complexity. We then define the latent preference of a user to be the posterior mean of another GP trained using the dataset specific to that user which allows us to evaluate their preference for any point in the complete domain of possible contexts and settings.

2.2 Context model

Nowadays HA users are not limited to elderly spending most of their time at home, but include many who are professionally and socially
Figure 2: Varying the ratio of context lengthscale parameters $\lambda_o$ and $\lambda_h$ for observed and latent components allows us to sample user preferences with different assumptions about how well the observed context explains the actual user preferences.

While auditory preferences can vary greatly across users, not all users are completely different [4, 17]. Within a large population of users, it is natural that there are some groups of users who exhibit similar preferences, e.g. a preference towards enhanced brightness in speech environments [17]. We hypothesize that similarities in user preferences are the result of similarities in some characteristics that define users. These characteristics might include both observable and hidden features. A promising strategy to define observable features is to characterize user’s hearing loss across two independent dimensions being audibility-related distortions (e.g. as specified by an audiogram) and non-audibility-related distortions characterizing reduction in binaural and temporal fine-structure processing abilities [19]. If such observable characteristics give rise to specific kind of preferences, knowing them would be crucial to alleviate cold start problem for new users when the agent has not collected any preference data about them yet.

Hidden features might relate to various cognitive aspects that are hard or impossible to measure. Hidden features can be modelled e.g. by treating some of the observed ones as hidden or constructing a Gaussian mixture (or a different model) that generates them.

We define user-specific feature vector $u$ as a concatenation of observable and hidden user-specific feature vectors, $u_o$ and $u_h$ respectively, i.e. $u = (u_o, u_h)$, and a distribution $p(u)$ that generates them. Through simulation, we expect to measure the impact that the presence of preferences induced by observable and hidden user characteristics has on the agent’s learning performance. To be able to generate such preferences in a controlled way, we propose a kernel that is a weighted combination of RBF kernels with lengthscales $\lambda_u = (\lambda_u^o, \lambda_u^h)$ operating on $u_o$ and $u_h$ respectively:

$$
\kappa_u(u, u', \lambda_u) = \theta \kappa_{RBF}(u_o, u'_o, \lambda_u^o) + (1 - \theta) \kappa_{RBF}(u_h, u'_h, \lambda_u^h),
$$

where $\kappa_{RBF}(x, x', \lambda)$ denotes a radial basis function (RBF) kernel with lengthscale $\lambda$ and unit output variance, in which the correlation between the modelled function falls with the increasing distance between the inputs (i.e. context).

In this paper, for the demonstration purposes, we do not assume any specific model on context $c$ and simply define $c = (c_{obs}, c_{hid})$ with $c_{obs} \in [-2, 2]$ and $c_{hid} \in [-2, 2]$ representing scalar observed and hidden context values. Moreover, we limit the settings to a single scalar value representing overall added gain.

By running simulations with different values of lengthscale parameters $\lambda^c$, we are able to evaluate different aspects of agent’s performance. One of the most important ones might be to measure how well the agent performs when user preferences can be fully explained by the observed context and how quickly its performance degrades with an increasing contribution of the hidden context. We can generate preferences with different degree of dependence on observed and hidden context by controlling ratio $\lambda^c_o/\lambda^c_h$ as demonstrated in Fig. 2. Another important aspect is how complex preferences the agent is able to learn efficiently. We can generate preferences of different complexity by adjusting the magnitude of the lengthscale parameters as shown in Fig. 3.

2.3 User model

While auditory preferences can vary greatly across users, not all users are completely different [4, 17]. Within a large population of users, it is natural that there are some groups of users who exhibit similar preferences, e.g. a preference towards enhanced brightness in speech environments [17]. We hypothesize that similarities in user preferences are the result of similarities in some characteristics that define users. These characteristics might include both observable and hidden features. A promising strategy to define observable features is to characterize user’s hearing loss across two independent dimensions being audibility-related distortions (e.g. as specified by an audiogram) and non-audibility-related distortions characterizing reduction in binaural and temporal fine-structure processing abilities [19]. If such observable characteristics give rise to specific kind of preferences, knowing them would be crucial to alleviate cold start problem for new users when the agent has not collected any preference data about them yet.

Hidden features might relate to various cognitive aspects that are hard or impossible to measure. Hidden features can be modelled e.g. by treating some of the observed ones as hidden or constructing a Gaussian mixture (or a different model) that generates them.

We define user-specific feature vector $u$ as a concatenation of observable and hidden user-specific feature vectors, $u_o$ and $u_h$ respectively, i.e. $u = (u_o, u_h)$, and a distribution $p(u)$ that generates them. Through simulation, we expect to measure the impact that the presence of preferences induced by observable and hidden user characteristics has on the agent’s learning performance. To be able to generate such preferences in a controlled way, we propose a kernel that is a weighted combination of RBF kernels with lengthscales $\lambda_u = (\lambda_u^o, \lambda_u^h)$ operating on $u_o$ and $u_h$ respectively:

$$
\kappa_u(u, u', \lambda_u) = \theta \kappa_{RBF}(u_o, u'_o, \lambda_u^o) + (1 - \theta) \kappa_{RBF}(u_h, u'_h, \lambda_u^h),
$$

where $\kappa_{RBF}(x, x', \lambda)$ denotes a radial basis function (RBF) kernel with lengthscale $\lambda$ and unit output variance, in which the correlation between the modelled function falls with the increasing distance between the inputs (i.e. context).

In this paper, for the demonstration purposes, we do not assume any specific model on context $c$ and simply define $c = (c_{obs}, c_{hid})$ with $c_{obs} \in [-2, 2]$ and $c_{hid} \in [-2, 2]$ representing scalar observed and hidden context values. Moreover, we limit the settings to a single scalar value representing overall added gain.

By running simulations with different values of lengthscale parameters $\lambda^c$, we are able to evaluate different aspects of agent’s performance. One of the most important ones might be to measure how well the agent performs when user preferences can be fully explained by the observed context and how quickly its performance degrades with an increasing contribution of the hidden context. We can generate preferences with different degree of dependence on observed and hidden context by controlling ratio $\lambda^c_o/\lambda^c_h$ as demonstrated in Fig. 2. Another important aspect is how complex preferences the agent is able to learn efficiently. We can generate preferences of different complexity by adjusting the magnitude of the lengthscale parameters as shown in Fig. 3.
Parameter $\theta$ controls the degree to which the preferences are induced by the observed and by the hidden features. For two users characterized by feature vectors $u_i$ and $u_j$, we expect their preferences to be identical if $k_u(u_i, u_j) = 1$ and completely independent if $k_u(u_i, u_j) = 0$. In this paper, as we focus on how to generate correlated preferences based on similarities between users rather than using real audiological data to model the actual features, we arbitrarily construct observed feature vectors $u_i$ and set $\theta$ to 1. We show the resulting kernel matrix shown in Fig. 4.

It is also important in a simulation to allow for behavioral and personal differences between users. HA users can differ in the extent of HA usage, engagement in the hearing loss treatment, familiarity with new technologies or cognitive abilities to distinguish between differences in sound produced by HA. In this paper, due to space limitations, we do not include them in the simulation, but we get back to them in the discussion in section 4.

2.4 Preference model

Over a wide range of settings, there tend to exist settings that a user finds more effective than average and ones less effective than average. We refer to them as positive and negative preferences. As the task of an agent is to learn which settings are the effective ones, we focus on modeling positive preferences. We assume that given a fixed context $c$ user’s preference peaks around a single setting $s$ and falls towards the average preference away from it. This assumption seems plausible if the settings are parametrized such that larger distance between two settings corresponds to a bigger perceptual difference in their resulting HA output sounds.

We model preferences for $M$ users whose feature vector $u$ we sample from the user generating distribution $P(u)$. For each user $j$, we aim to construct a training dataset $D_j = \{(c_i, s_i, f_{ij})|i = 1, \ldots, N\}$ that consists of $N$ pairs indicating optimal setting $s_i$ for context $c_i$ for which the user’s latent preference $f$ attains some high value $f_{ij}$. We assume that individual dimensions $s_k$ of setting $s$ can be modelled independently. We define a multi-task GP [6] with $\hat{s}_k \in \mathbb{R}^M$ output where $j$-th output/task models preferred $s_k$ for user $j$:

$$\hat{s}_k \sim \mathcal{GP}(0, \delta \cdot k_c(c, c'; \lambda_c^k) \cdot k_u(u, u'; \lambda_u^k)),$$

where $\delta \in \mathbb{R}^+$ controls the output scale, $\lambda_c$ is the context kernel as defined in (1) with $\lambda_c^k$ lengthscales and $k_u$ is the user kernel as defined in (2) with $\lambda_u^k$ lengthscales. As the correlations in the inputs and correlations in the outputs are independent of each other, the resulting kernel matrix is Kronecker factored [6] which enables efficient computations. We construct a representative set $C_k$ of $N$ contexts $c_i$ that covers the space of possible contexts well (we can do it e.g. by taking $k \gg N$ samples from $P(c)$ and selecting a subset of $N$ such that the distance between any two contexts in the subset is maximized) and draw a sample $\hat{s}_k \in \mathbb{R}^{N \times M}$ from the GP prior defined in (3) for each setting dimension $k$. Then, the optimal setting $\hat{s}_k$ in context $c_i$ for user $j$ is defined as $\hat{s}_i = ((\hat{s}_{ij})_1, \ldots, (\hat{s}_{Nj})_1)$. Having dataset $D_j$, we define a GP prior on $f_j$:

$$f_j \sim \mathcal{GP}(m_j, k_c(c, c'; \lambda_c^k_p) \cdot k_{RQ}(s, s'; \lambda^r, \alpha)),$$

where $m_j$ denotes user-dependent mean, $k_c$ is the context kernel as defined in (1) with $\lambda_c^k_p$ lengthscales and $k_{RQ}$ denotes a rational quadratic kernel with lengthscale $\lambda^r$ and scale mixture parameter $\alpha$ which controls how quickly $f_j$ decays towards the mean $m_j$ away from the optimal settings. By performing noiseless GP regression, we can evaluate the GP posterior mean, $\mathbb{E}[f'(D_j, c', s')]$ (which has a simple, well-known analytic solution [18]), at any context $c'$ for any setting $s'$, and we define it to be the latent preference of user $j$.

To visualize how the sampled settings and preferences reflect correlations between users, we construct a grid over $c_{obs}$ and $c_{hid}$, use the user kernel matrix shown in Fig. 4, set $\delta$ to 10 and $\lambda_c^k$ to (1,1) and draw a sample from the GP prior defined in (3). In Fig. 5a, we show the sampled preferred setting (gain) over the context grid for 8 users. Further, we set $m_j$ to 0 for each $j$, $\lambda_c^k_p$ to (0.15, 0.15), $\lambda^r$ to 0.75 and $\alpha$ to 0.5, and generate preference models by performing GP regression on the previously sampled preferred setting dataset for each user. In Fig. 5b, we show how user preference for gain setting $s \in [-10, 10]$ changes over observed context $c_{obs}$ with hidden context $c_{hid}$ fixed to a specific value.

2.5 Response model

The agent learns from users’ responses to offered recommendations. Even though users respond according to their latent preference model, their responses are inherently noisy (e.g. due to dependence on latent context or cognitive limitations to provide consistent feedback). The form of user response is typically decided by the system designer who takes into account suitability of a specific form from the users’ perspective (e.g. what kind of response is more engaging or can be given quicker) and the agent’s perspective (e.g. how efficiently it can learn). Some examples of response types proposed for personalization of hearing loss compensation include binary evaluations of sound quality for individual settings [20], or pair-wise comparisons of two settings using continuous response in $[0, 1]$ range (represented by a slider) where the deviation from the middle of the range indicates the degree of preference towards one or the other setting [15].

In our simulation, we assume users evaluate recommended settings independently by using a slider whose extreme positions,
corresponding to values 0 and 1, indicate maximum dissatisfaction or satisfaction with a given setting. A natural choice of likelihood for user response $y$ is then the Beta distribution as in [15]:

$$y \sim \text{Beta}(\nu \cdot \mu(f), \nu \cdot (1 - \mu(f))),$$

(5)

where $\mu(f)$ is a function mapping latent preference $f$ to $[0, 1]$ range and defines the mean of the response, and $\nu$ controls the noise of the response (for $\nu \to \infty$, user response becomes noise-free i.e. $y \to \mu(f)$). We define the mapping $\mu$ as:

$$\mu(f) = \Phi(\sigma \cdot f),$$

(6)

where $\Phi$ denotes the cumulative standard normal distribution and $\sigma$ controls the slope of the transition between negative and positive preference. Parameters $\nu$ and $\sigma$ are typically user dependent (e.g. $\nu$ may depend on users’ cognitive capabilities allowing to provide more or less consistent feedback). In a large-scale simulation, it would be reasonable to generate users with $\nu$ and $\sigma$ sampled according to some prior distribution that we could construct based on our assumptions about it or historical data, if available. In our simulation, we simply set $\nu$ to 10 and $\sigma$ to 0.4.

3 SIMULATIONS

In this paper, we do not focus on agent design and limit the scope of evaluation to learning preferences of an individual user. In our simulation we evaluate an agent based on Bayesian optimization. The agent maintains a surrogate GP model that, based on dataset $\mathcal{D}$ consisting of past interactions with the user, models user’s posterior latent preference $p(f|\mathcal{D})$ with mean $\mu_f(c,s)$ and variance $\sigma_f^2(c,s)$ for context $c$ and settings $s$. When observing a new context $c^*$, to decide on which setting $s^*$ to recommend the agent optimizes Upper Confidence Bound (UCB) [7] over a given possible setting space $\mathcal{S}$:

$$s^* = \arg\max_{s \in \mathcal{S}} \mu_f(c^*, s) + \beta \cdot \sigma_f(c^*, s),$$

(7)

where $\beta$ controls the trade-off between exploration and exploitation. The agent’s surrogate GP model uses Matern kernel and Beta likelihood parametrized as in (5) to map from latent function $f$ to actual user response $y$. As the Beta likelihood is not conjugate with the GP prior, we use Variational Inference to infer the GP posterior and tune the hyperparameters of the GP by maximizing the marginal likelihood of the data. We use GPyTorch [9] to implement the GP model with variational training scheme and employ BoTorch [3] to optimize the UCB acquisition function.

We compare the UCB agent to a simple baseline agent that always offers a constant setting $\hat{s}$ that maximizes average user preference over the distribution of user contexts $p(c)$ assumed in the simulation, i.e. $\hat{s} = \arg\max_c \mathbb{E}_{p(c)} f(c, s)$. This represents the optimal recommendation without context awareness. We refer to this agent as oracle mean agent. We measure the performance of each agent as the cumulative regret at interaction $t$ given by $r_t = \sum_{i=1}^t (\mu(f_i) - y_i)$, where $f_i$ is the maximum user preference in context $c_i$ observed at interaction $i$ and $y_i$ is the user response to the agent’s recommendation at interaction $i$.

We run the simulation for user 4 (characterized by preferences shown in Fig. 5) and limit the number of interactions to $T = 200$. We consider two scenarios: full observability with hidden context $c_{hid}$ fixed to 1 and partial observability with hidden context sampled uniformly from $[-2, 2]$ range at each interaction. In both scenarios, the observed context $c_{obs}$ is sampled uniformly (and independently of the hidden context) from $[-2, 2]$ range. For UCB agent we set $\beta = 1$ and start with 5 random setting recommendations before the agent switches to its standard UCB acquisition. UCB agent retrains its surrogate GP model after each acquisition. We repeat the simulation 5 times for both scenarios with a different sample of $T$ contexts.

We show the cumulative regret incurred by the agents in Fig. 6. In the full observability scenario, the UCB agent starts incurring consistently lower regrets than the oracle mean agent already after 15 interactions and finishes the simulation with 33% lower cumulative regret. In the partial observability scenario, the presence of hidden context greatly impedes the UCB agent’s performance, but
still it is able to take advantage of the observed context and perform better than oracle mean agent after 44 interactions. In both scenarios, one can clearly see the exploration and exploitation phases of the UCB agent.

The proposed framework allows to model contextual preferences considering both observable and latent aspects of cognitive perception in a flexible way. Simulation environments like this may be vital to shape new human-centered adaptive personalization systems in domains such as hearing healthcare where historical data is very limited and where (often dynamic) human factors play an important role in the personalization process.

**REFERENCES**


Figure 6: Cumulative regret in full (a) and partial (b) observability scenarios averaged over 5 runs. The dashed line indicates the point at which the UCB agent starts incurring consistently lower regret than the oracle mean agent.

4 DISCUSSION

In our simulation we evaluated a simple Bayesian optimization agent in a single user environment. In practice, we are interested in simulating personalization on a scale of thousands of users using agents that also discover and exploit similarities between users to boost the learning speed. With a large pool of users, we can model the diversity of contextual preferences, as well as human factors such as cognitive perception or motivation. If users differ in the ability of providing consistent feedback, the agent might dynamically optimize its interventions. For users giving very inconsistent responses, the agent might learn to offer more contrasting settings and ask for a binary response to compensate for a high level of noise. Conversely, the agent might provide more fine-grained recommendations if user responses are consistent. Similarly, the frequency and timing of interventions may impact the users’ motivation to interact with the agent. An intelligent agent should learn dynamically when and how to interact with users to keep them motivated over a long period of time. Simulation is a perfect tool to evaluate agents’ capabilities of adaptation to such human factors and dynamic behaviors.

The closer the simulation environment is to the actual one, the less difference we expect between how well agents perform in simulation and in real world. When deploying the developed system for the first time, it is very likely that due to inaccurate assumptions the performance of the agent will be significantly worse than in the simulation. It is therefore reasonable to start with an agent that operates in a simpler space of settings and observed context than intended in the long run. Even if the first version fails to reach satisfactory performance, valuable insights about the characteristics of user preferences and interactions are gathered and can be used to revise the implementations implemented in the simulator so that an improved agent can be developed. This creates a self-reinforcing loop - the data collected by agents improves the simulation which in turn leads to better performing agents.
Simulation environment for guiding the design of contextual personalization systems in the context of hearing aids
Appendix D

Approximate inference turns deep networks into Gaussian processes

Approximate Inference Turns Deep Networks into Gaussian Processes

Mohammad Emtiyaz Khan  
RIKEN Center for AI Project  
Tokyo, Japan  
emtiyaz.khan@riken.jp

Alexander Immer*†  
EPFL  
Lausanne, Switzerland  
alexander.immer@epfl.ch

Ehsan Abedi*†  
EPFL  
Lausanne, Switzerland  
ehsan.abedi@epfl.ch

Maciej Korzepa*†  
Technical University of Denmark  
Kgs. Lyngby, Denmark  
mjko@dtu.dk

Abstract

Deep neural networks (DNN) and Gaussian processes (GP) are two powerful models with several theoretical connections relating them, but the relationship between their training methods is not well understood. In this paper, we show that certain Gaussian posterior approximations for Bayesian DNNs are equivalent to GP posteriors. This enables us to relate solutions and iterations of a deep-learning algorithm to GP inference. As a result, we can obtain a GP kernel and a nonlinear feature map while training a DNN. Surprisingly, the resulting kernel is the neural tangent kernel. We show kernels obtained on real datasets and demonstrate the use of the GP marginal likelihood to tune hyperparameters of DNNs. Our work aims to facilitate further research on combining DNNs and GPs in practical settings.

1 Introduction

Deep neural networks (DNN) and Gaussian processes (GP) models are both powerful models with complementary strengths and weaknesses. DNNs achieve state-of-the-art results on many real-world problems providing scalable end-to-end learning, but they can overfit on small datasets and be overconfident. In contrast, GPs are suitable for small datasets and compute confidence estimates, but they are not scalable and choosing a good kernel in practice is challenging [3]. Combining their strengths to solve real-world problems is an important problem.

Theoretically, the two models are closely related to each other. Previous work has shown that as the width of a DNN increases to infinity, the DNN converges to a GP [4, 5, 13, 16, 22]. This relationship is surprising and gives us hope that a practical combination could be possible. Unfortunately, it is not clear how one can use such connections in practice, e.g., to perform fast inference in GPs by using training methods of DNNs, or to reduce overfitting in DNNs by using GP inference. We argue that, to solve such practical problems, we need the relationship not only between the models but also between their training procedures. The purpose of this paper is to provide such a theoretical relationship.

