Towards Efficient and Scalable Representation Learning

Hai Pham

CMU-LTI-23-003

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15123
www.lti.cs.cmu.edu

Thesis Committee:

Barnabás Póczos **(Co-Chair)**	Carnegie Mellon University
David P. Woodruff **(Co-Chair)**	Carnegie Mellon University
Lori Levin	Carnegie Mellon University
Zoltán Szabó	London School of Economics and Political Science

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies

Keywords: Representation Learning, Multimodal Learning, Sentiment Analysis, Handwriting Recognition, Document Intelligence, Multimodal Self-Attention, Multitask Machine Translation, Task-Aware Mixture of Experts, Sketching Algorithms, Implicit Matrix Trace Estimation

Abstract

Nowadays it becomes more and more challenging to tackle the quickly growing amounts of data to extract useful information for making informed decisions. Even with the recent advancements in deep learning, however, the question of how to make use of such enormous data for a diverse set of tasks in an efficient and scalable manner has yet to be resolved.

To undertake the two main aspects of representation learning from data, namely efficiency and scalability, this thesis presents techniques to deal with diverse tasks including sentiment analysis, handwriting recognition and document intelligence where data appear in different forms: multimodal data that includes text, audio, and videos, noisy scanned handwriting images, or long documents with differing layouts. Due to the availability and potential issues of their data and the distinct objectives of the associated tasks, there is no one-size-fits-all solution but a specific approach to each problem. In addition, in dealing with large-scale data, this thesis also presents some approximation techniques and analysis to estimate the essential components, learn effective representation and speed up the learning process, including matrix trace approximation with a parallel non-adaptive method, spectrum approximation in Gaussian Processes training, and task-based mixture-of-experts models for large-scale multitask neural machine translation models. Throughout those works, this thesis introduces novel approaches for tackling issues that are presented in the data and the tasks, learning efficient representation, and approximating models for practical scalability in the real world.

Acknowledgements

Thank you, LTI/SCS/CMU!
Thank you, America!!

It is quite an honor to have done my master's and doctoral journeys at Carnegie Mellon University. I should not have gone this far without the incessant care and support from so many people in my life, however.

Despite my uncountable limitations, both of my advisors took me in, educated me, supported me, and helped me improve every day. Their expertise is undoubtedly at the top world-class level. Yet their kindness to me is boundless and immeasurable, which could easily take pages to explain. In addition, I am always gratified for having all freedom to explore research and teaching.

I am inexplicably grateful and proud to have Prof. Barnabás Póczos as my first advisor. I am deeply inspired by his leading expertise and dedication in the field, and also his exceptional versatility in working with multiple topics, e.g. statistics, machine learning, deep learning, and physics, just to name a few. Not only having done his job in advising me, but he is also always available to support me emotionally at times when things are down. I will always treasure his unstopping belief and investment in me during this long journey, as well as his availability and hands in my need.

I am also immensely grateful and proud to call Prof. David Woodruff my co-advisor. His prominent popularity in the field is unquestionable. I have never ceased to be amazed at how much expertise he has, let alone his unparalleled dedication and working ethic. I wish I could have known his work and met him earlier to pick up more things before actually working under his guidance. That is not even to mention his default willingness to help, a rare kindness not seen everywhere daily.

I am extremely lucky to have Prof. Lori Levin on the committee. Probably it was too late to get in touch with her closely. But better late than never, when I realized how important linguistic theories are in doing NLP research, and equally importantly how it should be done properly. Even more so since I am a non-native English speaker. I will never forget her dedication to helping me with a series of sessions in preparation for this thesis. If I ever did my journey again, I would take her linguistic classes without hesitation.

I am also extraordinarily fortunate with Prof. Zoltán Szabó's acceptance to be on the committee. I am profoundly grateful for his overboard kindness in making time for me to review the presentation and materials regardless of his deadlines and the thorough prompt communication with care. Furthermore, it is a serious mistake

to forget insightful questions and discussions that elicit a whole new collection of approaches to dealing with my problems. I deeply regret not having more time to work with him on the interesting topics of kernels and information theory to learn more from him while following up on my current interested research directions.

All in all, it is quite an honor for me to have my committee members for my thesis. It is trivially redundant to extol their respective expertise. But as often as it is probably underrated in academia, their kindness to me personally is incalculable and thus evermore gratified. I don't think I would have a better committee if given any other chance.

One of the most rewarding of doing research at CMU is that you are surrounded by the best professors and students either from CMU or its peers, thus creating a tremendous opportunity of learning from them. I have been super fortunate to learn from and collaborate with many professors, students, and collaborators who have taken a great part in making this thesis and other of my research possible: Professors Andrej Risteski, Alessandro Rinaldo, Yoon Kim, Eunsu Kang and Louis-Philippe Morency; shining students Paul Liang, Thomas Manzini, Shuli Jiang, Han Nguyen, Ilqar Ramazanli, Yiwei Lyu, Ramon Sanabria, Harvey Zhang, Amrith Setlur, Saket Dingliwal, Tzu Hsiang Lin, Deepak Dilipkumar, Daniel Clothiaux; great collaborators Tam Vu, Kang Huang, Zhuo Li, Jae Lim, Collin McCormack, Scott Johnson, Quiyi Zhang, Sashank Reddi, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Young Yin Kim, Hany Hassan Awadalla, Yiren Wang, Subho Mukherjee, Muhammad El-Nokrashy.

I would not also have come to CMU without great help from many people in the first place, including Dr. Weikuan Yu, Dr. Rodrigo Sardinas, Dr. Wei-Shinn Ku, Dr, Saad Biaz. At CMU, it is always memorable for the quick chats with overly kind people in the building such as Catherine Copetas, or the cleaners such as Donny & his wife, JB, Jamie and Joe. From LTI in particular, I am super fortunate to get great favor from many people around. Prominently, Prof. Ravi Starzl kindly helped me substantially during my very first time being here and advised me beside Prof. Barnabás Póczos for the beginning of my master, and Prof. Michael Shamos accommodated me greatly in teaching. Last but not least, I do appreciate help and favor from particularly Mary Jo Bensasi, Kate Schaich, remarkably Prof. Bob Frederking and late revered Prof. Jamie Carbonell, and especially, things cannot be done without kind dedication, care, and help for students including me from Stacey Young.

I always remember as well the good time hanging around with friends around CMU during my time here who also kindly helped me when I needed it, including but not limited to Ramon Sanabria, Diego Penafiel & AnaMa, Carla & Christopher, Johnathan & Bingqing, Shruti Palaskar, Paul Liang, Yue Niu, Chun Kai Ling, Eva Spiliopoulou, Rajat Kulshreshtha, Maria Ryskina, Bhuwan Dhingra & Rolly, Sanket

Mehta, Chirag Nagpal, Ankit Shah, Chaitanya Ahuja, Salvador & Mariana, Dheeraj Rajagopal, Sreecharan Sankaranarayanan, Juncheng Billy Li, and many others from LTI/CMU Masters and Ph.D. programs.

And whenever I am stuck with something, music has played an important part. Susanna has come not only as a music teacher but also as a dear friend who always gives words of wisdom to help me through. I am fortunate to have Susanna and her family with Rob and Arielle besides me.

Furthermore, being originally from Vietnam, I feel fortunate to have the support and friendships with Vietnamese people across the US, who sometimes helped me with many things, such as Vu & Ha, Quan & Yen, Son Van, Han Nguyen, Hien Ngo, Huy Anh Nguyen, Tyler Vuong, Lan Nguyen, Minh Hoang, Hieu Pham, Hai Nguyen (Boeing),... I am also very lucky to receive help from many other Vietnamese such as Mr. Nam & Mrs. Nguyet Bún Bò or Mr. Giao the barber, and finally, my close friends in Vietnam who regularly check in on me including Thien Vu, Xuan Cuong Nguyen, Hai Thanh Nguyen, Diep Ngoc Nguyen and especially Lien Cao.

Special thanks to those people who have been going extra miles to help me enormously and that changed my life and continue to do so: Chuong & Trang, Emma & Hung, Thuong Truong, Mr. Nguyen H. Bach, Mr. Lam & Mrs. Ly, Mr. Linh & Mrs. Thu, Mr. Khoi & Mrs. Hoa, Prof. Nam Tran & Mrs. Nhung, and Mr. Tom (Tam) Vu. It is a gravely unforgiving mistake, however, to not also mention my closest friends Tan Le and Giang Doan who have been with me forever, especially when in need.