We present theoretical results aimed at connecting the training methods of deep learning and GP models. We show that the Gaussian posterior approximations for Bayesian DNNs, such as those obtained by Laplace approximation and variational inference (VI), are equivalent to posterior distributions of GP regression models. This result enables us to relate the solutions and iterations of a deep-learning algorithm to GP inference. See Fig. 1 for our approach called DNN2GP. In addition,
we can obtain GP kernels and nonlinear feature maps while training a DNN (see Fig. 2). Surprisingly, a GP kernel we derive is equivalent to the recently proposed neural tangent kernel (NTK) [8]. We present empirical results where we visualize the feature-map obtained on benchmark datasets such as MNIST and CIFAR, and demonstrate their use for DNN hyperparameter tuning. The code to reproduce our results is available at https://github.com/team-approx-bayes/dnn2gp. The work presented in this paper aims to facilitate further research on combining the strengths of DNNs and GPs in practical settings.

1.1 Related Work

The equivalence between infinitely-wide neural networks and GPs was originally discussed by Neal [16]. Subsequently, many works derived explicit expressions for the GP kernel corresponding to neural networks [4, 7, 16] and their deep variants [5, 6, 13, 18]. These works use a prior distribution on weights and derive kernels by averaging over the prior. Our work differs from these works in the fact that we use the posterior approximations to relate DNNs to GPs. Unlike these previous results, our results hold for DNNs of finite width.

A GP kernel we derive is equivalent to the recently proposed Neural Tangent Kernel (NTK) [8], which is obtained by using the Jacobian of the DNN outputs. For randomly initialized trajectories, as the DNN width goes to infinity, the NTK converges in probability to a deterministic kernel and remains asymptotically constant when training with gradient descent. Jacot et al. [8] motivate the NTK by using kernel gradient descent. Surprisingly, the NTK appears in our work with an entirely different approach where we consider approximations of the posterior distribution over weights. Due
to connections to the NTK, we expect similar properties for our kernel. Our approach additionally shows that we can obtain other types of kernels by using different approximate inference methods.

In a recent work, Lee et al. [14] derive the mean and covariance function corresponding to the GP induced by the NTK. Unfortunately, the model does not correspond to inference in a GP model (see Section 2.3.1 in their paper). Our approach does not have this issue and we can express Gaussian posterior approximations on a Bayesian DNN as inference in a GP regression model.

2 Deep Neural Networks (DNNs) and Gaussian Processes (GPs)

The goal of this paper is to present a theoretical relationship between training methods of DNNs and GPs. DNNs are typically trained by minimizing an empirical loss between the data and the predictions. For example, in supervised learning with a dataset $D := \{(x_i, y_i)\}_{i=1}^N$ of $N$ examples of input $x_i \in \mathbb{R}^D$ and output $y_i \in \mathbb{R}^K$, we can minimize a loss of the following form:

$$\bar{\ell}(D, w) := \sum_{i=1}^N \ell_i(w) + \frac{1}{2}\delta w^\top w,$$

where $f_w(x) \in \mathbb{R}^K$ denotes the DNN outputs with weights $w \in \mathbb{R}^P$, $\ell(y, f(x))$ denotes a loss function between an output $y$ and the function $f(x)$, and $\delta$ is a small $L_2$ regularizer. We assume the loss function to be twice differentiable and strictly convex in $f$ (e.g., squared loss and cross-entropy loss). An attractive feature of DNNs is that they can be trained using stochastic-gradient (SG) methods [11]. Such methods scale well to large data settings.

GP models use an entirely different modeling approach which is based on directly modeling the functions rather than the parameters. For example, for regression problems with scalar outputs $y_i \in \mathbb{R}$, consider the following linear basis-function model with a nonlinear feature-map $\phi(x) : \mathbb{R}^D \mapsto \mathbb{R}^P$:

$$y = \phi(x)^\top w + \epsilon, \quad \text{with } \epsilon \sim \mathcal{N}(0, \sigma^2), \quad \text{and } w \sim \mathcal{N}(0, \delta^{-1}I_P),$$

where $I_P$ is a $P \times P$ identity matrix and $\sigma^2$ is the output noise variance. Defining the function to be $f(x) := \phi(x)^\top w$, the predictive distribution $p(f(x_*)|\mathcal{D})$ at a new test input $x_*$ is equal to that of the following model directly defined with a GP prior over $f(x)$ [23]:

$$y = f(x) + \epsilon, \quad \text{with } f(x) \sim \mathcal{GP}(0, \kappa(x, x')),$$

where $\kappa(x, x') := \mathbb{E}[f(x)f(x')] = \delta^{-1}\phi(x)^\top \phi(x')$ is the covariance function or kernel of the GP. The function-space model is more general in the sense that it can also deal with infinite-dimensional vector feature maps $\phi(x)$, giving us a nonparametric model. This view has been used to show that as a DNN becomes infinitely wide it tends to a GP, by essentially showing that averaging over $p(w)$ with the feature map induced by a DNN leads to a GP covariance function [16].

An attractive property of the function-space formulation as opposed to the weight-space formulation, such as (1), is that the posterior distribution has a closed-form expression. Another attractive property is that the posterior is usually unimodal, unlike the loss $\bar{\ell}(D, w)$ which is typically nonconvex. Unfortunately, the computation of the posterior takes $O(N^3)$ which is infeasible for large datasets. GPs also require choosing a good kernel [23]. Unlike DNNs, inference in GPs remains much more difficult.

To summarize, despite the similarities between the two models, their training methods are fundamentally different. While DNNs employ stochastic optimization, GPs use closed-form updates. How can we relate these seemingly different training procedures in practical settings, e.g., without assuming infinite-width DNNs? In this paper, we provide an answer to this question. We derive theoretical results that relate the solutions and iterations of deep-learning algorithms to GP inference. We do so by first finding a Gaussian posterior approximation (Step A in Fig. 1), then use it to find a linear basis-function model (Step B in Fig. 1) and its corresponding GP (Step C in Fig. 1). We start in the next section with our first theoretical result.

---

2 We can assume that $\delta$ is small enough that it does not affect the DNN’s generalization.
3 Relating Minima of the Loss to GP Inference via Laplace Approximation

In this section, we present theoretical results relating minima of a deep-learning loss (1) to inference in GP models. A local minimizer \( w_\ast \) of the loss (1) satisfies the following first-order and second-order conditions [17]: \( \nabla_w \ell(D, w_\ast) = 0 \) and \( \nabla^2_{ww} \ell(D, w_\ast) \succ 0 \). Deep-learning optimizers, such as RMSprop and Adam, aim to find such minimizers, and our goal is to relate them to GP inference.

Step A (Laplace Approximation): To do so, we will use an approximate inference method called the Laplace approximation [1]. The minima of the loss (1) corresponds to a mode of the Bayesian model: \( p(D, w) := \prod_{i=1}^N e^{-\ell(w)}p(w) \) with prior distribution \( p(w) := \mathcal{N}(w|0, \delta^{-1}I_p) \), assuming that the posterior is well-defined. The posterior distribution \( p(w|D) = p(D, w)/p(D) \) is usually computationally intractable and requires computationally-feasible approximation methods. The Laplace approximation uses the following Gaussian approximation for the posterior:

\[
p(w|D) \approx \mathcal{N}(w|\mu, \Sigma), \quad \text{where } \mu = w_\ast \text{ and } \Sigma^{-1} = \sum_{i=1}^N \nabla^2_{ww} \ell_i(w_\ast) + \delta I_p.
\]

This approximation can be directly built using the solutions found by deep-learning optimizers.

Step B (Linear Model): The next step is to find a linear basis-function model whose posterior is well-defined. The posterior distribution \( p(w|D) \) of the linear model

\[
J(w) := \sum_{i=1}^N \nabla^2_{ww} \ell_i(w) = \sum_{i=1}^N \nabla^2_{ww} \ell_i(w_\ast) + \delta I_p,
\]

where \( \phi_u(x) \) is a \( P \times Q \) feature matrix with \( Q \) as a positive integer, \( v_u(w, x) \) is a \( Q \) length vector, and \( D_u(w, x, y) \) is a \( Q \times Q \) symmetric positive-definite matrix. We will now present results for a specific choice \( \phi_u, v_u, \) and \( D_u \). Our proof trivially generalizes to arbitrary choices of these quantities.

For the loss of form (1), the gradient and Hessian take the following form [15, 17]:

\[
\nabla_w \ell(w) = J_u(x)^\top r_u(x, y), \quad \nabla^2_{ww} \ell(w) = J_u(x)^\top A_u(x, y) J_u(x) + H_f r_u(x, y),
\]

where \( J_u(x) := \nabla_u r_u(x, y) \) is a \( K \times P \) Jacobian matrix, \( r_u(x, y) := \nabla_f \ell(x, y, f) \) is the residual vector evaluated at \( f := r_u(x) \), \( A_u(x, y) := \nabla_f^2 \ell(x, y, f) \) referred to as the noise precision, is the \( K \times K \) Hessian matrix of the loss evaluated at \( f := r_u(x) \), and \( H_f := \nabla^2_{ww} \ell_u(x) \). The similarity between (5) and (6) is striking. In fact, if we ignore the second term for the Hessian \( \nabla^2_{ww} \ell(w) \) in (6), we get the well-known Generalized Gauss-Newton (GGN) approximation [15, 17]:

\[
\nabla^2_{ww} \ell(w) \approx J_u(x)^\top A_u(x, y) J_u(x).
\]

This gives us one choice for the approximation (5) where we can set \( \phi_u(x) := J_u(x)^\top, v_u(x, y) := r_u(x, y), \) and \( D_u(x, y) := A_u(x, y) \).

We are now ready to present our first theoretical result. Consider a Laplace approximation (4) but with the GGN approximation (7) for the Hessian. We refer to this as Laplace-GGN, and denote it by \( \mathcal{N}(w|\mu, \Sigma) \) where \( \Sigma \) is the covariance obtained by using the GGN approximation. We denote the Jacobian, noise-precision, and residual at \( w = w_\ast \) by \( J_u(x), A_u(x, y), \) and \( r_u(x, y) \). We construct a transformed dataset \( \tilde{D} = \{ (x_i, \tilde{y}_i) \}_{i=1}^N \) where the outputs \( \tilde{y}_i \in \mathbb{R}^K \) are equal to \( \tilde{y}_i := J_u(x_i) w_\ast - A_u(x_i, y_i)^{-1} r_u(x_i, y_i) \). We consider the following linear model for \( \tilde{D} \):

\[
\tilde{y} = J_u(x) w + \epsilon, \quad \text{with } \epsilon \sim \mathcal{N}(0, (A_u(x, y))^{-1}) \text{ and } w \sim \mathcal{N}(0, \delta^{-1}I_p).
\]

The following theorem states our result.

**Theorem 1.** The Laplace approximation \( \mathcal{N}(w|\mu, \Sigma) \) is equal to the posterior distribution \( p(w|\tilde{D}) \) of the linear model (8).

A proof is given in Appendix A.1. The linear model uses \( J_u(x) \) as the nonlinear feature map, and the noise precision \( A_u(x, y) \) is obtained using the Hessian of the loss evaluated at \( r_u(x) \). The model is constructed such that its posterior is equal to the Laplace approximation and it exploits the quadratic approximation at \( w_\ast \). We now describe the final step relating the linear model to GPs.

\footnote{For notational convenience, we sometimes use \( \ell(w) \) to denote \( \ell(y, f_w(x)) \).}
Step C (GP Model): To get a GP model, we use the equivalence between the weight-space view shown in (2) and the function-space view shown in (3). With this, we get the following GP regression model whose predictive distribution \( p(f(x)|x_+, D) \) is equal to that of the linear model (8):

\[
\tilde{y} = f(x) + \epsilon, \quad \text{with } f(x) \sim \mathcal{GP} \left( 0, \delta^{-1}J_*(x)J_*(x)^\top \right).
\]  

Note that the kernel here is a multi-dimensional \( K \times K \) kernel. The steps A, B, and C together convert a DNN defined in the weight-space to a GP defined in the function-space. We refer to this approach as “DNN2GP”.

The resulting GP predicts in the space of outputs \( \tilde{y} \) and therefore results in different predictions than the DNN, but it is connected to it through the Laplace approximation as shown in Theorem 1. In Appendix B, we describe prediction of the outputs \( y \) (instead of \( \tilde{y} \)) using this GP. Note that our approach leads to a heteroscedastic GP which could be beneficial. Even though our derivation assumes a Gaussian prior and DNN model, the approach holds for other types of priors and models.

Relationship to NTK: The GP kernel in (9) is the Neural Tangent Kernel \(^4\) (NTK) \(^8\) which has desirable theoretical properties. As the width of the DNN is increasing to infinity, the kernel converges in probability to a deterministic kernel and also remains asymptotically constant during training. Our kernel is the NTK defined at \( w_\ast \) and is expected to have similar properties. It is also likely that, as the DNN width is increased, the Laplace-GGN approximation has similar properties as a GP posterior, and can be potentially used to improve the performance of DNNs. For example, we can use GPs to tune hyperparameters of DNNs. The function-space view is also useful to understand relationships between data samples. Another advantage of our approach is that we can derive kernels other than the NTK. Any approximation of the form (5) will always result in a linear model similar to (8).

Accuracy of the GGN approximation: This approximation is accurate when the model \( f_w(x) \) can fit the data well, in which case the residuals \( r_w(x,y) \) are close to zero for all training examples and the second term in (6) goes to zero \([2, 15, 17]\). The GGN approximation is a convenient option to derive DNN2GP, but, as it is clear from (5), other types of approximations can also be used.

4 Relating Iterations of a Deep-Learning Algorithm to GP Inference via VI

In this section, we present theoretical results relating iterations of an RMSprop-like algorithm to GP inference. The RMSprop algorithm \([21]\) uses the following updates (all operations are element-wise):

\[
w_{t+1} \leftarrow w_t - \alpha_t \left( \sqrt{\text{St}_{t+1}} + \Delta \right)^{-1} \hat{g}(w_t), \quad w_t \leftarrow (1 - \beta_t)w_t + \beta_t \left( \hat{g}(w_t) \right)^2,
\]  

where \( t \) is the iteration, \( \alpha_t > 0 \) and \( 0 < \beta_t < 1 \) are learning rates, \( \Delta > 0 \) is a small scalar, and \( \hat{g}(w) \) is a stochastic-gradient estimate for \( \bar{\ell}(D, w) \) obtained using minibatches. Our goal is to relate the iterates \( w_t \) to GP inference using our DNN2GP approach, but this requires a posterior approximation defined at each \( w_t \). We cannot use the Laplace approximation because it is only valid at \( w_\ast \). We will instead use a version of RMSprop proposed in \([10]\) for variational inference (VI), which enables us to construct a GP inference problem at each \( w_t \).

Step A (Variational Inference): The variational online-Newton (VON) algorithm proposed in \([10]\) optimizes the variational objective, but takes an algorithmic form similar to RMSprop (see a detailed discussion in \([10]\)). Below, we show a batch version of VON, derived using Eq. (54) in \([10]\):

\[
\mu_{t+1} \leftarrow \mu_t - \beta_t(S_{t+1} + \delta I \rho)^{-1} \mathbb{E}_{q_t(w)} \left[ \nabla_w \bar{\ell}(D, w) \right],
\]  

\[
S_{t+1} \leftarrow (1 - \beta_t)S_t + \beta_t \sum_{i=1}^N \mathbb{E}_{q_i(w)} \left[ \nabla^2_{w_\nu} \ell_i(w) \right],
\]  

where \( S_t \) is a scaling matrix similar to the scaling vector \( s_t \) in RMSprop, and the Gaussian approximation at iteration \( t \) is defined as \( q_t(w) := \mathcal{N}(w|\mu_t, S_t) \) where \( S_t := (S_t + \delta I \rho)^{-1} \). Since there are no closed-form expressions for the expectations, the Monte Carlo (MC) approximation is used.

Step B (Linear Model): As before, we assume the choices for (5) obtained by using the GGN approximation (7). We consider the variant for VON where the GGN approximation is used for the Hessian and MC approximation is used for the expectations with respect to \( q_t(w) \). We call this the

---

\(^4\)The NTK corresponds to \( \delta = 1 \) which implies a standard normal prior on weights.
Variational Online GGN or VOGGN algorithm. A similar algorithm has recently been used in [19] where it shows competitive performance to Adam and SGD.

We now present a theorem relating iterations of VOGGN to linear models. We denote the Gaussian approximation obtained at iteration $t$ by $\tilde{q}_t(w) := \mathcal{N}(w|\mu_t, \Sigma_t)$ where $\Sigma_t$ is used to emphasize the GGN approximation. We present theoretical results for VOGGN with 1 MC sample which is denoted by $w_t \sim \tilde{q}_t(w)$. Our proof in Appendix A.2 discusses a more general setting with multiple MC samples. Similarly to the previous section, we first define a transformed dataset: $\mathcal{D}_t := \{(x_i, y_{i,t})\}_{i=1}^n$ where $y_{i,t} := J_{w_t}(x_i)w_t - \Lambda_{w_t}(x_i, y_i)^{-1}r_{w_t}(x_i, y_i)$, and then a linear basis-function model:

$$\tilde{y}_t = J_{w_t}(x)w + \epsilon, \text{ with } \epsilon \sim \mathcal{N}(0, (\beta_t \Lambda_{w_t}(x, y))^{-1}) \text{ and } w \sim \mathcal{N}(m_t, V_t)$$  \hspace{1cm} (13)

with $V_t^{-1} := (1 - \beta_t)\Sigma_t^{-1} + \beta_t \delta I_P$ and $m_t := (1 - \beta_t)V_t\Sigma_t^{-1}w_t$. The model is very similar to the one obtained for Laplace approximation, but is now defined using the iterates $w_t$ instead of the minimum $w_*$. The prior over $w$ is not the standard Gaussian anymore, rather a correlated Gaussian derived from $q_t(w)$. The theorem below states the result (a proof is given in Appendix A.2).

**Theorem 2.** The Gaussian approximation $\mathcal{N}(w|w_{t+1}, \Sigma_{t+1})$ at iteration $t + 1$ of the VOGGN update is equal to the posterior distribution $p(w|\mathcal{D}_t)$ of the linear model (13).

**Step C (GP Model):** The linear model (13) has the same predictive distribution as the GP below:

$$\tilde{y}_t = \Phi_t(x) + \epsilon, \text{ with } \Phi_t(x) \sim \mathcal{GP}(J_{w_t}(x)m_t, J_{w_t}(x)V_tJ_{w_t}(x)^\top).$$ \hspace{1cm} (14)

The kernel here is similar to the NTK but now there is a covariance term $\Phi_t$ which incorporates the effect of the previous $q_t(w)$ as a prior. Our DNN2GP approach shows that one iteration of VOGGN in the weight-space is equivalent to inference in a GP regression model defined in a transformed function-space with respect to a kernel similar to the NTK. This can be compared with the results in [8], where learning by plain gradient descent is shown to be equivalent to kernel gradient descent in function-space. Similarly to the Laplace case, the resulting GP predicts in the space of outputs $\tilde{y}_t$, but predictions for $y_t$ can be obtained using a method described in Appendix B.

**A Deep-Learning Optimizer Derived from VOGGN:** The VON algorithm, even though similar to RMSprop, does not converge to the minimum of the loss. This is because it optimizes the variational objective. Fortunately, a slight modification of this algorithm gives us a deep-learning optimizer which is similar to RMSprop but is guaranteed to converge to the minimum of the loss. For this, we approximate the expectations in the updates (11)-(12) at the mean $\mu_t$. This is called the zeroth-order delta approximation; see Appendix A.6 in [9] for details of this method. Using this approximation and denoting the mean $\mu_t$ by $w_t$, we get the following update:

$$w_{t+1} \leftarrow w_t - \beta_t(\hat{S}_{t+1} + \delta I_P)^{-1}\nabla_w \ell(D, w_t), \quad \hat{S}_{t+1} \leftarrow (1 - \beta_t)\hat{S}_t + \beta_t \sum_{i=1}^N \nabla^2 \ell_i(w_t)$$.

We refer to this as Online GGN or OGGN method. A fixed point $w_*$ of this iteration is also a minimizer of the loss since we have $\nabla_w \ell(D, w_*) = 0$. Unlike RMSprop, at each iteration, we still get a Gaussian approximation $q_t(w) := \mathcal{N}(w|w_t, \Sigma_t)$ with $\Sigma_t := (\hat{S}_t + \delta I_P)^{-1}$. Therefore, the posterior of the linear model from Theorem (2) is equivalent to $\tilde{q}_t$ when $\hat{S}_t$ is replaced by $\Sigma_t$ (see Appendix A.3). In conclusion, by using VI in our DNN2GP approach, we are able to relate the iterations of a deep-learning optimizer to GP inference.