Assuredly, I would not have gone this far without the support from my family, with the first being my beloved parents, who have been always pushing and asking for the dates I defend and get married for at least a decade now, my sister who is always reticent but overly kind to me, and other relatives and close neighbors who are truly interested in me. I know they are always there for me no matter what. I am also unbelievably fortunate and blessed to receive unconditioned love and support from my girlfriend Allie and her family through any highs and lows.

I would love to reserve the very last dedicated profound gratitude to someone who is–undoubtedly–the special-only-one to me. He has mentored and corrected me with astonishingly unconditioned attention, kindness and care. He has influenced and helped me improve from intellect, life, culture, history, and work to–importantly–spirit, and in such a way has been playing an irreplaceable part in my life forever. It is definitely without any exaggeration to state that I would not have become this version of myself without his favor to me in the first place, with many layers of meaning. I usually call him by the respected name “Thay” and sometimes refer to him as “Thay Du”, and hope to keep seeing him and his super kind wife. It has been colossal honor and blessing to have met and known him, and his family!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Given a task in machine learning, e.g. regression or classification, typically the main approach is to learn a function $\tilde{y} = f(x)$ from input data x. In the context of supervised learning, there are equivalent labels y to build a loss for optimization of them against the representation just learned $\tilde{y}$. There has been no conclusive definition of representation that the models learn from data. The typical understanding is that representation is the result obtained by a learning model upon input data. As a result, representation can be $\tilde{y}$ or can also be any intermediate learned product between $\tilde{y}$ and x.

In an alternative setting which is more and more common nowadays where a huge amount of data are presented, the labels y are not–or in many cases, too prohibitively expensive to be–provided, the models have to apply approaches that are different from the fully-supervised setting above for the representation. In many cases, such representations are rich enough for generating fake data that look realistic (Goodfellow et al., 2014; Song et al., 2023; Song and Ermon, 2019).

In practice, data can appear in many forms such as numbers, text, images, audio, videos, or any combination of them. Whether there is supervision in the setting, the main problem of machine learning methods is to obtain the efficient representation–from such data–that serves to solve the task. Despite the optimistic prospect and many advancements in machine learning and artificial intelligence in general, however, there has been a lack of a systematic methodology on how to properly learn the efficient representation for task-oriented objectives, how to interpret such representation, and importantly, how to scale up the solutions to enterprise-scale levels. This dissertation helps answer part of those questions in the context of different applications such as multimodal sentiment analysis, handwriting recognition, document understanding and multitask machine translation.

1.1 Thesis Statement

Whether there is supervision or not in a given learning problem, at the heart of the approach is the representation learning. How to undertake this task properly is an open question, however. This thesis aims to address two main aspects of representation learning via multiple problems and data, which are efficiency and scalability. Each focus of the two aspects plays an essential role in machine learning solutions in the real world that require not only satisfactory representation to solve the task suitably but also a versatile model that can save resources and be implemented in a large-scale deployment. Those two aspects are not opposing each other, but in fact, can complement each other to make up a consistent and practical solution.

1.2 Thesis Contributions

At a high level, the contributions that this thesis has made include the following topics.

- Novel models in learning multimodal data in sentiment analysis problem by applying the classical translation techniques to cross-domain data.
- Implementing a robust real-world handwriting recognition system in the face of limited and exceedingly noisy data
- Theory of sample complexity efficiency in approximate models in popular machine learning problems of Gaussian Processes and Implicit Trace Estimation.
- Implementing a scalable transformer-based pre-train model with multimodal approximate self-attention that can deal efficaciously with long input with diverse layouts.
- Introducing new task-based techniques that effectively link the application level and the infrastructure level of Mixture-of-Experts in the transformer-based architectures, given the discordant nature in data of multitask learning problems.
- Throughout many works, it is shown that efficiency and scalability can be integral parts of a practical performant solution.

1.3 Thesis Layout

As stated, there are two main parts to this thesis which are the efficiency and scalability of representation learning. As a result, the layout is following those topics sequentially. Note that it does not necessarily means each chapter only covers only efficiency or only scalability. But rather there are several models that concurrently contribute to both topics.

- In Section 2.1, we present our work on representation learning in the context of multimodal sentiment analysis (Pham et al., 2019), in which input data has three types: text,

audio, and videos. Unlike other approaches, we cast our problem as a cross-domain translation problem where one modality is trained to translate into another modality, e.g. text to audio or audio to video. With three modalities, we simply apply a hierarchical approach: two phases of translation are undertaken, in which two modalities are involved in the first phase and the third modality is added in the next phase. The embedded representation of the translation model is the output that we used for sentiment analysis. Our modality translation model, namely MCTN, outperformed various state-of-the-art methods on different benchmark datasets. In addition, MCTN has a big advantage in that only one modality is required for inference or prediction, unlike any other methods. This part is mainly based on our AAAI publication (Pham et al., 2019).

- Section 2.2 deals with image data but requires textual output. The particular task is handwriting recognition, in the context where input data is limited. In addition, the task is even more challenging since the data has lots of random noise. We break down this problem into two sequential small problems that are object detection and text recognition. In object detection, unlike common approaches for color images with multiple objects, we cast our problem as a text spotting problem, in which the model is trained to detect text from the background and noises. Furthermore, the input is converted to grayscale images to simplify the task. For text recognition, we explore two methods that are word-based and character-based recognition. Both of them are based on convolutional neural networks and each of them has pros and cons in our data. We also compare our whole pipeline's performance to the contemporary state-of-the-art methods. Along with experimental results, we discuss in detail the rationale of choices that were made for the pipeline and its components, as well as the reasons and hypothesis for the results. This work is based on our ICFHR 2020 (Pham et al., 2020).
- Section 2.3 introduces a novel pre-train model where multimodal positional encoding is employed along with the important approximate self-attention with multimodal context information. We explore the pros and cons of the traditional textual-based attention with the novel distance-based one and also examine the possibilities of combining the best of both types in a single attention head module. The pre-trained model is nevertheless simple, which is based on the popular Mask Language Model (Devlin et al., 2019) and thus easy to train and deploy. Our experiments show that our new models outperform the state-of-the-art models in both criteria: performance by having higher scores in document and token classifications, and computation by having a much larger input limit of 4096 tokens instead of 512 on the identical hardware and infrastructure platform. This work is under submission to a natural language processing conference.
- Section 3.1 and Section 3.2 deal with different methods of approximating the model for big data input, which is typically the case in practice, where deep neural networks are employed. On one hand, Section 3.1 introduces a sparse approximation based on spectrum

in the context of Gaussian Processes, which are known to suffer from the heavy computation. On the other hand, Section 3.2 analyses the practicality of the non-adaptive Hutch++ method, which has the best tradeoff in terms of performance and running time based on various benchmarks including synthetic and real datasets. Those approximation methods offer another angle into big data approaches besides other traditional methods which deal with algorithms, neural network architectures, operating systems, or hardware. Those works are based on our recent NeurIPS publications (Hoang et al., 2020; Jiang et al., 2021) and are different from other chapters in that they emphasize more on theoretical contributions of sample complexity, which is the main concern in practical models.

- Section 3.3 proposes new approaches to representation learning in a large-scale deployment where Mixture-of-Experts (MoE) models are being used to boost the transformer-based models. The main contribution is to integrate the task-based information from the top level of the technology stack with the lower-level MoE layers. To enable that, it also designs a set of task adapters to follow up with task-based MoE routed data into proper adapters, which are learned to group similar task data and separate dissimilar data, in order to alleviate the interference problems that are prevalent in multitask learning. This work is in preparation for submission.

At the high level, those aforementioned sections sequentially address the two main topics of this thesis, namely efficiency (Chapter 2) and scalability (Chapter 3) of representation learning. Such allocation is, however, rather descriptive than disconnected. As those sections are unfolded in their main contents, it is achievable to take advantage of scalability techniques to help with task-oriented representation learning to have a more practical and scalable approach without sacrificing the capabilities. Consequently, those two topics help convey the argument that this thesis conveys that in practice, one should plan the two topics collectively to have the best of both in a single solution.

Chapter 2

Efficiency of Task-Oriented Representation Learning

Without knowledge about the task including its data, it is almost impossible to design a efficient model to solve the task. Likewise, since there are too many diverse tasks, as well as data types ranging from various fields, there is no one-size-fits-all approach. On the contrary, with such expert knowledge acquired, the representation learning from data can be efficiently learned and used to optimize towards the final target using machine learning optimization techniques.

The following sections will enumerate the representation learning techniques for different tasks ranging from sentiment analysis, handwriting recognition to rich document understanding. Data for those tasks are also available divergently including multimodal data with text, audio, and video, noisy scanned handwritten images, and multimodal long documents with diverse formats.