**Implementation of DNN2GP:** In practice, both VOGGN and OGGN are computationally more expensive than RMSprop because they involve computation of full covariance matrices. To address this issue, we simply use the diagonal versions of these algorithms discussed in [10, 19]. Specifically, we use the VOGN and OGN algorithms discussed in [19]. This implies that $V_t$ is a diagonal matrix and the GP kernel can be obtained without requiring any computation of large matrices. Only Jacobian computations are required. In our experiments, we also resort to computing the kernel over a subset of data instead of the whole data, which further reduces the cost.
We consider a version of the Snelson dataset [20] where, to assess the ‘in-between’ uncertainty, we remove the data points between $x$ and $y$. The rows and columns containing 300 data examples are grouped according to the classes. The kernel $K_J$ for each class with an individual GP and then summing the kernels of these GPs. We also visualize the GP kernel matrix clearly shows the correlations learned by the DNN. As expected, each row in the posterior mean also reflects that the classes are correctly classified (DNN test accuracy is 99%).

Fig. 4d shows the two components of the predictive variances that can be interpreted as “aleatoric” and “epistemic” uncertainty. As shown in Eq. (48) in Appendix B.2, for a multiclass classification loss, the variance of the prediction of a label at an input $x_*$ is equal to $\Lambda_* (x_*) + \Lambda_*(x_*) J_* (x_*) \Sigma J_* (x_*) ^\top \Lambda_*(x_*)$. Similar to the linear basis function model, the two terms here have an interpretation (e.g., see Eq. 3.59 in [1]). The first term can be interpreted as the aleatoric uncertainty (label noise), while the second term takes a form that resembles the epistemic uncertainty...
Figure 4: DNN2GP kernels, posterior means and uncertainties with LeNet5 of 300 samples on binary MNIST in Fig. (c), MNIST in Fig. (a), and CIFAR-10 in Fig. (b,d). The colored regions on the y-axis mark the classes. Fig. (a) shows the kernel and the predictive mean for the Laplace approximation, which gives 99% test accuracy. We see in the kernel that examples with same class labels are correlated. Fig. (c) shows the same for binary MNIST trained only on digits 0 and 1 by using VI. The kernel clearly shows the out-of-class predictive behavior where predictions are not certain. Fig. (b) and (d) show the Laplace-GP on the more complex CIFAR-10 data set where we obtain 68% accuracy. Fig. (d) shows the two components of the predictive variance for CIFAR-10 that can be interpreted as epistemic (left) and aleatoric (right) uncertainties. The estimated epistemic uncertainty is much lower than the aleatoric uncertainty, implying that the model is not flexible enough. This is plausible since the accuracy of the model is not too high (merely 68%). (model noise). Fig. 4d shows these for CIFAR-10 where we see that the uncertainty of the model is low (left) and the label noise rather high (right). This interpretation implies that the model is unable to flexibly model the data and instead explains it with high label noise.

In Fig. 4c, we study the kernel for classes outside of the training dataset using VI. We train LeNet-5 on digits 0 and 1 with VOGN and visualize the predictive mean and kernel on all 10 classes denoted by differently colored regions on the y-axis. We can see that there are slight correlations to the out-of-class samples but no overconfident predictions. In contrast, the pattern between 0 and 1 is quite strong. The kernel obtained with DNN2GP helps to interpret and visualize such correlations.

5.3 Tuning the Hyperparameters of a DNN Using the GP Marginal Likelihood

In this section, we demonstrate the tuning of DNN hyperparameters by using the GP marginal likelihood on a real and synthetic regression dataset. In the deep-learning literature, this is usually done using cross-validation. Our goal is to demonstrate that with DNN2GP we can do this by simply computing the marginal likelihood on the training set.

We generate a synthetic regression dataset ($N = 100$; see Fig. 5) where there are a few data points around $x = 0$ but plenty away from it. We fit the data by using a neural network with single hidden layer of 20 units and tanh nonlinearity. Our goal is to tune the regularization parameter $\delta$ to trade-off underfitting vs overfitting. Fig. 5b and 5c show the train log marginal-likelihood obtained with the GP obtained by DNN2GP, along with the test and train mean-square error (MSE) obtained using a point estimate. Black stars indicate the hyperparameters chosen by using the test loss and log marginal
We hope that our work enables the community to conduct further research on such problems.

We use the Laplace approximation and tune one parameter at a time while keeping the others fixed (we use respectively \( \sigma = 0.64, \delta = 30 \) and \( \sigma = 0.64, \delta = 3 \), 1 hidden layer). Similarly to the synthetic data case, the train marginal-likelihood selects hyperparameters that give low test error. These experiments show that the DNN2GP framework can be useful to tune DNN hyperparameters, although this needs to be confirmed for larger networks than we used here.

### 6 Discussion and Future Work

In this paper, we present theoretical results connecting approximate inference on DNNs to GP posteriors. Our work enables the extraction of feature maps and GP kernels by simply training DNNs. It provides a natural way to combine the two different models.

Our hope is that our theoretical results will facilitate further research on combining strengths of DNNs and GPs. A computational bottleneck is the Jacobian computation which prohibits application to large problems. There are several ways to reduce this computation, e.g., by choosing a different type of GGN approximation that uses gradients instead of the Jacobians. Exploration of such methods is a future direction that needs to be pursued.

Exact inference on the GP model we derive is still computationally infeasible for large problems. However, further approximations could enable inference on bigger datasets. Finally, our work opens many other interesting avenues where a combination of GPs and DNNs can be useful such as model selection, deep reinforcement learning, Bayesian optimization, active learning, interpretation, etc.

We hope that our work enables the community to conduct further research on such problems.
Acknowledgements

We would like to thank Kazuki Osawa (Tokyo Institute of Technology), Anirudh Jain (RIKEN), and Runa Eschenhagen (RIKEN) for their help with the experiments. We would also like to thank Matthias Bauer (DeepMind) for discussions and useful feedback. Many thanks to Roman Bachmann (RIKEN) for helping with the visualization in Fig. 1. We also thank Stephan Mandt (UCI) for suggesting the marginal likelihood experiment. We thank the reviewers and the area chair for their feedback as well. We are also thankful for the RAiDEN computing system and its support team at the RIKEN Center for Advanced Intelligence Project which we used extensively for our experiments.

References


A Proofs

In this section, we prove the theorems presented in the main text.

A.1 Proof of Theorem 1

We begin with the Gaussian approximation of the Laplace approximation. We will then express its
natural parameters in terms of the gradient and Hessians. Application of the GGN approximation
and some further manipulation will show that the distribution correspond to the posterior of a linear
model.

We start with the Laplace approximation (4) and express its natural parameters in terms of the gradient
and Hessians. We denote the natural-parameters of this Gaussian approximation \( N(w|\mu, \Sigma) \) by
\[ \eta := \{ \Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1} \}. \]
In (4), the second natural parameter is set to the following which is written in terms of the Hessian:
\[ -\frac{1}{2}\Sigma^{-1} = -\frac{1}{2} \left[ \sum_{i=1}^{N} \nabla_{ww}^2 \ell_i(w_*) + \delta \right] \]  
(15)

We can also express the first natural parameter in terms of the gradient and Hessians as shown below.

We use the first-order stationary condition, that is, \( \nabla w \bar{\ell}(D, w_*) = 0 \). Adding \( \Sigma^{-1}\mu \) to both sides of
this condition, we get the following:
\[ \Sigma^{-1}\mu = -\nabla w \ell(D, w_*) + \Sigma^{-1} \mu \]  
(16)
\[ = -\sum_{i=1}^{N} \nabla w \ell_i(w_*) - \delta w_* + \left[ \sum_{i=1}^{N} \nabla_{ww}^2 \ell_i(w_*) + \delta \right] w_* \]  
(17)
\[ = \sum_{i=1}^{N} \left[ -\nabla w \ell_i(w_*) + \nabla_{ww}^2 \ell_i(w_*) w_* \right], \]  
(18)

where in the second step, we substitute \( \mu \) by \( w_* \), and also use (15). With this, both natural parameters
are now expressed in terms of the gradient and Hessian.

We will now substitute these in the Laplace approximation, denoted by \( q_L(w) := N(w|\mu, \Sigma) \). Using
(15) and (18), we get the following expression:
\[ q_L(w) = \frac{1}{\sqrt{(2\pi)^r |\Sigma|}} \exp \left[ -\frac{1}{2} (w - \mu)^T \Sigma^{-1} (w - \mu) \right] \]  
(19)
\[ \propto \exp \left[ -\frac{1}{2} w^T (\Sigma^{-1}) w + w^T \left( \Sigma^{-1} \right) \mu \right] \]  
(20)
\[ = \exp \left[ -\frac{1}{2} \nabla \Sigma^{-1} w \right] \prod_{i=1}^{N} \exp \left[ -\frac{1}{2} \nabla w \ell_i(w_*) w + w^T \left\{ -\nabla w \ell_i(w_*) + \nabla_{ww}^2 \ell_i(w_*) w_* \right\} \right], \]  
(21)

where in the last line we use (15) and (18).

Now, we will employ the GGN approximation (7) which gives us the Laplace-GGN approximation
\( \tilde{N}(w|\mu, \tilde{\Sigma}) \), shown below:
\[ \exp \left( -\frac{1}{2} \nabla \tilde{\Sigma} w \right) \prod_{i=1}^{N} \exp \left[ -\frac{1}{2} \nabla w J_i(x_i)^T \Lambda_{i,*} J_i(x_i) w + w^T J_i(x_i)^T \left\{ \Lambda_{i,*} J_i(x_i) w_* - r_{i,*} \right\} \right]. \]  
(22)

where for notational convenience we have denoted \( \Lambda_{i,*} := \Lambda_i(x_i, y_i) \) and \( r_{i,*} := r_i(x_i, y_i) \).

A key point here is that each term in the product over \( i \) in (22) is proportional to a Gaussian
distribution, provided that \( \Lambda_{i,*} > 0 \), which is the case since we assume the loss function to be strictly
convex. We will now express each term in the product, as a likelihood over a pseudo-output defined
as \( \tilde{y}_i := J_i(x_i) w_* - \Lambda_{i,*}^{-1} r_{i,*} \). Using this and completing the square within each term in the product
over \( i \) in (22), we get the following:
\[ \tilde{q}_L(w) := N(w|\mu, \tilde{\Sigma}) \propto N(w|0, \delta^{-1} I_P) \prod_{i=1}^{N} N(\tilde{y}_i | J_i(x_i) w, \Lambda_{i,*}^{-1}). \]  
(23)
The right hand side of the above equation is proportional to the posterior distribution \( p(w|\tilde{D}) \) given a transformed dataset \( \tilde{D} := \{(x_i, \tilde{y}_i)\}_{i=1}^N \) of a linear basis-function model \( \tilde{y} = J_s(x)w + \epsilon \) with Gaussian noise \( \epsilon \sim \mathcal{N}(0, (\Lambda_s(x, y))^{-1}) \) and prior distribution \( w \sim \mathcal{N}(0, \delta^{-1}I_p) \). This completes the proof.

It is easy to see that the same proof works when using the approximations shown in (5). In that case, only the steps from (22) need to be modified. The proof also holds when a prior other than Gaussian and a model other than DNN is used.

A.2 Proof of Theorem 2

Similarly to the previous section, we start by writing the Gaussian approximation for VON. We will express its natural parameters in terms of the gradient and Hessians. A GGN approximation and some manipulation will show that the distributions found by VON correspond to posteriors of linear models.

The Gaussian approximation at the \( t \) th iteration of VON is denoted by \( q_t(w) := \mathcal{N}(\mu_t, \Sigma_t) \) where \( \Sigma_t := (\mathbf{S}_t + \delta I_p)^{-1} \) is obtained from \( \mathbf{S}_t \). Using this, we can rewrite the updates (11) and (12) in terms of \( \mu_t \) and \( \Sigma_t^{-1} \) as follows:

\[
\mu_{t+1} = \mu_t - \beta_t \Sigma_{t+1} \left[ \sum_{i=1}^{N} E_{q_t(w)} [\nabla_w \ell_i(w)] + \delta \mu_t \right], \tag{24}
\]

\[
\Sigma_{t+1}^{-1} = (1 - \beta_t) \Sigma_t^{-1} + \beta_t \left[ \sum_{i=1}^{N} E_{q_t(w)} [\nabla^2_{ww} \ell_i(w)] + \delta I_p \right]. \tag{25}
\]

It is again more convenient if we can have an update formula for the natural-parameters of the Gaussian distribution \( \mathcal{N}(\mu_t, \Sigma_t) \), denoted by \( \eta_t := \{ \Sigma_t^{-1} \mu_t, -\frac{1}{2} \Sigma_t^{-1} \} \). So we use similar techniques to find an update for \( \eta_t \). In addition, since there are no closed-form expressions for the expectations above, we use \( S \) number of samples \( w_t^{(s)} \sim q_t(w) \), for \( s = 1, 2, \ldots, S \), and use Monte Carlo (MC) approximation.

Given (25), the update corresponding to the second natural-parameter is obvious and given by:

\[
-\frac{1}{2} \Sigma_t^{-1} = (1 - \beta_t) \left[ -\frac{1}{2} \Sigma_t^{-1} \right] - \frac{1}{2} \beta_t \left[ \sum_{i=1}^{N} E_{q_t(w)} [\nabla^2_{ww} \ell_i(w)] + \delta I_p \right] \tag{26}
\]

\[
\approx (1 - \beta_t) \left[ -\frac{1}{2} \Sigma_t^{-1} \right] - \frac{1}{2} \beta_t \left[ \frac{1}{S} \sum_{i,s=1}^{N,S} \nabla^2_{ww} \ell_i(w_t^{(s)}) + \delta I_p \right]. \tag{27}
\]

where we have used an MC approximation in the second step.

To write the update for the first natural-parameter, we multiply (24) by \( \Sigma_t^{-1} \) and get

\[
\Sigma_t^{-1} \mu_{t+1} = \Sigma_t^{-1} \mu_t - \beta_t \left[ \sum_{i=1}^{N} E_{q_t(w)} [\nabla_w \ell_i(w)] + \delta \mu_t \right] \tag{28}
\]

\[
= (1 - \beta_t) \left[ \Sigma_t^{-1} \mu_t \right] + \beta_t \left[ -E_{q_t(w)} [\nabla_w \ell_i(w)] + E_{q_t(w)} [\nabla^2_{ww} \ell_i(w)] \right] \mu_t \tag{29}
\]

\[
\approx (1 - \beta_t) \left[ \Sigma_t^{-1} \mu_t \right] + \beta_t \frac{1}{S} \sum_{i,s=1}^{N,S} \left[ -\nabla_w \ell_i(w_t^{(s)}) + \nabla^2_{ww} \ell_i(w_t^{(s)}) \right] \mu_t, \tag{30}
\]

where in the second step, we replaced \( \Sigma_t^{-1} \mu_t \) in the first term by (25). The posterior approximation \( q_{t+1}(w) \) at time \( t + 1 \) can be written in terms of natural parameters as shown below:

\[
q_{t+1}(w) = \frac{1}{\sqrt{(2\pi)^p|\Sigma_{t+1}|}} \exp \left[ -\frac{1}{2} (w - \mu_{t+1})^T \Sigma_{t+1}^{-1} (w - \mu_{t+1}) \right] \tag{31}
\]

\[
\propto \exp \left[ -\frac{1}{2} w^T \left( \Sigma_{t+1}^{-1} \right) w + w^T (\Sigma_{t+1}^{-1} \mu_{t+1}) \right]. \tag{32}
\]
By substituting the natural parameters from (27) and (30), we get the following update for \( q_{t+1}(w) \), expressed in terms of the MC samples:

\[
q_{t+1}(w) \propto p(w)^{\beta_t} q_t(w)^{1-\beta_t} \times \prod_{i,s=1}^{N,S} \exp \left[ -\frac{\beta_t}{2S} w^T \nabla w \ell_i(w_t(s)) + \frac{\beta_t w_i^T}{S} \left\{ -\nabla w \ell_i(w_t(s)) + \nabla^2 w \ell_i(w_t(s)) \mu_j \right\} \right].
\]  

(33)

where \( p(w) = \mathcal{N}(w|0, \delta^{-1}I_P) \) is the prior distribution. For the product of posterior approximation at time \( t \) and prior in (33), we obtain the following unnormalized Gaussian

\[
p(w)^{\beta_t} q_t(w)^{1-\beta_t} = \mathcal{N}(w|0, \delta^{-1}I_P)^{\beta_t} \mathcal{N}(w|\mu_t, \Sigma_t)^{1-\beta_t} \propto \mathcal{N}(w|m_t, V_t),
\]

where \( V_t \) and \( m_t \) are given by

\[
V_t^{-1} := (1 - \beta_t) \Sigma_t^{-1} + \beta_t \delta \mathbf{1}_P, \quad m_t := (1 - \beta_t)V_t^{-1} \mu_t.
\]

(34)

(35)

Next, for the product over \( i \) and \( s \) in (33), we employ the GGN approximation (7) and get

\[
\tilde{q}_{t+1}(w) \propto \mathcal{N}(w|m_t, V_t) \times \prod_{i,s=1}^{N,S} \exp \left[ -w^T \frac{\beta_t A_{i,s,t}}{2S} J_{s,t}(x_i) w + \frac{\beta_t w_i^T}{S} \left\{ A_{i,s,t} J_{s,t}(x_i) \mu_t - r_{i,s,t} \right\} \right].
\]

(36)

where we have defined \( J_{s,t}(x_i) := J_{w_i(s)}(x_i), r_{i,s,t} := r_{w_i(s)}(x_i, y_t), \) and \( A_{i,s,t} := A_{w_i(s)}(x_i, y_t). \) The notation \( \tilde{q}_{t+1}(w) \) is used to emphasize that GGN approximation is used in this update.

We are now ready to express each term in the product above as a Gaussian distribution. First, we define three quantities: \( J_t(x), r_t(x, y) \) and \( A_t(x, y) \) which are obtained by concatenating all the sampled Jacobians, residuals, and noise-precision matrices:

\[
J_t(x) := \begin{bmatrix}
J_{w_i(1)}(x) \\
J_{w_i(2)}(x) \\
\vdots \\
J_{w_i(S)}(x)
\end{bmatrix}, \quad r_t(x, y) := \begin{bmatrix}
r_{w_i(1)}(x, y) \\
r_{w_i(2)}(x, y) \\
\vdots \\
r_{w_i(S)}(x, y)
\end{bmatrix},
\]

(37)

\[
A_t(x, y) := \begin{bmatrix}
A_{w_i(1)}(x, y) & 0 & \ldots & 0 \\
0 & A_{w_i(2)}(x, y) & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & A_{w_i(S)}(x, y)
\end{bmatrix}.
\]

(38)

Using this, we define a transformed output of length \( KS \times 1 \) as

\[
\tilde{y}_{t,t} := J_t(x_i) \mu_t - A_t(x_i, y_t)^{-1} r_t(x_i, y_t).
\]

(39)

The distribution \( \tilde{q}_{t+1}(w) \) defined in (36) can then be expressed as the following:

\[
\tilde{q}_{t+1}(w) \propto \mathcal{N}(w|m_t, V_t) \prod_{i=1}^{N} \mathcal{N}(\tilde{y}_{t,t}, J_t(x_i) w, S(\beta_t A_t(x_i, y_t))^{-1}).
\]

(40)

As before, we can show that this distribution is equal to the posterior distribution of a linear on a transformed dataset defined as \( \tilde{D}_t := \{(x_i, \tilde{y}_{t,t})\}_{i=1}^{N} \). To model such outputs, we define a linear model for an output \( \tilde{y}_t \in \mathbb{R}^{KS} \) defined as follows:

\[
\tilde{y}_t = J_t(x) w + \epsilon_t, \text{ with } \epsilon_t \sim \mathcal{N}(0, S(\beta_t A_t(x, y))^{-1}), \text{ and } w \sim \mathcal{N}(m_t, V_t).
\]

(41)

The theorem presented in the main text is a simpler version of this theorem where \( S = 1 \). This completes the proof.
A.3 Linear Model Corresponding to OGGN

In OGGN, we evaluate the gradient and Hessian at the mean $\mu_t$ defined to be equal to the current iterate $w_t$. This corresponds to $S = 1$ in the setting described in the proof of theorem 2 (see Appendix A.2) with $w_1^{(1)} := w_t$. Therefore, the linear model is the same as before but with $J_t(x), r_t(x, y)$ and $\Lambda_t(x, y)$ defined at $w_t$.