2.1 Learning Multimodal Representation for Sentiment Analysis

The first task addresses an important problem of sentiment analysis, which is an essential part of interactive platforms such as forums or social networks. The problem here is complex since it contains not only text but also videos and audio. The intra- and inter-interactions among those three modalities require an architecture that is capable of modeling such complexities in both efficient and swift manners. The model in the following sections possesses those two capabilities.

2.1.1 Problem and Movitation

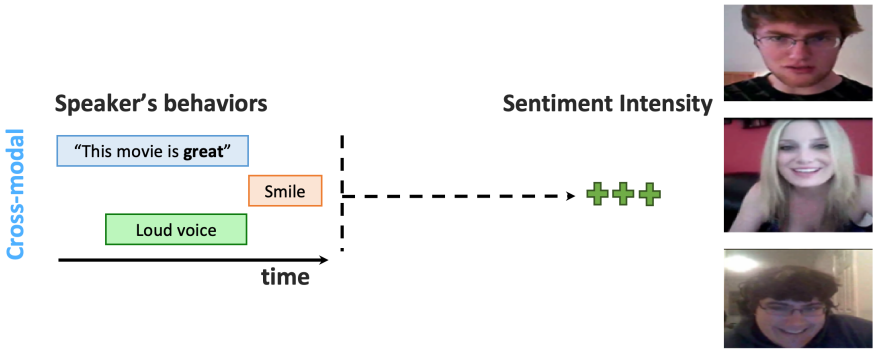


Figure 2.1: The representation learning of multimodal data is complicated due to not only intra-modal but also cross-modal interactions between different modalities.

Given a complicated set of multimodal data, the question is how to effectively learn the representation from it. This question is not totally clear yet, because it is not based on the context of usage. In more detail, how do we evaluate that representation after learning? As a result, when we want to learn a representation of any data, it has to be put in concrete evaluation metrics, such as in the form of a downstream task, such as multimodal sentiment analysis, an open research problem in machine learning, and natural language processing which involves identifying a speaker's opinion based on given data (Pang et al., 2002).

This problem is one of the cornerstones of unsupervised learning where we learn the representation directly from the data. Techniques used for representation learning vary depending on the specific downstream tasks, but they do all share the same characteristics that this is a very challenging problem. For example, text-only sentiment analysis through words, phrases, and their compositionality can be found to be insufficient for inferring sentiment content from spoken opinions (Morency et al., 2011), especially in the presence of rich nonverbal behaviors which can accompany language (Shaffer, 2018). In another example for the newly emerged task of document understanding (or in some contexts known as document intelligence), text-only models

show the disadvantages compared to the ones using text with more modalities, e.g. text+layout or text+layout+images (e.g. in (Xu et al., 2020a,b)). Finally, as an illustration, Figure 2.1 shows the inherent complication of dealing with learning with multimodal data where the complex interactions amongst them are usually not straightforward.

To address this problem, we propose a method of how to effectively learn the representation of data given different data sources and objectives. As a specific case, in the challenging task of multimodal sentiment analysis, where we have to make use of all modalities including text, audio, and videos, and in turn effectively produce the fused representation of such sources after being aligned, it will be shown that the simple model of Sequence-to-Sequence (Sutskever et al., 2014), which is the combination of 2 recurrent neural networks (RNNs) with different lengths originally used for machine translation because it can model the sequential relationship of languages and able to learn the alignment between two different sets of representations. We will be referring to this model as Seq2Seq.