B Approximating Posterior Predictive with DNN2GP Approach

Typically, we can always predict using Monte Carlo sampling from the Gaussian approximation, however, this might be too noisy sometimes. In this section, we show how DNN2GP approach enables us to directly use the GP regression model for approximating the posterior predictive distribution.

We elaborate on the method for Laplace approximation but this can be generalized to VI as briefly explained in subsection B.3.

Given a test input, denoted by $x_*$, we first compute the feature map $J_*(x_*)^T$. Using the linear model found in the DNN2GP approach, we can compute the posterior predictive distribution of the output, which we denote by $y_*$. However, to be able to compute the predictive distribution for the true output $y_*$, we need to invert the map from $y_*$ to $\tilde{y}_*$. The expressions for this map can be obtained by using the definition of the transformed output $\tilde{y}_* := J_*(x_*)w_s - \Lambda_s(x_*)^{-1}r_s(x_*, y_*)$. We demonstrate this for two common cases of squared loss and logistic loss.

B.1 Laplace Approximation and Squared Loss

Consider the squared loss, $\ell(y, f_w(x)) = \frac{1}{2}\|y - f_w(x)\|^2$ with $\sigma^2$ as the noise variance. According to section 3, in this case, we have $r_s(x, y) := \sigma^{-2}(f_w(x) - y)$ and $\Lambda_s(x, y) := \sigma^{-2}I_K$. Using these expressions in the definition for $\tilde{y} := J_*(x)w_s - \Lambda_s(x, y)^{-1}r_s(x, y)$, we get the following map for the test input $x_*$:

$$\tilde{y}_* = J_*(x_*)w_s - (f_w_*(x_*) - y_*) \quad (42)$$

$$\Rightarrow y_* = \tilde{y}_* + f_w_*(x_*) - J_*(x_*)w_s \quad (43)$$

Given a predictive distribution for $\tilde{y}_*$ computed by the linear model (8) with the posterior distribution $N(w_s, \Sigma)$, we can therefore derive the predictive distribution for $y_*$. In the example above, the predictive variance of $\tilde{y}_*$ and $y_*$ will be the same, while the predictive mean of $y_*$ is obtained by adding $f_w_*(x_*) - J_*(x_*)w_s$ to the mean of $\tilde{y}_*$. The result is as follows

$$y_* \sim N \left( r_*(x_*, J_*(x_*)\Sigma J_*(x_*)^T + \sigma^2I_K \right) \quad (44)$$

We use this technique to compute the predictive distribution in Fig. 3 (labeled as ‘DNN2GP’ in the plots).

B.2 Laplace Approximation and Logistic Loss

The procedure above for inversion of maps generalizes to other loss functions derived using generalized linear models. We need to assume that the loss corresponds to a log probability distribution, i.e., $\ell(y, f_w(x)) := -\log p(y|f_w(x))$ where $h(\cdot)$ is a link function. We now describe this for a Bernoulli distribution $y_i \in \{0, 1\}$ using the results in section 3.

Similarly to the squared-loss case, we need to write $\tilde{y}$ in terms of the true output $y$. For a Bernoulli likelihood, the link function is $\sigma(f_w(x)) := p_s(x)$ where $s$ is the sigmoid function, the residual is $r_s(x, y) = p_s(x) - y$, and the noise precision is $\Lambda_{w_s}(x, y) = p_s(x)(1 - p_s(x)) := \lambda_s(x)$. We again use the definition for the transformed output and write the map for the test input $x_*:

$$\tilde{y}_* = J_*(x_*)w_s - \lambda_s(x_*)^{-1}(p_s(x_*) - y_*) \quad (45)$$

$$\Rightarrow y_* = p_s(x_*) + \lambda_s(x_*)\tilde{y}_* - \lambda_s(x_*)J_*(x_*)w_s \quad (46)$$

Given the predictive distribution over $\tilde{y}_*$ at the test input $x_*$, we can then compute the corresponding distribution over $y_*$. The predictive distribution of $\tilde{y}_*$ in the linear model (8) with the posterior distribution $N(w_s, \Sigma)$ is given as follows:

$$y_* \sim N \left( \tilde{y}_*|J_*(x_*)w_s, \lambda_s(x_*)^{-1} + J_*(x_*)\Sigma J_*(x_*)^T \right) \quad (47)$$
Therefore, using the map (46), we get the following predictive distribution over $y^*$:

$$y^*|x^*, D \sim N \left( y^*|\sigma(f_{w^*}(x^*)), \lambda^*(x^*)^2J_{\text{c}}(x^*)\Sigma J_{\text{c}}(x^*)^\top \right).$$ (48)

Similar to the linear basis function model, the two terms in the predictive variance have an interpretation (e.g., see [1] Eq. 3.59). The first term can be interpreted as the aleatoric uncertainty (label noise), while the second term takes a form that resembles the epistemic uncertainty (model noise). Such interpretation is possible due to the conversion of a DNN to a linear-basis function model in our DNN2GP framework.

This approach can be similarly written for other Gaussian approximations. It can also be generalized to loss functions obtained using the generalized linear model. The inversion of the map is possible whenever the link function $h(.)$ is invertible.

### B.3 Generalization to VI

For the VOGGN update with one MC sample, we can use the same procedure as above. The same is true for OGGN since one MC sample is replaced by the mean. For VOGGN with multiple MC samples, we get $S$ such maps. Each of those maps give us a prediction, denote it by $\tilde{y}_{s,t}$ for sample $s$ at iteration $t$. To obtain the final prediction, we can use the average all predictions $\tilde{y}_{s,t}$ over $s = 1, 2, 3, \ldots, S$ to get the predictive distribution for $y^*$.

### C Additional Results

In this appendix, we provide additional figures to the ones presented in Sec. 5.2.

#### C.1 Further Posteriors and Kernels for MNIST and CIFAR

Fig. 7 is similar to Fig. 4a but uses the variational approximation instead of a Laplace approximation. While the posterior mean on MNIST shows very similar structure for both approximations, the kernel shows some interesting differences. There are many more negative correlations between examples from different classes in the kernel corresponding to the variational approximation. The posterior mean on CIFAR-10 has similar structure yet it appears to exhibit higher uncertainty. In Fig. 8, we show the kernel matrix on 300 data points of CIFAR-10 with the respective class labels. The kernel is computed for both the Laplace and variational approximation but shows less structure than that of the MNIST dataset.

![GP kernel matrix and posterior mean](image)

Figure 7: This figure visualizes the GP kernel matrix and posterior mean for LeNet5 trained with VOGN on MNIST (left) and CIFAR-10 (right). The kernel matrix clearly shows the correlations learned by the DNN. A higher posterior mean is assigned to the correct label which reflects the accuracy obtained by the DNN.

#### C.2 Uncertainties according to DNN2GP for Classification

In this section we present a toy example for the classification task in line with the regression experiment in Fig. 3. We use the reparameterization introduced in App. B.2, in particular Eq. (48).
Figure 8: GP kernels due to Laplace and variational approximation for neural networks on CIFAR-10. The kernels show slight traces of structure but are not as significant as the ones presented on MNIST in Sec. 5.

Figure 9: This figure demonstrates the decomposition of predictive variances due to the reparameterization introduced in App. B.2 on a binary toy classification task (red vs. blue half moons). We plot the quantities of Eq. (48) in figures (a)-(c): (a) is the prediction of a trained NN while the sum of (b) and (c) give us the posterior predictive uncertainties. Around the decision boundary, the label noise (b) is high and remains unchanged further from the data while the predictive uncertainty is low where supported by data and strongly grows away from it. Here, the model fits the data well in contrast to Fig. 4d where the model is unable to do so which results in high estimated label noise.

We train a neural network with single hidden layer of 10 units and tanh activation to fit the non-linear decision boundary. We have $\delta = 0.26$ and train on 100 samples for 5000 full-batch epochs. Fig. 9 shows how the reparameterization allows to decompose predictive variance into label noise due to the decision boundary, see. (b), and model uncertainty, see (c), that grows away from the data.

D Author Contributions Statement

Author List: Mohammad Emtiyaz Khan, Alexander Immer, Ehsan Abedi, Maciej Korzepa.

M.E.K. conceived a rough idea using the gradients and Hessians of the loss, and wrote the first version of the proofs. A.I. and E.A. made major corrections to M.E.K.’s original version and introduced version used in the final paper. They also came up with the prediction method for DNN2GP. E.A. formalized the NTK connection, and extensively studied its connection to the GP posterior. A.I. did most of the experiments and introduced the necessary reparameterization for applications. M.K. helped on the hyperparameter-tuning experiments, as well as with the visualizations. M.K. did the regression uncertainty experiment with some help from E.A. and A.I.

M.E.K. wrote the main content of the paper. E.A. wrote all the proofs, and A.I. and M.K. summarized the experiment section. All the authors proof-read the paper and revised it.
E Camera-Ready Version vs the Submitted Version

We made several changes taking reviewers feedback into account.

1. The writing and organization of the papers were modified to emphasize that we are able to relate the iterations of a deep-learning algorithm to GP inference.
2. To improve clarity, Fig. 1 was added as a summary of our approach. The writing was modified to follow Step A, B, and C given in Fig. 1.
3. Titles of Section 3 and 4 were changed to emphasize relationship to "solutions and iterations" of a deep-learning algorithm.
4. Theorem 1 and 2 were simplified to focus only on the posterior of linear model only. Relation to GP is discussed separately.
5. Experiment on GP regression was modified to focus on uncertainty instead of the width of the DNN.
6. Visualization of the GP predictive uncertainty and noise was added on top of predictive mean on CIFAR-10
7. A real-world experiment on Wine dataset was added, where we tune the width of the DNN.
Approximate inference turns deep networks into Gaussian processes
Appendix E

Improving predictions of Bayesian neural networks via local linearization

Improving predictions of Bayesian neural networks via local linearization

Alexander Immer * 1 Maciej Korzepa * 2 Matthias Bauer 3

Abstract
In this paper we argue that in Bayesian deep learning, the frequently utilized generalized Gauss-Newton (GGN) approximation should be understood as a modification of the underlying probabilistic model and should be considered separately from further approximate inference techniques. Applying the GGN approximation turns a BNN into a locally linearized generalized linear model or, equivalently, a Gaussian process. Because we then use this linearized model for inference, we should also predict using this modified likelihood rather than the original BNN likelihood. This formulation extends previous results to general likelihoods and alleviates underfitting behaviour observed e.g. by Ritter et al. (2018). We demonstrate our approach on several UCI classification datasets as well as CIFAR10.

1. Introduction
Inference in Bayesian neural networks (BNNs) usually requires posterior approximations due to intractable integrals and high computational cost. Given such an approximate posterior, one can make predictions by combining it with the original Bayesian neural network likelihood.

Recently, Foong et al. (2019) showed empirically that predictions using a “linearized Laplace” predictive approximation can match or outperform other approximate inference approaches, such as mean field variational inference (MFVI) in the original BNN model (Blundell et al., 2015) and provide better “in-between” uncertainties. Here we explain that their approach relies on an implicit change in probabilistic model due to the generalized Gauss-Newton (GGN) approximation. The GGN is often jointly applied with approximate inference in BNNs using the Laplace (Ritter et al., 2018; Foresee & Hagan, 1997; Foong et al., 2019) or variational approximation (Khan et al., 2018; Zhang et al., 2018).

We argue that the GGN and approximate inference should really be considered separately: (1) the GGN locally linearizes the underlying probabilistic model in the parameters and gives rise to a generalized linear model (GLM); (2) approximate inference enables posterior inference in this linearized GLM. Because we have done inference in a modified probabilistic model (the GLM), we should also predict with this modified model. We call the resulting predictive the “GLM predictive” in contrast to the “BNN predictive” that uses the original BNN likelihood, see Fig. 1.

Our formulation generalizes previous results by Khan et al. (2019) and Foong et al. (2019) to non-Gaussian likelihoods and explains why the GLM predictive works well compared to the BNN predictive. The BNN predictive with Laplace approximation can show underfitting (Lawrence, 2001) especially severely when combined with the GGN (Ritter et al., 2018). Our experiments show that the GLM predictive alleviates these underfitting issues and consistently outperforms the BNN predictive; it is on par or better than the neural network MAP or MFVI. Further, the GLM in weight space can be viewed as an equivalent Gaussian process (GP) in function space, which enables different inference algorithms.

Figure 1. The generalized Gauss Newton approximation (GGN) turns a Bayesian neural network (BNN) into a generalized linear model (GLM) with same likelihood distribution, but network function $f(x, \theta)$ linearized around $\theta^*$. When using GGN, we should use the GLM predictive and not the BNN predictive. $q(\theta)$ is an approximate posterior and $\theta^*$ is found by MAP estimate, Eq. (1).

2. Methods
2.1. Background
We consider a supervised learning task with inputs $x_n \in \mathbb{R}^D$ and outputs $y_n \in \mathbb{R}^C$ (regression) or $y_n \in \{0, 1\}^C$ (classification), $D = \{x_n, y_n\}_{n=1}^N$, and a probabilistic model with likelihood parameters $\theta$, $p(D|\theta) = \prod_n p(y_n|f(x_n, \theta))$. The features $f(x, \theta)$ are mapped to the outputs using an inverse link function $g^{-1}$, $E[y] = g^{-1}(f(x, \theta))$. 

BNN predictive (Eq. (3))
$$p_{\text{BNN}}(y|x, D) = \frac{1}{Z} \int q(\theta) p(y|f(x, \theta)) p(\theta|D) d\theta$$

GLM predictive (Eq. (8))
$$p_{\text{GLM}}(y|x, D) = \frac{1}{Z} \int q(\theta) p(y|f(x, \theta)) p(\theta|D) d\theta$$

$$f_{\text{GLM}}(x, \theta) = f(x, \theta^*) + \nabla \phi f(x, \theta)|_{\theta^*} (\theta - \theta^*)$$

* Equal contribution 1 ETH Zürich, CH and MPI-IS, DE 2 DTU, Copenhagen, DK 3 DeepMind, London, UK. * Work partly done during an internship at RIKEN AIP, Japan. Correspondence to: Matthias Bauer <msbauer@google.com>.

Presented at the ICML 2020 Workshop on Uncertainty and Robustness in Deep Learning. Copyright 2020 by the author(s).
Improving predictions of Bayesian neural networks via local linearization

In Bayesian DL we impose a prior \( p(\theta) = \mathcal{N}(m_0, S_0) \) and aim to compute the posterior \( p(\theta|\mathcal{D}) \). This posterior inference requires computation of a high-dimensional integral, the model evidence or marginal likelihood \( p(y|x) = \int p(\mathcal{D}|\theta)p(\theta) \, d\theta \). Exact inference is usually not possible and approximation techniques are used: \( q(\theta) \approx p(\theta|\mathcal{D}) \).

Many practical approaches compute the maximum a posteriori (MAP) solution given by \( \theta_{\text{MAP}} = \arg\max_{\theta} q(\theta) \) and return a point estimate or a distribution \( q(\theta) \) around \( \theta_{\text{MAP}} \).

\[
\ell(\theta) = \sum_{n=1}^N \log p(y_n|f(x_n, \theta)) + \log p(\theta). \tag{1}
\]

Popular in recent years, mean-field variational inference (MFVI) approximates the posterior \( p(\theta|\mathcal{D}) \) by a factored variational distribution \( q(\theta) \) optimizing using an evidence lower bound (ELBO) to the marginal likelihood.

To obtain the BNN predictive distribution, we integrate the (approximate) posterior \( q(\theta) \) against the BNN likelihood:

\[
p_{\text{BNN}}(y|x, \mathcal{D}) = \mathbb{E}_{q(\theta)}[p(y|f(x, \theta))] \approx \frac{1}{M} \sum_{m=1}^M p(y|f(x, \theta_m)), \quad \theta_m \sim q(\theta), \tag{2}
\]

Curvature (Hessian) information can be used for optimization or approximate inference. We express the involved Jacobian \( J \in \mathbb{R}^{C \times P} \) and Hessian \( H \in \mathbb{R}^{C \times P \times P} \) of the feature extractor \( f, |J_f(x)|_{ij} = \frac{\partial f_i(x, \theta)}{\partial \theta_j} \) and \( |H_f(x)|_{ijkl} = \frac{\partial^2 f_i(x, \theta)}{\partial \theta_j \partial \theta_l} \).

\[
\begin{align*}
\nabla_{\theta} \log p(y|f(x, \theta)) &= J_f(x)^T r(y; f) \\
\nabla_{\theta}^2 \log p(y|f(x, \theta)) &= H_f(x)^T r(y; f) - J_f(x)^T \Lambda(y; f) J_f(x), \tag{4}
\end{align*}
\]

where \( r(y; f) = \nabla_f \log p(y|f) \) can be interpreted as a residual, and \( \Lambda(y; f) = -\nabla_{\theta}^T \log p(y|f) \).

### 2.2. Generalized Gauss-Newton turns Bayesian neural nets into generalized linear models

The network Hessian \( H_f(x) \) in Eq. (5) is infeasible to compute, such that many approaches employ the generalized Gauss-Newton (GGN) approximation, which drops this term (Martens, 2014). The GGN can be viewed as an approximation that becomes exact if the network is a perfect predictor, \( r(y; f(x, \theta)) = 0 \forall (x, y) \). However, this is neither desired (it indicates overfitting) nor realistic.

Alternatively, the GGN can be interpreted as a local linearization of the network function \( f(x, \theta); f^\theta_{\text{lin}}(x, \theta) = f(x, \theta^* + J_f(x)(\theta - \theta^*)) \) at a parameter setting \( \theta^* \) (Bottou et al., 2018). In Fig. A1), this linearization reduces the BNN to a Bayesian generalized linear model (GLM) with log joint distribution

\[
\sum_{n=1}^N \log p(y_n|f^\theta_{\text{lin}}(x_n, \theta)) + \log p(\theta), \tag{6}
\]

where \( f^\theta_{\text{lin}}(x, \theta) \) is linear in \( \theta \) but not in the input data \( x \). Applying the GGN approximation turns the underlying probabilistic model locally from a BNN into a GLM.

In practice, the linearization point \( \theta^* \) is given by a MAP estimate found by regular (S)GD of Eq. (1).

### 2.3. Approximate inference in the GLM

For general likelihoods, inference in \( \theta \) is still intractable, and we have to resort to approximate inference. Here we focus on the Laplace approximation and variational inference, both of which lead to a Gaussian posterior approximation \( q(\theta) \) to \( p(\theta|\mathcal{D}) \) in Fig. A1), and we consider full covariance and diagonal approximations. Both approximations are computed iteratively for general likelihoods, see e.g. Bishop (2006, Chapter 4), whereas for Gaussian likelihoods they can be evaluated in a single step.

When the GGN and Laplace approximation are applied jointly one assumes that (S)GD optimization of Eq. (1) finds \( \theta^* \approx \theta_{\text{MAP}} \) and constructs a posterior approximation with mode \( \theta^* \) (Ritter et al., 2018; Foong et al., 2019). We found that approximate inference in the GLM model at \( \theta^* \) was able to improve on it using several update steps; i.e. the mode of \( q(\theta) \) can be different from \( \theta^* \). The GLM objective is convex and therefore easier to optimize and guarantees convergence.

### 2.4. The GLM predictive distribution

To make predictions, we combine the approximate posterior with the likelihood. Because we have done inference in the GGN-linearized model, we should predict using this modified model and not the original BNN likelihood:

\[
p_{\text{GLM}}(y|x, \mathcal{D}) = \mathbb{E}_{q(\theta)}[p(y|f^\theta_{\text{lin}}(x, \theta))], \quad \theta_m \sim q(\theta), \tag{7}
\]

We stress that the GLM predictive in Eq. (8) typically uses the same approximate posterior as the BNN predictive, Eq. (3), but locally linearized features in the likelihood.

### 2.5. Gaussian process formulation of the GLM

A Bayesian GLM in weight space is equivalent to a (generalized) Gaussian process (GPP) in function space with a particular kernel (Fig. A1) (Rasmussen & Williams, 2006). The corresponding log joint is given by \( \sum_{n=1}^N \log p(y_n|x_n) + \log p(f) \), with GP prior \( p(f) \) mean and covariance function:

\[
\begin{align*}
\mathbf{m}(x) &= f^\theta_{\text{lin}}(x; \mathbf{m}_0) \\
\mathbf{k}(x, x') &= J_f(x) \mathbf{S}_0 J_f(x')^T,
\end{align*} \tag{9}
\]

where \( \mathbf{m}_0 \) and \( \mathbf{S}_0 \) are the prior parameters of \( p(\theta) \). Similar to the GLM, we can derive a GGP predictive and perform ap-
Improving predictions of Bayesian neural networks via local linearization

Figure 2. Mean and std dev of the predictive distribution on the banana dataset. Our proposed GLM predictive utilizes the GGN-linearized likelihood and the Laplace posterior (either diagonal or full covariance) for predictions and shows improved growing uncertainties away from the data compared to the MAP. It also alleviates the underfitting problem (Ritter et al., 2018) of the BNN predictive with Laplace approximation that uses the same posterior but the (mismatched) original BNN likelihood. Further results in App. B.2.

approximate inference, see e.g. Rasmussen & Williams (2006, Chapter 3.5). While both formulations are equivalent, they offer two different ways of reducing computational complexity. In particular, for a full posterior covariance approximation, the \(\mathcal{O}(P^3)\) operations on the Jacobians are replaced by a \(\mathcal{O}(N^3)\) kernel matrix inversion, which enables us to scale to large network architectures. DNN2GP (Khan et al., 2019) uses this same construction to relate approximate inference in DNNs to GPs with neural tangent kernels (Jacot et al., 2018) but always results in a Gaussian likelihood.

2.6. Generalizing previous results for regression

For Gaussian likelihoods, we recover the “linearized Laplace” model by Foong et al. (2019) as well as DNN2GP (Khan et al., 2019) when assuming that neural network training has converged and \(\theta^* = \theta_{\text{MAP}}\). In this case, the GLM predictive can be computed analytically instead of by sampling. We highlight that applying the GGN first was key for this derivation – there is nothing special about the subsequent Laplace approximation and any posterior approximation in the GLM can be used instead. These results explain why the GLM predictive outperforms the BNN predictive in Foong et al. (2019) and how underfitting in Ritter et al. (2018) can be addressed.

3. Experiments

The GLM predictive for \(\theta^* = \theta_{\text{MAP}}\) on regression is equivalent to the results in Foong et al. (2019) and Khan et al. (2019). Therefore, we focus on classification here.

First, we optimize the MAP objective Eq. (1) and obtain \(\theta^* \approx \theta_{\text{MAP}}\). We then linearize the network around \(\theta^*\) and employ either a Laplace or Gaussian variational approximation to the posterior of this GLM. As discussed in Section 2.3 we found that the mode \(\mu\) of the resulting approximate posterior \(q(\theta) = \mathcal{N}(\mu, \Sigma)\) could be somewhat different from \(\theta^*\) (in particular for larger networks). The BNN predictive uses a Laplace approximation (with GGN-approximated Hessian) around \(\theta^*\) as posterior (Foresee & Hagan, 1997; Ritter et al., 2018). We stress that these Laplace posteriors are very similar and this choice does not influence the qualitative results.

In all experiments, we use a diagonal prior, \(p(\theta) = \mathcal{N}(0, \delta^{-1} I_P)\), with precision \(\delta\) chosen through a validation set. We compare the GLM/GGP predictive that utilises the GGN-linearized likelihood, Eq. (8) (“GLM Laplace/VI”), to the commonly used BNN predictive, Eq. (3) (“BNN Laplace”), as well as to just using \(\theta^*\) as point estimate (“NN MAP”) and to MFVI maximizing the BNN ELBO with Bayes by backprop (“BBB”) (Blundell et al., 2015).

3.1. Illustrative example

We illustrate our method on the banana binary classification dataset with 2d input space, see Fig. 2. We use a 2-layer network with 50 \text{tanh} units (further experiments, comparisons and details in App. B.2).

The BNN predictive with Laplace posterior severely underfits like previously reported (Ritter et al., 2018); underfitting is worse for the full covariance posterior compared to the diagonal. Using the same Laplace posterior but the GLM predictive instead alleviates this problem. Thus, the underfitting is not due to the Laplace posterior but to using a mismatched predictive model. In contrast to using the MAP point-estimate, our GLM predictive also leads to growing
Improving predictions of Bayesian neural networks via local linearization

Dataset NN MAP BBB B N N Lapl G L M Lapl G L M ... our approach on UCI classification datasets as well as CIFAR10. In future work we aim to scale our approach further.

We now compare the different methods on a set of UCI classification tasks with 1 hidden layer (50 tanh). Approximate inference in the GLM is strictly superior to a Laplace approximation in the BNN and also provides a good alternative to Bayes-by-Backprop (Blundell et al., 2015). Results on accuracy and calibration can be found in Appendix B.3.

Approximate inference in the GLM is strictly superior to a Laplace approximation in the BNN and also provides a good alternative to Bayes-by-Backprop (Blundell et al., 2015). Results on accuracy and calibration can be found in Appendix B.3.

The results clearly show again that the predictive distribution has to match inference: when the GGP approximation is used, the GLM predictive (“GLM Lapl”) clearly outperforms the BNN predictive (“BNN Lapl”). Applying variational inference to the GLM (“GLM VI”) can further boost performance, and our methods perform on par or better than MFVI. Further, our method works out of the box for multiple hidden layers where MFVI is hard to tune.

The results clearly show again that the predictive distribution has to match inference: when the GGP approximation is used, the GLM predictive (“GLM Lapl”) clearly outperforms the BNN predictive (“BNN Lapl”). Applying variational inference to the GLM (“GLM VI”) can further boost performance, and our methods perform on par or better than MFVI. Further, our method works out of the box for multiple hidden layers where MFVI is hard to tune.

In Fig. 4 we highlight that the GLM predictive consistently outperforms the BNN predictive for any setting of the prior precision hyperparameter and that GLM VI can improve over the MAP estimate.

3.3. Larger scale results on CIFAR10

Finally, we evaluate our method on a larger problem, CIFAR10 with LeNet5 network. Due to the size of the model, we perform the inference in function space with the GGP instead of GLM. We split the original training set into 10 splits of 5000 examples each and evaluate performance of GGP VI and Laplace, see Tab. 2. For Laplace, we compare the GGP with mode at \( \theta^* \), i.e. \( f^* \), and with mode at an improved estimate \( f_{\text{GP mode}} \) obtained by several update steps through approximate inference in the linearized model.

While GGP Laplace evaluated with mode at \( f^* \) shows comparable performance to the NN MAP, several approximate update steps to \( f_{\text{GP mode}} \) result in a GGP model with considerably improved performance. This might be because it is hard to fully converge to \( \theta_{\text{MAP}} \) when training a large, highly non-linear NN model, but it is easier to find the mode of a convex posterior of a GLM. GGP VI with inducing points attains equivalent predictive performance to GGP Laplace.

4. Conclusion

In this paper we argued that in Bayesian deep learning, the frequently utilized generalized Gauss-Newton (GGN) approximation should be understood as a modification of the underlying probabilistic model and should be considered separately from further approximate inference techniques. Applying the GGN approximation turns a Bayesian neural network (BNN) into a locally linearized generalized linear model or, equivalently, a generalized Gaussian process. Because we then use this linearized model for inference, we should also predict using this modified likelihood rather than the original BNN likelihood. This formulation extends previous results by Khan et al. (2019) and Foong et al. (2019) to general likelihoods and alleviates underfitting behaviour observed e.g. by Ritter et al. (2018). We demonstrate our approach on UCI classification datasets as well as CIFAR10. In future work we aim to scale our approach further.
Improving predictions of Bayesian neural networks via local linearization

![Graph]

Figure 4. The GLM predictive (---) outperforms the BNN predictive (-----) for all settings of the model hyperparameter $\delta$.

Acknowledgements

We thank Emtiyaz Khan for the many fruitful discussions that lead to this work as well as Michalis Titsias for feedback on the manuscript. We are also thankful for the RAIDEN computing system and its support team at the RIKEN AIP.

References


Improving predictions of Bayesian neural networks via local linearization


A. Derivations

A.1. Impact of Gauss-Newton on the probabilistic model

The combination of a Laplace (Foresee & Hagan, 1997; Ritter et al., 2018) or a Gaussian variational (Khan et al., 2018; Zhang et al., 2018) approximation with the Generalized Gauss-Newton (GGN) approximation gives rise to recently successful approximate inference algorithms. In the aforementioned algorithms, the GGN facilitates an approximation to the Hessian of the log likelihood with respect to the neural network parameters $\theta$. The Hessian can be decomposed over the $N$ data points due to independence, such that we focus on a representative input-output pair $x \in \mathbb{R}^D, y \in \mathbb{R}^C$.

Recall from Section 2.1, that we define the Jacobian $J \in \mathbb{R}^{C \times P}$ and the Hessian $H \in \mathbb{R}^{C \times P \times P}$ of the neural network $f$ w.r.t. its parameters $\theta$ as

$$[J_\theta(x)]_{ci} = \frac{\partial f_c(x, \theta)}{\partial \theta_i} \quad \text{(A1)}$$

$$[H_\theta(x)]_{cij} = \frac{\partial^2 f_c(x, \theta)}{\partial \theta_i \partial \theta_j} \quad \text{(A2)}$$

Further, we define the gradient of the log likelihood w.r.t. the neural network features $f$ as $[r(y; f)]_c = \frac{\partial \log p(y|f)}{\partial f_c}$ and the Hessian of the log likelihood w.r.t. the neural network features $f$ as $[H_\theta(x)]_{cij} = -\frac{\partial^2 \log p(y|f)}{\partial f_i \partial f_j}$, respectively. Then, we can express the Jacobian and Hessian of the log likelihood w.r.t. the neural network parameters $\theta$ by using the chain rule as:

$$\nabla_\theta \log p(y|f(x, \theta)) = f_\theta(x) ^T r(y; f) \quad \text{(4)}$$

$$\nabla^2_{\theta\theta} \log p(y|f(x, \theta)) = f_\theta(x)^T H_\theta(x) + H_\theta(x)^T A(y; f) f_\theta(x), \quad \text{(5)}$$

The Gauss-Newton approximation to the Hessian of the log likelihood w.r.t. neural network parameter $\theta$ can then be written as (Martens, 2014)

$$\nabla^2_{\theta\theta} \log p(y|f(x, \theta)) \approx -f_\theta(x)^T A(y; f) f_\theta(x). \quad \text{(A3)}$$

This approximation is exact if all residuals are zero, i.e., $r(y_n; f(x_n; \theta)) = 0 \forall n \in \{1, \ldots, N\}$. This typically does not hold in practice and is also not desirable as it indicates overfitting.

Alternatively and equivalently, one can interpret the Gauss-Newton approximation as the true Hessian of a log likelihood of a locally linearized neural network (Martens & Grosse, 2015; Bottou et al., 2018). To show this, let $\theta^*$ be the parameter at which we want to evaluate the above approximation to the Hessian. The linearized neural network at $\theta^*$ is given by

$$f_{lin}^\theta(x, \theta) = f(x, \theta^*) + J_{\theta^*}(x)(\theta - \theta^*), \quad \text{(A4)}$$

and we obtain for its (un-approximated) Hessian of the log likelihood

$$\nabla^2_{\theta\theta} \log p(y|f_{lin}^\theta(x, \theta)) = -J_{\theta^*}(x)^T A(y; f_{lin}^\theta(x, \theta)) J_{\theta^*}(x), \quad \text{(A5)}$$

where the first term in Eq. (5) vanishes because the network is linear. Thus, we find that the GGN-Hessian approximation in Eq. (A3) and the un-approximated Hessian of the locally linearized model at $\theta = \theta^*$ in Eq. (A5) are equivalent; the Jacobians become $J_{\theta^*}(x)$ in both cases and $f_{lin}^\theta(x; \theta^*) = f(x; \theta^*)$.

This second view allows us to clearly understand the impact of the GGN on the underlying probabilistic model. The linearization reduces the BNN to a Bayesian generalized linear in the parameters model (GLM), see → in Fig. A1.
Improving predictions of Bayesian neural networks via local linearization

We now apply the linearization that leads to the \textit{GGN} approximation of the Hessian in our probabilistic neural network model. The joint of the original \textit{BNN} model is given by

\[
p_{\text{BNN}}(\theta, \mathcal{D}) = p(\theta) \prod_{n=1}^{N} p(y_n | f(x_i; \theta)). \tag{A6}
\]

Applying the local linearization at some parameter \( \theta^* \), we obtain the following modified probabilistic model:

\[
p_{\text{GLM}}(\theta, \mathcal{D}) = p(\theta) \prod_{n=1}^{N} p(y_n | f_{\text{lin}}(x_n, \theta)). \tag{A7}
\]

\section*{A.2. Recovering the Laplace-\textit{GGN}}

Suppose \( \theta^* \approx \arg \max_\theta \log p_{\text{BNN}}(\theta, \mathcal{D}) \), i.e., \( \theta^* \) is a MAP estimate of the \textit{BNN} model, Eq. (A6). To construct a Laplace approximation, Ritter et al. (2018) and Foresee & Hagan (1997) approximate the Hessian of the log joint distribution at \( \theta^* \). In their derivation, both make use of the \textit{GGN} as either a Hessian approximation or optimization algorithm and therefore do not identify the underlying formulation of the \textit{GLM}. They approximate the Hessian of the \textit{BNN} log joint as follows:

\[
\nabla^2_{\theta\theta} \log p_{\text{BNN}}(\theta, \mathcal{D}) |_{\theta = \theta^*} \approx \nabla^2_{\theta\theta} \log p(\theta) |_{\theta = \theta^*} + \sum_{n=1}^{N} \mathcal{J}_{\theta^*}(x) \Lambda(y_n; f(x_n; \theta^*)) \mathcal{J}_{\theta^*}(x) \tag{A8}
\]

\[
= \nabla^2_{\theta\theta} \log p_{\text{GLM}}(\theta, \mathcal{D}) |_{\theta = \theta^*}, \tag{A9}
\]

where the last equality emphasizes that the Hessian approximation used is in fact equivalent to that of our \textit{GLM} formulation. Therefore, when we care about the underlying probabilistic model we should make use of the \textit{GLM} as we have a proper Laplace approximation of it. In contrast, we do not know how good of an approximation we obtain in Eq. (A8) unless we know that all residuals are zero. In classification problems with \textit{GLM} likelihoods, the residuals can only become numerically zero due to the natural \textit{sigmoid} inverse link function mapping to \((0, 1)\) and the labels being in \(0, 1\). Critically, while in optimization we are interested in how the \textit{GGN} Hessian impacts convergence in second-order algorithms, in approximate inference we need to estimate the impact on the underlying model.

\section*{A.3. From \textit{GLM} to Generalized Gaussian process model (\textit{\iffalse} in Fig. A1\textit{\)}}

The identification of an underlying \textit{GLM} in approximate inference further allows specification of an equivalent Gaussian process (\textit{GGP}) model. The parametric prior distribution \( p(\theta) \) induces a functional distribution on \( f_{\text{lin}}(x, \theta) \):

\[
\mathbf{f} \sim f_{\text{lin}}^\theta(x, \theta_m) = f(x, \theta^*) + \mathcal{J}_{\theta^*}(x)(\theta_m - \theta^*) \quad \text{with} \quad \theta_m \sim p(\theta), \tag{A10}
\]

which induces a Gaussian process in \( \mathbf{f} \) because we assume a Gaussian prior \( p(\theta) \). Computing the mean and covariance under the Gaussian prior \( p(\theta) = \mathcal{N}(\mathbf{m}_0, \mathbf{S}_0) \) allows us to obtain the mean and covariance function of the Gaussian process prior

\[
\mathbf{m}(x) = f_{\text{lin}}^\theta(x; \mathbf{m}_0) \tag{9}
\]

which define the GP prior \( p(\mathbf{f}) = \mathcal{GP}(\mathbf{m}(x), \mathbf{k}(x, x')) \). This Gaussian process prior is equivalent to the Gaussian process prior of \textit{DNN GP} and similar to the neural tangent kernel (Khan et al., 2019; Jacot et al., 2018). The key difference is that we define a \textit{GGP} model as opposed to a Gaussian process regression model and can therefore handle non-Gaussian likelihoods in contrast to previous approaches. The \textit{GGP} model is then given by the joint distribution \( p(\mathbf{f}, \mathcal{D}) = p(\mathbf{f}) \prod_{n=1}^{N} p(y_n | f(x_n)). \) As for the \textit{GLM}, we can approximately infer this model using a Laplace approximation (Williams & Barber, 1998; Rasmussen & Williams, 2006) or variational approximation (Titsias, 2009; Hensman et al., 2012; 2015).

\section*{A.4. \textit{GLM} vs. \textit{GGP}}

A naive Gaussian posterior approximation to either the \textit{GLM} or the \textit{GGP} typically has computational costs of \( \mathcal{O}(P^3) \) or \( \mathcal{O}(N^3) \), respectively. In practice, however, we can employ diagonal or structured posterior covariance approximations to
Improving predictions of Bayesian neural networks via local linearization

reduce the cost from $O(P^3)$ down to $O(P)$ when working with the GLM. As the experiments suggest, using a diagonal variational approximation for the GLM is computationally cheaper and also performs decently well and is well-calibrated. Nonetheless, our experiments suggest that a full covariance leads to improvements over a diagonal covariance. In particular, the full covariance posterior approximation is well-calibrated compared to the diagonal version.

In contrast, scaling approximate inference in a GGP typically goes down other routes: For the Laplace approximation, we can reduce the computational cost to $O(NM^2)$ using, for example, the Nyström approximation. Similarly, a variational approximation (Titsias, 2009) can make use of $M \ll N$ inducing points to make inference scalable as $O(NM^2)$ and is typically superior to the Nyström approximation. Recent advances in scalable GP inference allow us to evaluate approximations on minibatches and scale as $O(NM^2)$, where $N$ denotes the minibatch size (Hensman et al., 2012).

Our GGP formulation enables us to do functional inference for parametric Bayesian neural networks with general likelihoods and therefore presents an alternative to other recent approaches like “neural processes” (Garnelo et al., 2018). Interestingly, the parametric GLM and functional GGP both offer different posterior approximations that enable scalability. This can open up new ways for highly flexible functional posterior approximations combining neural networks and Gaussian process inference.

B. Experimental details and additional results

B.1. Inference in GLM and GGP

GLM Laplace: We use the Adam optimizer with learning rate $10^{-3}$ and run for 1000 iterations using full-batch gradient descent in all experiments.

GLM VI: We use natural gradient variational inference (Amari, 1998; Khan et al., 2018; Zhang et al., 2018) to optimize the mean and covariance of the variational Gaussian posterior approximation. We take 250 full batch iterations in all experiments and use the learning rate $10^{-2}$ for diagonal covariance and $10^{-3}$ for full covariance.

GGP Laplace: Following (Williams & Barber, 1998; Rasmussen & Williams, 2006), we optimize for $f^{\text{train}}$ using Newton’s method, combining it with a line search similar to how it is implemented in the gpml toolbox (Rasmussen & Nickisch, 2010). For multiclass classification, we assume independence of prior functions across the outputs to reduce the computational complexity per step from $O(C^2N^3)$ to $O(CN^3)$ (Rasmussen & Williams, 2006).

GGP VI: We use VI inference with “whitening” and inducing points available in GPyTorch (Gardner et al., 2018). It assumes independence of both prior and posterior functions across the outputs to reduce the computational complexity per update to $O(CNM^2)$. We optimize the ELBO using Adam optimizer with initial learning rate $10^{-1}$. We do not learn the inducing points, fixing them to a random subset of the training data.

B.2. Illustrative example

We use a synthetic dataset known as ‘banana’ and separate 5% ($N = 265$) of it as training as 5% as validation dataset. For NN MAP, we tune the prior precision $\delta$ using the validation dataset on a uniformly-spaced grid of 10 values in range $[0.1, 2.0]$ for all architectures with at least two layers, otherwise we use a smaller range $[0.02, 0.4]$; for BBB (Blundell et al., 2015) we use 10 log-spaced values between $10^{-3}$ and 1. We optimize the models using full-batch gradient descent with the Adam optimizer with initial learning rate $10^{-2}$ (NN MAP) and $10^{-1}$ (BBB) for 3000 epochs, decaying the learning rate by a factor of 10 after 2100 and 2800 epochs. We run inference in GGP for 1000 iterations without inducing points as the training dataset is very small. After 500 epochs, we start decaying the learning rate by a factor of 0.99 each epoch. We show the predictive distribution for a $100 \times 100$ grid with input features $x_1$ and $x_2$ in $[-4, 4]$ range using 1000 Monte Carlo-samples to estimate the posterior predictive distribution for all methods.