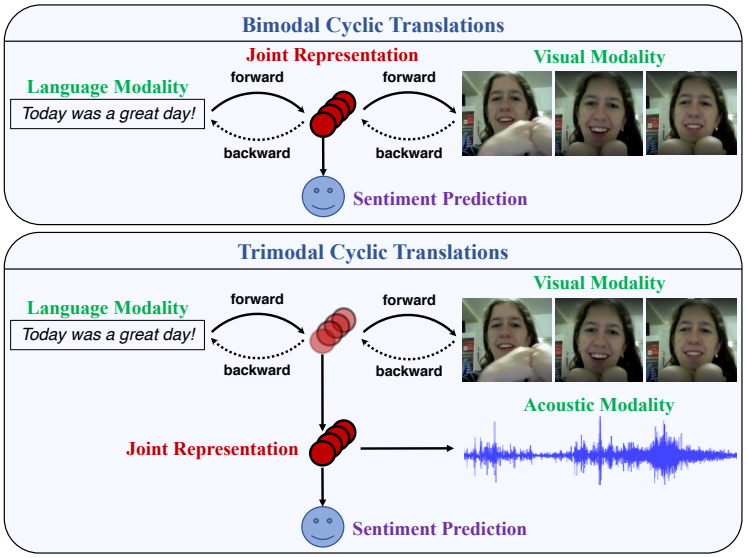


Figure 2.2: Learning robust joint representations via multimodal cyclic translations. Top: cyclic translations from a source modality (language) to a target modality (visual). Bottom: the representation learned between language and vision is further translated into the acoustic modality, forming the final joint representation. In both cases, the joint representation is then used for sentiment prediction.

We draw inspiration from the recent success of Seq2Seq models for unsupervised representation learning (Tu et al., 2016; **?**). We propose the Multimodal Cyclic Translation Network model (MCTN) to learn robust joint multimodal representations by translating between modalities. Figure 2.2 illustrates these translations between two or three modalities. Our method is based on the key insight that translation from a source modality S to a target modality T results in an intermediate representation that captures joint information between modalities S and T. MCTN extends

this insight using a cyclic translation loss involving both *forward translations* from source to target modalities, and *backward translations* from the predicted target back to the source modality. Together, we call these *multimodal cyclic translations* to ensure that the learned joint representations capture maximal information from both modalities. We also propose a hierarchical MCTN to learn joint representations between a source modality and multiple target modalities. MCTN is trainable end-to-end with a coupled translation-prediction loss which consists of (1) the cyclic translation loss, and (2) a prediction loss to ensure that the learned joint representations are task-specific (*i.e.* multimodal sentiment analysis). Another advantage of MCTN is that once trained with multimodal data, we *only* need data from the source modality at test time to infer the joint representation and label. As a result, MCTN is completely robust to test time perturbations or missing information on other modalities.

In more detail, the intermediate representation of this model offers interpretable information for the sentiment task if we cast the learning of multiple modalities as cross-domain translation tasks. Likewise, one of our contributions is that we use this model to translate not between texts/languages but between two different modalities, e.g. text and audio, of different domains. In addition, if we arrange two Seq2Seq models in a hierarchical manner, where we have two phases of cross-domain translation, we can achieve state-of-the-art (SoTA) results on multiple multimodal datasets. One key advantage of this simple technique is the simplicity and ease of prediction, unlike all other methods, in which we only need only 1 input modality, e.g. either text or audio.

2.1.2 Related Work

Early work on sentiment analysis focused primarily on written text (Pang and Lee, 2008; Pang et al., 2002; Socher et al., 2013). Recently, multimodal sentiment analysis has gained more research interest (Baltrusaitis et al., 2017). Probably the most challenging task in multimodal sentiment analysis is learning a joint representation of multiple modalities. Earlier work used fusion approaches such as concatenation of input features (Lazaridou et al., 2015; Ngiam et al., 2011). Several neural network models have also been proposed to learn joint multimodal representations. (Liang et al., 2018) presented a multistage approach to learn hierarchical multimodal representations. The Tensor Fusion Network (Zadeh et al., 2017) and its approximate low-rank model (Liu et al., 2018) presented methods based on Cartesian-products to model unimodal, bimodal and trimodal interactions. The Gated Multimodal Embedding model (Chen et al., 2017) learns an on-off switch to filter noisy or contradictory modalities. Other models have been proposed using attention (Cheng et al., 2017) and memory mechanisms (Zadeh et al., 2018) to learn multimodal representations.

In addition to purely supervised approaches, generative methods based on Generative Adversarial Networks, or GANs (Goodfellow et al., 2014) have attracted significant interest in learning joint distributions between two or more modalities (Donahue et al., 2016; Li et al., 2017). An-

other method for multimodal data is to develop conditional generative models (Kingma et al., 2014; Pandey and Dukkipati, 2017) and learn to translate one modality to another. Generative-discriminative objectives have been used to learn either joint (Kiros et al., 2014; Pham et al., 2018) or factorized (Tsai et al., 2018) representations. Our work takes into account the sequential dependency of modality translations and explores the effect of a cyclic translation loss on modality translations.