In machine learning, uncertainty is often classified into aleatoric and epistemic. The aleatoric uncertainty is due to inherent noise in the data (e.g. two or more different classes overlapping in the input domain) and will always be there regardless of how many data points we sample. Therefore we might be able to quantify this uncertainty better by sampling more data, but we will not be able to reduce it. On the other hand, epistemic uncertainty is caused by lack of knowledge and can be minimized by sampling more data.

Therefore, the decomposition into aleatoric and epistemic uncertainty allows to establish to what extent a model is uncertain about its predictions due to inherent noise in the data and to what extent the uncertainty is due to the lack of data. Such
Improving predictions of Bayesian neural networks via local linearization

a distinction can be helpful in certain areas of machine learning such as active learning. To decompose the uncertainty of a Bernoulli variable, we follow (Kwon et al., 2020) which derives the following split of total variance into aleatoric and epistemic uncertainty:

\[
\text{Var}_{p_{\text{GLM}}(y^*|x^*,\mathcal{D})}(y^*) = \int \text{Var}_{p_{\text{GLM}}(y^*|x^*,\theta)}(y^*) q(\theta) d\theta + \int \left[\text{E}_{p_{\text{GLM}}(y^*|x^*,\theta)}(y^*) - \text{E}_{p_{\text{GLM}}(y^*|x^*,\mathcal{D})}(y^*)\right]^2 q(\theta) d\theta
\]

\(\text{(B11)}\)

Figure B2. Mean, std dev, aleatoric and epistemic uncertainty of the predictives on the banana dataset for all proposed methods for a network with 2 hidden layers with 50 units and tanh activation function. Laplace posterior approximation provides less confident predictions than VI-based one. Diagonal approximations (GLM Laplace and GLM VI) provide much worse uncertainty estimates than their full covariance counterparts. For diagonal Laplace approximation, the predictions get underconfident and miscalibrated, i.e. the uncertainty decreases away from data in many directions. On the other hand, for diagonal VI approximation, the predictions away from data get more confident when comparing to VI approximation with full covariance. Both diagonal approximations fail to capture aleatoric uncertainty. As expected, GGP predictives look very similar to GLM predictives.

In Fig. B2 we compare all proposed methods: GLM/GGP with Laplace/VI approximations. For GLM, we additionally show diagonal approximations. Because function space and weight space view are equivalent up to approximation, the GGP predictives look very similar to the corresponding GLM predictives. The Laplace approximation results in much less confident predictions than the VI approximation. This is expected as the Laplace approximation fits a Gaussian distribution locally at a mode of the true posterior, which can be highly skewed for classification likelihoods. As a result, when sampling from the Laplace approximation there are many samples with low posterior probability which leads to overestimated variance. On the other hand, VI fits a Gaussian approximation globally, avoiding sampling very improbable parameters / functions. Furthermore, the comparison with diagonal approximations shows that full covariance might be necessary to provide reasonable uncertainty estimates. The diagonal approximation fails to capture aleatoric uncertainty and results in greatly overestimated variance for Laplace approximation, even in the region of the input space densely populated by data. On the other hand, for VI, it seems to underestimate the uncertainty.
Improving predictions of Bayesian neural networks via local linearization

Figure B3. Mean and std dev of the predictives on the banana dataset for a network with 1 hidden layer with 50 units and tanh activation function. A significant decrease of parameters compared to a network with 2 layers results in a decrease in confidence of NN MAP as well as the corresponding GLM predictions. BNN Laplace predictive improves slightly compared to predictions in a model with 2 hidden layers, but the variance is still greatly overestimated.

Figure B4. Mean and std dev of the predictives on the banana dataset for a network with 3 hidden layers with 50 units each and tanh activation function (with the exception of BBB where number of hidden units per layer was limited to 5 due to failing to fit the data with more units per layer). Increasing the model capacity leads to more confident predictions in NN MAP and the corresponding GLM models (narrower decision boundary), yet the GLM model is still able to produce reasonable uncertainties growing away from data.

In Figs. B3 and B4, we show the NN MAP, BBB, BNN Laplace, and GLM Laplace predictives for a network with one and three layers, respectively. A lower number of parameters slightly improves the performance of BNN Laplace, but still the variance is severely overestimated. The GLM Laplace predictive has higher variance compared to the model with two layers (Fig. 2) similarly to NN MAP on which it was based. When the model is greatly overparameterized at 3 layers, BNN Laplace completely fails even for the diagonal approximation. For GLM Laplace, similarly as for NN MAP, the confidence of the predictions increases, however, the former still produces reasonable uncertainties growing away from data.

In Fig. B5, we show the predictives for relu activation function. GLM predictives show that both mean and variance are sliced by linear surfaces as can be expected from a piece-wise linear activation function. However, even though the predictive variance still decomposes reasonably into aleatoric and epistemic uncertainty, the overall quality and consistency of the predictive distribution in the context of this classification problem seems much lower than for tanh activation function. This suggests that the activation function may play an important role in ensuring the quality of predictive distributions for BNNs.
Finally, in Fig. B6 we show our GLM Laplace and VI predictives cope with quantifying the uncertainty in the regions of space without any data, but surrounded by data - so-called in-between uncertainty as shown for regression in (Foong et al., 2019). In order to produce a ‘data gap’, we remove all the data points with \(x_1 \in [-0.5, 0.5]\) from training and validation datasets. Our results suggest that full covariance is necessary to capture this kind of uncertainty. Both GLM Laplace and VI predictives show a considerable increase in epistemic uncertainty in \(x_1 \in [-0.5, 0.5]\) range. NN MAP and mean-field methods (both BBB and GLM with diagonal posterior approximation) fail to capture epistemic uncertainty - the width of the decision boundaries does not change in the in-between data gap.
B.3. UCI

We test the mentioned methods for approximate inference on 8 UCI classification data sets available from UCI Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets.php, see Tab. B1

We compare the neural network MAP, a BNN with mean-field variational inference trained with Bayes-by-Backprop (Blundell et al., 2015) (BBB) using the local reparameterization trick (Kingma et al., 2015), and a BNN with Laplace approximation using the GGN (Ritter et al., 2018; Foresee & Hagan, 1997) (Eq. 3) with our methods. We compare both a Laplace and Gaussian variational approximation of the GLM formulation (ref to main section) with both diagonal and full posterior covariance approximations. For all methods, we train until convergence (for BBB 5000 steps, for MAP 10000 steps) with the Adam optimizer (Kingma & Ba, 2014) using a learning rate of $10^{-3}$ for MAP and $10^{-3}$ for BBB. For inference in the GLM, we use 1000 iterations for the Laplace approximation and 250 for the variational approximations. For the Laplace approximation, we further optimize the GLM using Adam and a learning rate of $10^{-3}$. For the variational approximation the GLM, we use a natural gradient VI algorithm to update the mean and covariance of the posterior approximation (Amari, 1998; Khan et al., 2018; Zhang et al., 2018) with learning rates $10^{-3}$ for a full and $10^{-2}$ for a diagonal covariance. We split each dataset 10 times randomly into train/validation/test sets with ratios 70%/15%/15% and stratify by the labels to obtain proportional number of samples with particular classes. For each split, we train all above-mentioned methods with 10 different prior precisions $\delta$ on a log-spaced grid from $10^{-2}$ to $10^{2}$, except for the larger datasets satellite and digits where the grid is from $10^{-1}$ to $10^{2}$. In the resulting tables, we report the performance with the standard error on the test set after selecting the best hyperparameter $\delta$ on the validation set. We run above experiment for three network architectures: one hidden layer with $\text{tanh}$ activation and 50 hidden units and a two-layer network with $\text{tanh}$ activation and 50 units on each hidden layer. Further, we run the experiment for one hidden layer and $\text{relu}$ activation function.

Tables B2 and B3 complement Table 1 with results on accuracy and expected calibration error (ECE) which is a metric that
Improving predictions of Bayesian neural networks via local linearization

<table>
<thead>
<tr>
<th>Dataset</th>
<th>number of datapoints ( N )</th>
<th>input dimension ( D )</th>
<th>number of classes ( C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>australian credit approval</td>
<td>690</td>
<td>14</td>
<td>2</td>
</tr>
<tr>
<td>breast cancer Wisconsin</td>
<td>569</td>
<td>32</td>
<td>2</td>
</tr>
<tr>
<td>ionosphere</td>
<td>351</td>
<td>34</td>
<td>2</td>
</tr>
<tr>
<td>glass identification</td>
<td>214</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>vehicles silhouettes</td>
<td>846</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>waveform</td>
<td>1000</td>
<td>31</td>
<td>3</td>
</tr>
<tr>
<td>digits</td>
<td>1797</td>
<td>64</td>
<td>10</td>
</tr>
<tr>
<td>satellite</td>
<td>6435</td>
<td>36</td>
<td>6</td>
</tr>
</tbody>
</table>

Table B1. Overview of UCI classification datasets used

tells us the mismatch between classification accuracy and predictive certainty (Naeini et al., 2015). For the ECE, we use 10 bins. Further, Tables B4, B5, and B6 list the corresponding results for two hidden layers with 50 hidden units for each layer and \( \tanh \) activation. The two-layer network builds a stark contrast to the single layer network as it has more parameters than data points for all of the benchmarks except on the \textit{satellite} dataset. In Tables B7, B8, and B9 we report the corresponding results for a \textit{relu} instead of \( \tanh \) activation function on the hidden layer. The results are slightly different but overall in favor of the \textit{GLM VI} method, especially with respect to the calibration performance as shown in Table B9.

In the single layer experiment, the additional results on accuracy and ECE suggest that the MAP generally gives very accurate predictions and is on par with \textit{GLM VI} both in terms of accuracy and ECE. Despite the suboptimal calibration (according to ECE and test log likelihood), \textit{BNN} Laplace performs well in terms of accuracy on the \textit{australian} and \textit{breast cancer} datasets. The results for two hidden layers show similar trends but the performance of BBB suffers from higher non-linearity and more parameters. The \textit{GLM VI} method achieves the best nlls except on the \textit{digits} dataset and is also best calibrated according to the ECE metric. When we swap the \textit{tanh} activation in the hidden layer with a \textit{relu} activation, the results do not change significantly: \textit{GLM VI} and BBB perform best in the test log likelihood while the MAP and \textit{GLM VI} perform best in accuracy. Strikingly, \textit{GLM VI} performs by far best in terms of calibration. Overall, the proposed \textit{GLM VI} method dominates the benchmarks on the test log likelihood and ECE metrics while the MAP often provides the best accuracies.

The further results suggest that the proposed methods, especially variational inference in the proposed \textit{GLM}, perform on par or better than MAP and the BBB algorithm for BNNs. In particular for more parameters (see e.g. Tab. B4), the gap between BBB and the \textit{GLM} grows. While the test negative log likelihood and calibration results for \textit{BNN} Laplace are worse compared to the other methods, the accuracy can still be on par with other methods in some cases. Generally, the proposed \textit{GLM Laplace} yields strictly better results than \textit{BNN} Laplace and alleviates its underfitting problem.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>NN MAP</th>
<th>BNN</th>
<th>BBB Laplace</th>
<th>\textit{GLM} Laplace</th>
<th>\textit{GLM} ( \text{Laplace} ) d</th>
<th>\textit{GLM VI}</th>
<th>\textit{GLM VI} d</th>
</tr>
</thead>
<tbody>
<tr>
<td>australian</td>
<td>0.881 ± 0.010</td>
<td>0.883 ± 0.008</td>
<td>0.889 ± 0.008</td>
<td>0.882 ± 0.010</td>
<td>0.879 ± 0.013</td>
<td>0.880 ± 0.011</td>
<td>0.880 ± 0.009</td>
</tr>
<tr>
<td>breast cancer</td>
<td>0.967 ± 0.004</td>
<td>0.970 ± 0.003</td>
<td>0.973 ± 0.003</td>
<td>0.968 ± 0.003</td>
<td>0.972 ± 0.003</td>
<td>0.963 ± 0.003</td>
<td>0.971 ± 0.003</td>
</tr>
<tr>
<td>ionosphere</td>
<td>0.875 ± 0.010</td>
<td>0.909 ± 0.007</td>
<td>0.881 ± 0.008</td>
<td>0.872 ± 0.010</td>
<td>0.883 ± 0.009</td>
<td>0.898 ± 0.007</td>
<td>0.896 ± 0.007</td>
</tr>
<tr>
<td>glass</td>
<td>0.706 ± 0.011</td>
<td>0.672 ± 0.015</td>
<td>0.550 ± 0.014</td>
<td>0.691 ± 0.010</td>
<td>0.691 ± 0.013</td>
<td>0.700 ± 0.013</td>
<td>0.669 ± 0.013</td>
</tr>
<tr>
<td>vehicle</td>
<td>0.809 ± 0.005</td>
<td>0.802 ± 0.003</td>
<td>0.750 ± 0.003</td>
<td>0.815 ± 0.003</td>
<td>0.817 ± 0.004</td>
<td>0.827 ± 0.004</td>
<td>0.808 ± 0.004</td>
</tr>
<tr>
<td>waveform</td>
<td>0.862 ± 0.003</td>
<td>0.852 ± 0.003</td>
<td>0.858 ± 0.002</td>
<td>0.860 ± 0.003</td>
<td>0.845 ± 0.003</td>
<td>0.857 ± 0.003</td>
<td>0.859 ± 0.003</td>
</tr>
<tr>
<td>digits</td>
<td>0.977 ± 0.001</td>
<td>0.969 ± 0.001</td>
<td>0.954 ± 0.001</td>
<td>0.971 ± 0.001</td>
<td>0.963 ± 0.001</td>
<td>0.966 ± 0.001</td>
<td>0.967 ± 0.001</td>
</tr>
<tr>
<td>satellite</td>
<td>0.919 ± 0.001</td>
<td>0.900 ± 0.001</td>
<td>0.863 ± 0.001</td>
<td>0.918 ± 0.001</td>
<td>0.917 ± 0.001</td>
<td>0.918 ± 0.001</td>
<td>0.912 ± 0.001</td>
</tr>
</tbody>
</table>

Table B2. Test accuracy on UCI binary/multiclass classification tasks. 1 hidden layer, 50 units, \textit{tanh} activation. In comparison the the test log likelihoods (Tab. 1), \textit{BNN} Laplace can achieve good performance on three data sets here. The neural network MAP performs better than on test log likelihoods. The proposed \textit{GLM VI} approach achieves consistently good results.
Improving predictions of Bayesian neural networks via local linearization

**Table B3.** Test ECE on UCI binary /multiclass classification tasks. 1 hidden layer, 50 units, tanh activation. The GLM VI method provides overall the best calibrated predictive probabilities according to the ECE metric. The MAP excels on the digits dataset.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>NN MAP</th>
<th>BBB</th>
<th>B NN Lapl</th>
<th>GLM Lapl</th>
<th>GLM Lapl d</th>
<th>GLM VI</th>
<th>GLM VI d</th>
</tr>
</thead>
<tbody>
<tr>
<td>australian</td>
<td>0.063±0.005</td>
<td>0.061±0.003</td>
<td>0.144±0.007</td>
<td>0.061±0.006</td>
<td>0.082±0.006</td>
<td>0.055±0.005</td>
<td>0.058±0.003</td>
</tr>
<tr>
<td>cancer</td>
<td>0.035±0.003</td>
<td>0.032±0.003</td>
<td>0.085±0.004</td>
<td>0.034±0.003</td>
<td>0.048±0.003</td>
<td>0.026±0.002</td>
<td>0.027±0.002</td>
</tr>
<tr>
<td>ionosphere</td>
<td>0.080±0.006</td>
<td>0.083±0.002</td>
<td>0.274±0.006</td>
<td>0.081±0.004</td>
<td>0.132±0.007</td>
<td>0.077±0.004</td>
<td>0.067±0.005</td>
</tr>
<tr>
<td>glass</td>
<td>0.155±0.010</td>
<td>0.180±0.008</td>
<td>0.215±0.010</td>
<td>0.173±0.008</td>
<td>0.232±0.010</td>
<td>0.157±0.005</td>
<td>0.189±0.008</td>
</tr>
<tr>
<td>vehicle</td>
<td>0.078±0.005</td>
<td>0.060±0.002</td>
<td>0.255±0.003</td>
<td>0.088±0.003</td>
<td>0.181±0.005</td>
<td>0.065±0.003</td>
<td>0.070±0.003</td>
</tr>
<tr>
<td>waveform</td>
<td>0.044±0.002</td>
<td>0.064±0.002</td>
<td>0.160±0.002</td>
<td>0.055±0.001</td>
<td>0.078±0.003</td>
<td>0.052±0.002</td>
<td>0.049±0.002</td>
</tr>
<tr>
<td>digits</td>
<td>0.014±0.000</td>
<td>0.041±0.001</td>
<td>0.410±0.001</td>
<td>0.156±0.001</td>
<td>0.257±0.001</td>
<td>0.049±0.002</td>
<td>0.033±0.002</td>
</tr>
<tr>
<td>satellite</td>
<td>0.020±0.000</td>
<td>0.021±0.001</td>
<td>0.119±0.001</td>
<td>0.044±0.001</td>
<td>0.090±0.001</td>
<td>0.018±0.001</td>
<td>0.021±0.001</td>
</tr>
</tbody>
</table>

**Table B4.** Negative test log likelihood on UCI binary /multiclass classification tasks. 2 hidden layers, 50 units each, tanh activation. In comparison to a single hidden layer, BBB performs much worse in this benchmark. Further, MAP and GLM Laplace achieve slightly better performance compared to other approaches. Overall, test negative log likelihoods for a single layer are slightly better.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>NN MAP</th>
<th>BBB</th>
<th>B NN Lapl</th>
<th>GLM Lapl</th>
<th>GLM Lapl d</th>
<th>GLM VI</th>
<th>GLM VI d</th>
</tr>
</thead>
<tbody>
<tr>
<td>australian</td>
<td>0.31±0.01</td>
<td>0.34±0.01</td>
<td>0.42±0.00</td>
<td>0.32±0.02</td>
<td>0.33±0.01</td>
<td>0.32±0.02</td>
<td>0.31±0.01</td>
</tr>
<tr>
<td>cancer</td>
<td>0.11±0.02</td>
<td>0.11±0.01</td>
<td>0.19±0.00</td>
<td>0.10±0.01</td>
<td>0.11±0.01</td>
<td>0.11±0.01</td>
<td>0.12±0.02</td>
</tr>
<tr>
<td>ionosphere</td>
<td>0.35±0.02</td>
<td>0.41±0.01</td>
<td>0.50±0.00</td>
<td>0.29±0.01</td>
<td>0.35±0.01</td>
<td>0.35±0.01</td>
<td>0.32±0.03</td>
</tr>
<tr>
<td>glass</td>
<td>0.95±0.03</td>
<td>1.06±0.01</td>
<td>1.41±0.00</td>
<td>0.86±0.01</td>
<td>0.99±0.01</td>
<td>0.98±0.07</td>
<td>0.83±0.02</td>
</tr>
<tr>
<td>vehicle</td>
<td>0.42±0.007</td>
<td>0.504±0.006</td>
<td>0.885±0.002</td>
<td>0.42±0.005</td>
<td>0.618±0.003</td>
<td>0.402±0.007</td>
<td>0.432±0.005</td>
</tr>
<tr>
<td>waveform</td>
<td>0.33±0.004</td>
<td>0.393±0.003</td>
<td>0.516±0.002</td>
<td>0.33±0.004</td>
<td>0.388±0.003</td>
<td>0.33±0.004</td>
<td>0.364±0.008</td>
</tr>
<tr>
<td>digits</td>
<td>0.094±0.005</td>
<td>0.219±0.004</td>
<td>0.875±0.002</td>
<td>0.255±0.002</td>
<td>0.409±0.002</td>
<td>0.150±0.002</td>
<td>0.149±0.008</td>
</tr>
<tr>
<td>satellite</td>
<td>0.230±0.002</td>
<td>0.307±0.002</td>
<td>0.482±0.001</td>
<td>0.241±0.001</td>
<td>0.327±0.002</td>
<td>0.227±0.002</td>
<td>0.248±0.002</td>
</tr>
</tbody>
</table>

**Table B5.** Test accuracy on UCI binary /multiclass classification tasks. 2 hidden layers, 50 units each, tanh activation. In comparison to test nll, many models achieve similar accuracies. GLM VI and MAP achieve consistently high performances.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>NN MAP</th>
<th>BBB</th>
<th>B NN Lapl</th>
<th>GLM Lapl</th>
<th>GLM Lapl d</th>
<th>GLM VI</th>
<th>GLM VI d</th>
</tr>
</thead>
<tbody>
<tr>
<td>australian</td>
<td>0.884±0.010</td>
<td>0.885±0.008</td>
<td>0.887±0.009</td>
<td>0.883±0.009</td>
<td>0.885±0.009</td>
<td>0.888±0.008</td>
<td>0.886±0.008</td>
</tr>
<tr>
<td>cancer</td>
<td>0.681±0.004</td>
<td>0.696±0.003</td>
<td>0.972±0.003</td>
<td>0.969±0.003</td>
<td>0.969±0.003</td>
<td>0.971±0.003</td>
<td>0.971±0.003</td>
</tr>
<tr>
<td>ionosphere</td>
<td>0.878±0.006</td>
<td>0.876±0.006</td>
<td>0.866±0.009</td>
<td>0.879±0.009</td>
<td>0.878±0.008</td>
<td>0.891±0.007</td>
<td>0.892±0.005</td>
</tr>
<tr>
<td>glass</td>
<td>0.684±0.010</td>
<td>0.531±0.013</td>
<td>0.459±0.009</td>
<td>0.679±0.013</td>
<td>0.669±0.013</td>
<td>0.675±0.015</td>
<td>0.669±0.012</td>
</tr>
<tr>
<td>vehicle</td>
<td>0.827±0.005</td>
<td>0.717±0.003</td>
<td>0.712±0.005</td>
<td>0.828±0.005</td>
<td>0.814±0.006</td>
<td>0.824±0.005</td>
<td>0.820±0.005</td>
</tr>
<tr>
<td>waveform</td>
<td>0.855±0.003</td>
<td>0.861±0.003</td>
<td>0.858±0.003</td>
<td>0.854±0.003</td>
<td>0.852±0.004</td>
<td>0.853±0.003</td>
<td>0.849±0.003</td>
</tr>
<tr>
<td>digits</td>
<td>0.974±0.001</td>
<td>0.951±0.002</td>
<td>0.924±0.001</td>
<td>0.964±0.001</td>
<td>0.960±0.001</td>
<td>0.966±0.001</td>
<td>0.970±0.001</td>
</tr>
<tr>
<td>satellite</td>
<td>0.916±0.001</td>
<td>0.891±0.001</td>
<td>0.842±0.001</td>
<td>0.915±0.001</td>
<td>0.910±0.001</td>
<td>0.917±0.001</td>
<td>0.911±0.001</td>
</tr>
</tbody>
</table>

**Table B6.** Test ECE on UCI binary/multiclass classification tasks. 2 hidden layers, 50 units each, tanh activation. In line with the single layer results, the GLM VI methods are calibrated best.
Improving predictions of Bayesian neural networks via local linearization

Table B7. Negative test log likelihood on UCI binary/multiclass classification tasks. 1 hidden layers, 50 units, relu activation. In comparison to relu activation, BBB improves over MAP and is close in performance to GLM VI methods. The best performance of the single layer tanh networks are slightly better than for relu presented here.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>NN MAP</th>
<th>BBB</th>
<th>BNN Lap</th>
<th>GLM Lap</th>
<th>GLM Lap d</th>
<th>GLM VI</th>
<th>GLM VI d</th>
</tr>
</thead>
<tbody>
<tr>
<td>australian</td>
<td>0.32±0.01</td>
<td>0.32±0.01</td>
<td>0.38±0.01</td>
<td>0.32±0.01</td>
<td>0.33±0.01</td>
<td>0.32±0.01</td>
<td>0.31±0.01</td>
</tr>
<tr>
<td>cancer</td>
<td>0.13±0.03</td>
<td>0.17±0.05</td>
<td>0.14±0.01</td>
<td>0.10±0.01</td>
<td>0.12±0.01</td>
<td>0.16±0.05</td>
<td>0.17±0.05</td>
</tr>
<tr>
<td>ionosphere</td>
<td>0.28±0.02</td>
<td>0.23±0.02</td>
<td>0.43±0.01</td>
<td>0.28±0.01</td>
<td>0.34±0.01</td>
<td>0.25±0.02</td>
<td>0.35±0.08</td>
</tr>
<tr>
<td>glass</td>
<td>0.89±0.03</td>
<td>1.07±0.07</td>
<td>1.21±0.01</td>
<td>0.88±0.01</td>
<td>0.95±0.01</td>
<td>0.78±0.03</td>
<td>0.80±0.02</td>
</tr>
<tr>
<td>vehicle</td>
<td>0.387±0.008</td>
<td>0.367±0.006</td>
<td>0.759±0.003</td>
<td>0.460±0.004</td>
<td>0.524±0.004</td>
<td>0.386±0.007</td>
<td>0.417±0.006</td>
</tr>
<tr>
<td>waveform</td>
<td>0.363±0.004</td>
<td>0.354±0.005</td>
<td>0.438±0.002</td>
<td>0.368±0.004</td>
<td>0.379±0.003</td>
<td>0.377±0.006</td>
<td>0.356±0.005</td>
</tr>
<tr>
<td>digits</td>
<td>0.078±0.003</td>
<td>0.136±0.008</td>
<td>0.460±0.003</td>
<td>0.264±0.002</td>
<td>0.367±0.002</td>
<td>0.122±0.002</td>
<td>0.143±0.008</td>
</tr>
<tr>
<td>satellite</td>
<td>0.230±0.003</td>
<td>0.278±0.002</td>
<td>0.348±0.001</td>
<td>0.246±0.002</td>
<td>0.310±0.002</td>
<td>0.229±0.002</td>
<td>0.251±0.002</td>
</tr>
</tbody>
</table>

Table B8. Test accuracy on UCI binary/multiclass classification tasks. 1 hidden layer, 50 units, relu activation. The MAP provides consistently the best accuracies on this architecture followed by the GLM VI methods.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>NN MAP</th>
<th>BBB</th>
<th>BNN Lap</th>
<th>GLM Lap</th>
<th>GLM Lap d</th>
<th>GLM VI</th>
<th>GLM VI d</th>
</tr>
</thead>
<tbody>
<tr>
<td>australian</td>
<td>0.888±0.009</td>
<td>0.872±0.016</td>
<td>0.878±0.007</td>
<td>0.886±0.009</td>
<td>0.885±0.010</td>
<td>0.885±0.009</td>
<td>0.887±0.008</td>
</tr>
<tr>
<td>cancer</td>
<td>0.968±0.004</td>
<td>0.964±0.004</td>
<td>0.969±0.003</td>
<td>0.970±0.003</td>
<td>0.970±0.003</td>
<td>0.967±0.004</td>
<td>0.968±0.004</td>
</tr>
<tr>
<td>ionosphere</td>
<td>0.906±0.007</td>
<td>0.926±0.007</td>
<td>0.864±0.014</td>
<td>0.906±0.006</td>
<td>0.902±0.006</td>
<td>0.904±0.009</td>
<td>0.921±0.007</td>
</tr>
<tr>
<td>glass</td>
<td>0.672±0.014</td>
<td>0.644±0.014</td>
<td>0.562±0.011</td>
<td>0.653±0.014</td>
<td>0.659±0.013</td>
<td>0.678±0.014</td>
<td>0.684±0.015</td>
</tr>
<tr>
<td>vehicle</td>
<td>0.882±0.003</td>
<td>0.814±0.004</td>
<td>0.789±0.004</td>
<td>0.809±0.003</td>
<td>0.824±0.004</td>
<td>0.832±0.004</td>
<td>0.825±0.003</td>
</tr>
<tr>
<td>waveform</td>
<td>0.857±0.003</td>
<td>0.856±0.003</td>
<td>0.859±0.003</td>
<td>0.857±0.003</td>
<td>0.859±0.003</td>
<td>0.857±0.003</td>
<td>0.847±0.002</td>
</tr>
<tr>
<td>digits</td>
<td>0.981±0.001</td>
<td>0.973±0.001</td>
<td>0.952±0.001</td>
<td>0.971±0.001</td>
<td>0.957±0.001</td>
<td>0.909±0.001</td>
<td>0.976±0.001</td>
</tr>
<tr>
<td>satellite</td>
<td>0.916±0.001</td>
<td>0.899±0.001</td>
<td>0.892±0.001</td>
<td>0.913±0.001</td>
<td>0.908±0.002</td>
<td>0.912±0.001</td>
<td>0.908±0.001</td>
</tr>
</tbody>
</table>

Table B9. Test ECE on UCI binary/multiclass classification tasks. 1 hidden layer, 50 units, relu activation. In line with the other architectures, the proposed GLM VI method provides the best calibrated predictive probabilities.
B.4. CIFAR

We use LeNet5 with 62K parameters and train on CIFAR10 dataset splitting the original training set (50K examples) into 10 splits of 5K examples to reduce complexity of inference in GGP and evaluate standard errors. We always test on the full original test set (10K examples). We tune the prior precision for i-th split using (i + 1)-th split as validation dataset. We test 12 values for the prior precision uniformly spaced between 10 and 120. We train with Adam optimizer for 60 epochs. The initial learning rate is $10^{-3}$ and decayed by a factor of 10 after 35 and 50 epochs to ensure that we converge as close to $\theta_{\text{MAP}}$ as possible. We compare the performance of NN MAP (evaluated at $\theta^*$), GGP Laplace evaluated at both $\theta^*$ and inferred $\theta_{\text{MAP}}$ and GGP VI with 512, 1024 and 2048 inducing points. We train for 200 epochs and minibatch size 512, decaying the learning rate by a factor of 0.97 at each epoch. We evaluate the performance of GLM and GGP methods by drawing 50,000 Monte Carlo samples from the diagonal GP predictive posterior, passing them through softmax function and estimating the expectation. We note that sampling from the functional posterior is very cheap as the functional posterior for a single data point is a $C$-dimensional diagonal Gaussian distribution.
Appendix F

Bayesian meta-learning using functional inference in linearized neural networks

BAYESIAN META-LEARNING USING FUNCTIONAL INFERENCE IN LINEARIZED NEURAL NETWORKS *

Maciej Korzepa
Technical University of Denmark
Kongens Lyngby, Denmark
mjko@dtu.dk

ABSTRACT

Learning from few examples is a remarkable feature of human intelligence and is linked to the ability of adequately applying the accumulated prior knowledge to a new situation. Bayesian modelling offers a powerful theoretical framework to conceptualize learning algorithms that could mimic this skill, yet it is a major computational challenge to apply Bayesian modelling to complex models that offer enough expressive power. We propose a new meta-learning method that combines the flexibility of neural networks with the simplicity of Bayesian inference in Gaussian Processes. We achieve this by defining our meta-model to be a first-order Taylor expansion of a neural network for which imposing a simple parametric Gaussian prior facilitates efficient functional inference in the corresponding Gaussian process model, exploiting the minuscule size of datasets typical in meta-learning. We investigate the properties of the proposed method trained using two different objectives, marginal likelihood and posterior predictive likelihood, on a set of synthetic regression and binary classification tasks. Finally, we discuss potential directions of future work that would help make the method more versatile.

1 INTRODUCTION

Deep neural networks have for long time excelled at modelling complex functions and processing huge amounts of data. However, due to the tendency to overfit due to overparametrization as well as limited control over inductive biases, they have struggled with achieving good generalization at low data regimes. Humans, on the other hand, are great examples of efficient and generalizable learners from few observations. It is argued that this skill stems from accumulated prior knowledge and applying it adequately to a new problem (Lake et al., 2017). Mimicking this human skill by machines is considered to play an important role in the development of artificial general intelligence (Griffiths et al., 2019).

Meta-learning, or learning to learn, is a subfield of machine learning addressing this challenge. It focuses on building a kind of representation of prior knowledge based on a collection of training tasks (meta-training) and applying it to solve efficiently previously unseen tasks (meta-testing). In the context of neural networks, one of the methods that has gained particular popularity is Model-Agnostic Meta-Learning (MAML) Finn et al. (2017). MAML aims to learn initial parameters of a neural network such that they can be easily adapted to solve a new task by taking a small number of gradient steps on the task’s training loss. A theoretically more compelling alternative is offered by Bayesian modelling framework. Instead of learning a point-estimate initialization, one would learn a prior distribution over model parameters that, when combined with the model, gives high support to functions that solve the tasks from a given task distribution well. Then, instead of adapting to a new task by taking a number of gradient steps minimizing the task’s training loss, one would compute the posterior distribution of the parameters by updating the learned prior based on the task’s data.

Unfortunately, exact Bayesian inference in neural networks is in practice computationally infeasible. Typically, this problem is tackled by approximate inference methods, such as Laplace approximation

---

*Work in progress
(MacKay, 1992; Ritter et al., 2018) or variational inference (Graves, 2011; Blundell et al., 2015; Khan et al., 2018), that approximate the exact posterior with a multivariate Gaussian distribution. In meta-learning, instead of performing a single inference on a large dataset, we need to solve a huge number of problems on very small datasets but typically using models with comparable number of parameters as when training on a single large dataset. To make Bayesian inference feasible in such scenario, special inference schemes have been proposed. Ravi & Beatson (2018) amortize VI inference by estimating the posterior parameters using a small number of gradient descent steps, similarly to MAML, but using a lower bound to the marginal likelihood as the objective. On the other hand, Gordon et al. (2019) perform stochastic inference only on the last (linear) layer of the neural networks predicting posterior parameters using amortization technique known from VAEs (Kingma & Welling, 2014) and keep the nonlinear part as a deterministic feature extractor.

In this paper we propose a different approach and facilitate efficient Bayesian inference not by focusing on the inference procedure itself but rather on simplifying the underlying meta-model. Inspired by prior work on linearized neural networks (Khan et al., 2019; Immer et al., 2020), we define our meta-model to be a first-order Taylor series expansion of a neural network. Representing neural network as a linear model brings a variety of computational benefits. Particularly, in regression with Gaussian likelihood and prior, closed-form solutions are available for the key quantities in Bayesian inference such as posterior distributions and marginal likelihood. On the other hand, for many non-Gaussian likelihoods such as Bernoulli or Categorical used in classification, the relevant quantities can be computed in functional form, greatly benefiting from tiny task datasets present in meta-learning and avoiding the huge computational costs of inference in parametric space which is usually the main limiting factor in neural networks.

2 Method

We consider a meta-learning problem with a collection of \( T \) training tasks, \( \mathcal{D} = \{(\mathbf{X}_t, \mathbf{Y}_t)\}_{t=1}^{T} \), where each task dataset \( (\mathbf{X}_t, \mathbf{Y}_t) \) consists of \( N \) input examples \( \mathbf{x}_t \), and corresponding \( C \)-dimensional outputs \( \mathbf{y}_t \). Further, we define a probabilistic task model with likelihood \( p(\mathbf{Y}_t | \mathbf{X}_t, \theta_t, \nu) \) \( \overset{i.i.d.}{=} \prod_{i=1}^{N} p(\mathbf{y}_{ti} | f(\mathbf{x}_{ti}, \theta_t), \nu) \), where \( \theta_t \in \mathbb{R}^D \) denotes tasks-specific parameters of neural network \( f \) and \( \nu \) is the set of parameters shared across all tasks. The features \( f(\mathbf{X}_t, \theta) \) are mapped to the mean of the likelihood by an inverse link function (e.g. identify for regression with Gaussian likelihood, or sigmoid for Bernoulli likelihood in binary classification).

2.1 Hierarchical Bayesian Model as Empirical Bayes

In probabilistic Bayesian setting, it is natural to approach the meta-learning problem using a hierarchical Bayesian model. To that end, we assume that task specific parameters \( \theta_t \) are random variables sampled from prior \( p(\theta_t | \gamma) \) where \( \gamma \), together with shared likelihood parameters \( \nu \), are global meta-parameter drawn from hyperprior \( p(\gamma, \nu) \). As a training objective, it is natural to consider the marginal likelihood of the observed data (or more specifically, its logarithm):

\[
\log p(\mathcal{D}) = \log \int \prod_{t=1}^{T} \int \int p(\mathbf{Y}_t | \mathbf{X}_t, \theta_t, \nu) p(\theta_t | \gamma) d\theta_t \left[ p(\gamma, \nu) d\gamma d\nu \right].
\]

To keep the inference feasible, instead of treating global parameters \( \gamma \) and \( \nu \) as random variables, we use point estimates for them. This simplification can be justified by the fact that in meta-learning we typically have a very large collection of tasks for training (i.e. large \( T \)) that allows to pinpoint the values of these variables accurately enough (which is not the case for task-specific parameters whose
values are inferred from very small amounts of data. Using point estimates for global parameters, we can rewrite the objective in Eq. (1) as:

\[
\log p_{\mu, \gamma}(D) = \log \prod_{t=1}^{T} p_{\nu}(Y_t | X_t, \theta_t)p_{\gamma}(\theta_t)d\theta_t = \sum_{t=1}^{T} \log p_{\mu, \gamma}(Y_t | X_t).
\] (2)

This formulation allows to estimate \( \gamma \) and \( \nu \) using stochastic gradients and parallelize the computations across tasks within a batch. While such formulation has been used previously in meta-learning with probabilistic neural networks as meta-models (Grant et al., 2018; Ravi & Beatson, 2018), inspired by prior work on neural network linearization (Immer et al., 2020), our work focuses on using a first-order expansion of a neural network as the meta-model and explores practical implications associated with it.

2.2 META-LEARNING AS INFERENCE IN GENERALIZED LINEAR MODEL

Inspired by prior works on linearized neural networks (Khan et al., 2019; Immer et al., 2020), we define our meta-model to be a first-order Taylor series expansion of a neural network. We start by defining a prior distribution \( p_{\gamma}(\theta) = \mathcal{N}(\mu, \Sigma) \) where \( \mu \in \mathbb{R}^P \), \( \Sigma \in \mathbb{R}^{P \times P} \) are global meta-parameters that we will learn (i.e. \( \gamma = \{ \mu, \Sigma \} \)) and compute the first-order expansion of \( f(x, \theta) \) at prior mean \( \mu \):

\[
f_{\mu}^{\text{lin}}(x, \theta) = f(x, \mu) + J_{\mu}(x)(\theta - \mu), \] (3)

where \( J_{\mu}(x) \in \mathbb{R}^{C \times P} \) denotes the Jacobian of \( f(x, \theta) \) evaluated at prior mean, \([J_{\mu}(x)]_{ct} = \frac{\partial f(x, \theta)}{\partial \theta_t} \bigg|_{\theta = \mu}\). Note that we drop the task index \( t \) from task-specific quantities (such as \( X_t, Y_t \) and \( \theta_t \)) as due to the formulation of the objective in Eq. (2) it is sufficient that we consider inference only within a single task.

Equipped with a linear model combined with a Gaussian prior, we could directly proceed to Bayesian inference (exact for Gaussian likelihoods and approximate for non-Gaussian ones) using the well-known formulas for marginal likelihood or posterior distribution of linear models (Bishop, 2006). However, even if we keep the prior covariance \( \Sigma \) diagonal, the resulting posterior covariance \( \Sigma_N \) would not have diagonal structure and consequently the computation of relevant quantities would have \( O(P^3) \) complexity due to matrix inversions and determinant evaluations. Instead of resorting to posterior approximations assuming simpler covariance structure (such as diagonal (Blundell et al., 2015) or Kronecker-factored (KFAC) (Ritter et al., 2018; Zhang et al., 2018) used in Bayesian neural networks), we exploit the equivalence between parameter and function space formulations (Rasmussen & Williams, 2006) and convert the parametric prior and model in Eq. (3) into a Gaussian Process (i.e., functional) prior \( f_{\mu}^{\text{lin}} \sim \mathcal{GP}(m, \kappa) \), whose mean function \( m(x) \in \mathbb{R}^C \) and kernel \( \kappa(x, x') \in \mathbb{R}^{C \times C} \) can be simply computed as the first and second moment of the linear model output under the parametric prior:

\[
m(x) = \mathbb{E}_{\theta \sim \mathcal{N}(\mu, \Sigma)} [f_{\mu}^{\text{lin}}(x, \theta)] = f(x, \mu), \] (4)

\[
\kappa(x, x') = \mathbb{E}_{\theta \sim \mathcal{N}(\mu, \Sigma)} [(f_{\mu}^{\text{lin}}(x, \theta) - m(x))(f_{\mu}^{\text{lin}}(x', \theta) - m(x'))^T] = J_{\mu}(x) \Sigma J_{\mu}(x')^T. \] (5)

For a task with \( N \) training examples and \( C \) outputs, the resulting functional prior will have \( CN \times CN \) covariance leading to operations of \( O(C^3 N^3) \) complexity given a precomputed kernel. In meta-learning, typically both \( C \) and \( N \) are small enough to keep this complexity realistic. However, one can assume that processes for individual outputs are independent (as it is commonly assumed for multi-output Gaussian Processes (Rasmussen & Williams, 2006; Liu et al., 2020)) which brings the complexity down to \( O(CN^3) \). One should also consider the complexity of GP prior evaluation. The prior mean in Eq. (4) is computed using a single forward pass with the task’s data. The kernel matrix evaluation (Eq. (5)) complexity depends on the structure of the prior covariance of the parameters. To keep the complexity low, we assume diagonal structure leading to \( O(C^2 N^2 P) \) time (or...
with the output independence assumption). One could also use diagonal plus low-rank structure to allow for more expressivity with little extra cost.

We also note that computing the predictive posterior or marginal likelihood using the GP prior introduced above is equivalent to computing the same quantities in parametric space with full posterior covariance, independently of the number of parameters in the underlying neural network model.

2.2.1 Gaussian likelihoods

For a task with data \( \mathbf{X}, \mathbf{Y} \) and Gaussian likelihood with noise variance \( \sigma^2 \), the marginal likelihood of the GP defined in Eq. (4) and Eq. (5) has a closed-form solution:

\[
\log p_{\mu, \Sigma}^{\text{lin}}(\mathbf{Y}|\mathbf{X}) = -\frac{1}{2} (\mathbf{Y} - \mathbf{m})^T (\mathbf{K} + \sigma^2 I)^{-1} (\mathbf{Y} - \mathbf{m}) - \frac{1}{2} \log |\mathbf{K} + \sigma^2 I| - \frac{N}{2} \log 2\pi, \quad (6)
\]

where \( \mathbf{m} \in \mathbb{R}^{CN} \) and \( \mathbf{K} \in \mathbb{R}^{CN \times CN} \) are the GP prior mean and covariance matrix evaluated for task data \( \mathbf{X} \). To update the parametric prior mean \( \mu \), we use calculate the gradient of Eq. (6):

\[
\frac{\partial \log p_{\mu, \Sigma}^{\text{lin}}(\mathbf{Y}|\mathbf{X})}{\partial \mu} = \frac{\partial \mathbf{m}}{\partial \mu} = \mathcal{J}_\mu(\mathbf{X})^T \mathbf{\alpha}, \quad (7)
\]

where \( \mathbf{\alpha} = (\mathbf{K} + \sigma^2 I)^{-1} (\mathbf{Y} - \mathbf{m}) \). We note that we do not propagate gradients through \( \mathcal{J}_\mu(\mathbf{X}) \) that is part of kernel matrix \( \mathbf{K} \) assuming that \( f(\mathbf{x}, \mu) \) is well approximated by the corresponding linear model (in which case derivative wrt derivative i.e. Hessian is zero). A similar assumption was made by the authors of MAML which allowed to simplify the backpropagation during meta-training with negligible impact on the performance (Finn et al., 2017).

Similarly, we calculate the gradients of Eq. (6) wrt. \( \Sigma \):

\[
\frac{\partial \log p_{\mu, \Sigma}^{\text{lin}}(\mathbf{Y}|\mathbf{X})}{\partial \Sigma} = \frac{1}{2} \mathcal{J}_\mu(\mathbf{X})^T (\mathbf{\alpha} \mathbf{\alpha}^T - (\mathbf{K} + \sigma^2 I)^{-1}) \mathcal{J}_\mu(\mathbf{X}) \quad (8)
\]

In practice, we calculate gradients wrt reparametrizations of \( \Sigma \) to ensure positive semidefiniteness constraint on \( \Sigma \). Similarly, we update the global likelihood parameters (i.e., \( \nu = \{\sigma\} \)) optimizing the same marginal likelihood objective.

2.3 Non-conjugate likelihoods

When the likelihood is non-conjugate with the prior, the marginal likelihood of the model in Eq. (3) does not have a closed-form solution and we need to apply some approximation to be able to optimize it wrt the parameters of the prior distribution. Similarly as in regression, we use the GP formulation to keep the computational complexity low while allowing full parametric space covariance structure. Inference in non-conjugate GP models can be approached with a number of approximations such as Laplace, variational or expectation propagation (Nickisch & Rasmussen, 2008). We choose the Laplace approximation due to its computational advantage. To that end, we first find the mode of the unnormalized functional posterior distribution, \( \hat{f} = \arg \max f p(\mathbf{Y}|f)p(f) \in \mathbb{R}^{CN} \), and estimate quadratic curvature at it to get an approximate Gaussian posterior. For likelihoods such as categorical with softmax inverse link function or Bernoulli with sigmoid inverse link function, the true posterior is convex meaning that the mode \( \hat{f} \) is unique and can be found by using optimization methods such as Newton method or even simple first-order such as gradient ascent or quasi second-order methods such as (L)BFGS. Following Rasmussen & Williams (2006), we can write the approximate Laplace marginal likelihood as:

\[
\log p_{\mu, \Sigma}^{\text{lin}}(\mathbf{Y}|\mathbf{X}) \approx \log p(\mathbf{Y}|\hat{f}) - \frac{1}{2} (\hat{f} - \mathbf{m})^T (\mathbf{K}^{-1}(\hat{f} - \mathbf{m}) - \frac{1}{2} \log |\mathbf{I} + \mathbf{W}^\dagger \mathbf{K} \mathbf{W}^\dagger|, \quad (9)
\]

where \( \mathbf{W} = -\nabla_\hat{f} \log p(\mathbf{Y}|\hat{f})|_{\hat{f}=\hat{f}} \).

To update the prior distribution, we calculate the gradients of Eq. (9) wrt \( \mu \) and \( \Sigma \) analogically to the regression case. The complexity of a single update given \( \hat{f} \) is identical as in the case of the regression.
(O(C^3N^3), or O(CN^3) with the output independence assumption). The main bottleneck is however the computation of the mode ̂f which requires optimization (yet in relatively low-dimensional space as f ∈ R^{CN}). Depending on the available resources, the optimization can be truncated to a fixed number of steps or a tolerance of the change of the objective value, resulting in an approximation of the true mode ̂f.

2.4 META-ADAPTATION

Within the Bayesian framework, it is natural to consider meta-adaptation as posterior predictive inference. Given a test task with training data (X, Y) and test input x∗, and prior on task parameters, θt, p∗(θt), learned during meta-training, our goal is to infer posterior parameter distribution p(θt|X, Y) and marginalize over its uncertainty to estimate the probability distribution of the predicted outcomes y∗, ̂p_{lin}(y∗|x∗, x, y). Mathematically, we write:

\[
p_{\text{lin}}(y∗|x∗, x, y) = \int p_{\text{lin}}(y∗|x∗, θt)p(θt|x, y)dθt.
\]

We can equivalently perform marginalization in function space using the GP formulation. For Gaussian likelihoods, the GP posterior predictive distribution is also a Gaussian distribution, with mean m∗_{post} ∈ R^C and covariance K∗_{post} ∈ R^{C×C} given by:

\[
m∗_{\text{post}} = k∗_T(K + σ^2I)^{-1}(y − m) + m(x∗) \quad \text{and} \quad K∗_{\text{post}} = κ(x∗, x∗) − k∗_T(K + σ^2I)^{-1}k∗ + σ^2I,
\]

where k∗ ∈ R^{NC×C} is a matrix whose i-th C × C block stores the evaluation of the kernel κ between training example x∗ ∈ X and the test example x∗.

For non-Gaussian likelihoods, we need to resort to approximations. Typically, we first approximate the true posterior (in either parametric or functional space) with Gaussian posterior and use MC sampling to approximate the integral in Eq. (10). Here, following the approach taken for marginal likelihood approximation we use Laplace approximation to find the mode of the concave functional posterior, ̂f and evaluate the quadratic curvature at that mode as detailed in Rasmussen & Williams (2006). The approximate posterior predictive of latent prediction f∗, ̂q_{lin}(f∗|x∗, X, Y), is then a Gaussian distribution with mean ̂m∗_{post} ∈ R^C and covariance ̂K∗_{post} ∈ R^{C×C} given by:

\[
̂m∗_{\text{post}} = k∗_T̂(K − 1)(̂f − m) + m(x∗) \quad \text{and} \quad ̂K∗_{\text{post}} = κ(x∗, x∗) − k∗_T̂(K + W^{-1})^{-1}k∗.
\]

Similarly as with marginal likelihood, we can assume independence of prior processes for different outputs to reduce the complexity of matrix inversions from O(C^3N^3) to O(CN^3) (assumption of posterior processes independence is not necessary here). For the details on efficient and numerically stable implementation, we refer to (Rasmussen & Williams, 2006).

We note that the presented approach is not directly applicable to the case where the outputs can be permuted across tasks, e.g. when different tasks have different combinations of classes to classify. We will address this limitation and potential solutions in Section 4.

2.5 ALTERNATIVE OBJECTIVE - POSTERIOR PREDICTIVE LIKELIHOOD

While marginal likelihood is a natural objective in Bayesian inference, it might have some limitations. First of all, it might lead to priors that give high support to functions that explain the training data well but that are not optimal in terms of generalization. Secondly, in few-shot problems where the number of shots is very low, the learning signal provided by the marginal likelihood might be very noisy and weak, and result in inefficient meta-training as well as underconfident predictions during meta-adaptation. To address these issues, we can use posterior predictive likelihood (i.e. Eq. (10)) as the meta-training objective evaluated on a set of test examples coming from the same function as the corresponding training examples. Optimization of the posterior predictive likelihood has been previously addressed in the context of large-scale Gaussian Processes for regression (Jankowiak et al. (2020)) as well as probabilistic meta-learning using VI (Gordon et al. (2019)).
For regression with Gaussian likelihood, as the posterior predictive distribution is Gaussian, we can optimize the posterior predictive log likelihood of the test data using closed-form updates. For non-Gaussian likelihoods, we again need to resort to approximations. If we choose the Laplace approximation, we can approximate it by MC sampling of latent functions from the approximate posterior \( \mathcal{N}(\mathbf{m}_\text{post}, \mathbf{K}_\text{post}) \) and passing the samples through the likelihood. To calculate gradients \( \text{wrt } \mu \text{ and } \Sigma \), we need to backpropagate through the latent posterior samples which we can do using reparametrization trick (Kingma & Welling, 2014).

3 Experiments

We demonstrate the properties of the proposed method on a few synthetic problems. In all cases, we use a fully-connected neural network with 2 hidden layers with 40 units each and with \text{tanh} nonlinearity. We use a Gaussian prior with fully specified diagonal covariance that in combination with the linearized neural network defines the meta-model. We initialize the prior mean (i.e. neural network parameters) using the Pytorch’s default weight initialization method and the diagonal prior variances to 0.5. For regression, we use Gaussian likelihood with identity inverse link function and initialize homoscedastic, task-independent observation noise \( \sigma \) to 0.5. For binary classification we use Bernoulli likelihood with sigmoid inverse link function. We optimize the meta-models using Adam optimizer with 1-e3 learning rate for prior mean and 5e-3 for prior and likelihood (log) variances, using marginal likelihood objective (or posterior predictive likelihood when stated so) for 5000 steps with batch size of 16 tasks. As the final meta-model, we choose the one from the step with the lowest average value of the objective over the previous 50 steps.

3.1 Regression

We train the method using two different task distributions - sinusoidal and linear functions. The sine task distribution is defined by 1-d sine functions having the same period \( 2\pi \) but with different amplitude and phase shift drawn uniformly from ranges \([0.1, 2.0]\) and \([0, \pi]\) respectively, and the line task distribution by linear functions with slope and vertical offset sampled uniformly from ranges \([-0.3, 0.3]\) and \([-1, 1]\) respectively. To construct a single task, we first sample a function \( f \) from a given task distribution and then sample a task train dataset with \( N = 5 \) data points, \( \{(x_i, f(x_i) + \epsilon_i)\}_{i=1}^{N} \), with \( x_i \sim \mathcal{U}(-5, 5) \) and noise \( \epsilon_i \sim \mathcal{N}(0, 0.05^2) \). For each task, we similarly sample a test dataset with \( N_{\text{test}} = 5 \) data points that we use when comparing marginal likelihood and posterior predictive likelihood objectives.

First, we qualitatively inspect the properties of the method when trained on the marginal likelihood objective for both linear and sinusoid task distributions. In Fig. 4 we show the analytic functional form of the posterior mean and standard deviation over \( \text{objective} \) for both linear and sinusoid task distributions. In Fig. 2 we show the analytic functional form of the posterior mean and standard deviation over \( \text{objective} \) for both linear and sinusoid task distributions. In Fig. 1b we can see that this is indeed the case. For sine tasks, while the learned prior fails to consistently produce samples with constant amplitude, a stable period and varying amplitude across samples are captured reasonably well which results in very good generalization as can be seen in Fig. 1b. Importantly, inference in the proposed meta-model seems to capture the uncertainty well which could prove very useful in different applications such as active learning.

Next, we investigate the impact of the training objective on the generalization of the meta-model. To that end, we compare the meta-models trained using marginal likelihood evaluated on the tasks’ training datasets and posterior predictive likelihood evaluated on both training and test datasets. As the second objective uses more data, we also train a third meta-model using marginal likelihood objective evaluated on combined train and test datasets. We evaluate the performance of the models on a meta-testing set of 1000 tasks, each consisting of 10 training and 50 test examples. We calculate the mean per-test-example negative log likelihood using a different number of training shots \((3, 4, \ldots, 10)\). We report the mean and standard error of the calculated metric over 5 training runs with different random initializations. We note that we keep the meta-training and meta-testing data fixed over all runs and tested methods. We show the results in Fig. 2. Both objectives show
3.2 CLASSIFICATION

We further test the method on a synthetic 2-d binary classification task following the definition in Finn et al. (2018). Each task has a positive class within a circle of radius sampled uniformly from \([0, 1, 2.0]\) range and center sampled from \([1, 4]\) range for both input dimensions independently. The input space beyond this circle corresponds to the negative class. For each task, we sample a training set with \(N_C = 5\) data points per class, uniformly from the space corresponding to the given class, but restricting the negative examples to lie within the \([0, 5]\) range for both input dimensions. In the same way, for each task, we also sample a test data set with \(N_{\text{test}}^C\) examples per class that we use when comparing the two different objectives. To find the mode of the functional posterior that is needed for the Laplace approximation, we use the LBFGS optimizer available in Pytorch with the default line search functions using strong Wolfe conditions, termination tolerance of \(1e-3\) on the mean per-point objective value and maximum number of 20 steps. To accelerate the optimization of the MAP objective, following Matthews (2016) we apply whitening on the function values we
Figure 3: Functional (after sigmoid) prior mean (a), prior samples (b) and posterior predictive distributions (b) for a linearized neural network meta-model trained using marginal likelihood objective on binary classification tasks with 5 training data points per class. The dashed black lines denote the ground truth boundaries. The posterior predictive means are approximated using 1000 samples from the corresponding latent functional posterior passed through the inverse link.

To optimize for. To optimize the posterior predictive log likelihood, we use 5 samples from the latent functional posterior.

Figure 4: Functional (after sigmoid) prior mean (a), prior samples (b) and posterior predictive distributions (b) for a linearized neural network meta-model trained using posterior predictive likelihood objective. The prior is much broader than the one trained using marginal likelihood and gives more flexibility to fit data better.

We start by visually inspecting the meta-learned functional prior (after sigmoid). The prior mean (Fig. 3a) shows higher probability of the positive class towards the center of the input space which agrees with the task distribution the prior was trained on. The sampled functions (Fig. 3b) capture well the circular shape of the decision boundary as well as show some diversity in its position and size. However, the samples appear to be very underconfident which might be the result of using Laplace approximation in the meta-training (the posterior mode used as the mean of the approximate Gaussian posterior can be very far from the true mean of the posterior). The means of the posterior predictive distributions (Fig. 3c) reflect the underconfidence present in the prior and exhibit problems with concentrating the support for the positive class to smaller regions of the input space. While Laplace approximation might lead to inherently inaccurate results, we might potentially improve the
situation by directly targeting the posterior log likelihood\(^1\) (of the test data to prevent overfitting). In Fig. 4a and Fig. 4b, we show that the prior trained using the posterior predictive log likelihood objective appears to be much broader and leaves more space to the data to influence the posterior predictions (Fig. 4c).

Finally, we compare the two objectives quantitatively using similar setup to the regression experiment, but with a meta-testing dataset consisting of up to 3-10 training examples per class and 100 test examples, sampled uniformly from the input space. In Fig. 5 we report the mean and standard error of mean per-test-example negative log likelihood averaged over 5 runs with different prior mean initializations. For classification using the Laplace approximation, the posterior predictive likelihood objective shows to be clearly better than the marginal likelihood, consistently decreasing the negative log likelihood of the test data with increasing number of training data points. Interestingly, the model trained with marginal likelihood with \(N_C = 5\) examples per class fails to contract the posterior probability towards the which suggests that training the meta-model on marginal likelihood objective using the Laplace approximation tends results in an overly strong prior that makes posterior inference ineffective. Increasing the amount of per-task data during meta training (\(N_C = 10\)) improves this behaviour, yet the quality of the resulting prior is sensitive to initialization.

4 DISCUSSION

By means of linearization, the presented meta-learning method combines flexibility of neural networks with simplicity and theoretical advantages of functional Bayesian inference in Gaussian Processes. We showed that using simple diagonal Gaussian priors in combination with linear models arising from linearization of neural networks, we can learn meaningful priors on the function space that encourage good generalization from few data examples as well as help in quantifying the uncertainty effectively. It would be interesting to see how the performance of the proposed method compares to other probabilistic meta-learning approaches. We leave carrying out such a comparison as a direction for future work.

The main limitation of the proposed method might be the choice of Laplace approximation as a method to approximate the meta-training objective for non-conjugate likelihoods. Especially in low-data regimes that are common in meta-training, the approximation of marginal likelihood might lead to meta-models that do not generalize well. A potential solution is using the posterior predictive likelihood objective, however, more work is needed on how to effectively optimize for functional posterior’s mode to ensure better convergence of the main objective. An alternative approach is to

---

\(^1\)While the posterior predictive log likelihood objective helps to achieve better results, we observed that in classification, at some point the objective starts diverging. We discovered that as the prior improves, the functional MAP optimization becomes more difficult and the LBFGS optimizer fails to converge properly, outputting a solution that makes the objective approximation more and more inaccurate. Nonetheless, as the objective at its best value before diverging gave good results consistently, we chose to report them and leave a closer investigation of how the problem can be solved to future work.
use a different approach to approximation of the objective such as VI. While standard VI could prove too slow (even in low dimensional function space), some amortization techniques could be explored to improve scalability of the approach, e.g. using ideas from (Ravi & Beatson, 2018) or (Gordon et al., 2019), yet applied in function space.

Another limitation is that the linearization at the prior mean leads to the same functional prior for all tasks, which renders meta-learning ineffective if different tasks have different combinations of classes to classify (similarly methods like MAML or Reptile (Nichol et al., 2018) fail to learn in such settings if a single gradient step is used for task adaptation). A potential solution to this problem might be linearizing at some other point than the prior mean (e.g. at MAP estimate) or considering the model as bilinear (by linearizing only up to the layer before the last linear layer). Both these approaches would however increase the complexity of inference and we leave careful investigation of the problem and the possible solutions for future work.

Currently, in the experiments, we assumed diagonal Gaussian priors which turned out to be quite effective in inducing meaningful priors on the function space. However, an interesting direction for future work is investigating how more complex structured priors (such as diagonal + low-rank) would influence the performance and training characteristics. Likewise, we could explore whether the proposed method could be used to learn priors for neural networks modules (e.g. convolutional kernels) that promote generalization and/or faster convergence in some applications, e.g. on tasks related to image or audio processing.

REFERENCES