Finally, there has been some progress in accounting for noisy or missing modalities at test time. One general approach is to infer the missing modalities by modeling the probabilistic relationships among different modalities. Srivastava and Salakhutdinov (2014) proposed using Deep Boltzmann Machines to jointly model the probability distribution over multimodal data. Sampling from the conditional distributions over each modality allows for test-time inference in the presence of missing modalities. Sohn et al. (2014) trained Restricted Boltzmann Machines to minimize the variation of information between modality-specific latent variables. Recently, neural models such as cascaded residual autoencoders (Tran et al., 2017), deep adversarial learning (Cai et al., 2018), or multiple kernel learning (Mario Christoudias et al., 2010) have also been proposed for these tasks. It was also found that training with modalities dropped at random can improve the robustness of joint representations (Ngiam et al., 2011). These methods approximately infer the missing modalities before prediction (Collell et al., 2017; Hill et al., 2014), leading to possible error compounding, because the final results have to go through the prediction, which adds (and intensify) the current errors of the previous inference, which is sometimes uncontrollable. On the other hand, our method, described in Section 2.1.3 remains fully robust to missing or perturbed modalities during testing.

2.1.3 Proposed Model: Multimodal Cyclic Transalation Network (MCTN)

2.1.3.1 Problem Formulation

A multimodal dataset consists of N labeled video segments defined as $\mathbf{X} = (\mathbf{X}^l, \mathbf{X}^v, \mathbf{X}^a)$ for the language, visual, and acoustic modalities respectively. The dataset is indexed by N such that $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_N)$ where $\mathbf{X}_i = (\mathbf{X}_i^l, \mathbf{X}_i^v, \mathbf{X}_i^a)$, $1 \leq i \leq N$. The corresponding labels for these N segments are denoted as $\mathbf{y} = (y_1, y_2, ..., y_N)$, $y_i \in \mathbb{R}$. Following prior work, the multimodal data is synchronized by aligning the input based on the boundaries of each word and zero-padding each example to obtain time-series data of the same length (Liang et al., 2018). The ith sample is given by $\mathbf{X}_i^l = (\mathbf{w}_i{}^{(1)}, \mathbf{w}_i{}^{(2)}, ..., \mathbf{w}_i{}^{(L)})$ where $\mathbf{w}_i{}^{(\ell)}$ stands for the ℓth word and L is the length of each example (after padding). To accompany the language features, we also have a sequence of visual features $\mathbf{X}_i^v = (\mathbf{v}_i{}^{(1)}, \mathbf{v}_i{}^{(2)}, ..., \mathbf{v}_i{}^{(L)})$ and acoustic features $\mathbf{X}_i^a = (\mathbf{a}_i{}^{(1)}, \mathbf{a}_i{}^{(2)}, ..., \mathbf{a}_i{}^{(L)})$.

2.1.3.2 Learning Joint Representation

Learning a joint representation between two modalities $\mathbf{X}^S$ and $\mathbf{X}^T$ is defined by a parametrized function f_θ that returns an embedding $\mathcal{E}_{ST} = f_\theta(\mathbf{X}^S, \mathbf{X}^T)$, where S stands for source and T for target. From there, another function g_w is learned that predicts the label given this joint representation: $\hat{\mathbf{y}} = g_w(\mathcal{E}_{ST})$.

Most work follow this framework during both training and testing (Liang et al., 2018; Liu et al., 2018; Tsai et al., 2018; Zadeh et al., 2018). During training, the parameters θ and w are learned by empirical risk minimization over paired multimodal data and labels in the training set $(\mathbf{X}^S_{tr}, \mathbf{X}^T_{tr}, \mathbf{y}_{tr})$:

$$\mathcal{E}_{ST} = f_\theta(\mathbf{X}^S_{tr}, \mathbf{X}^T_{tr}), \tag{2.1}$$

$$\hat{\mathbf{y}}_{tr} = g_w(\mathcal{E}_{ST}), \tag{2.2}$$

$$\theta^*, w^* = \arg\min_{\theta,w} \mathbb{E}\left[\ell_{\mathbf{y}}(\hat{\mathbf{y}}_{tr}, \mathbf{y}_{tr})\right]. \tag{2.3}$$

for a suitable choice of loss function $\ell_{\mathbf{y}}$ over the labels (tr denotes training set).

During testing, paired multimodal data in the test set $(\mathbf{X}^S_{te}, \mathbf{X}^T_{te})$ are used to infer the label (te denotes test set):

$$\mathcal{E}_{ST} = f_{\theta^*}(\mathbf{X}^S_{te}, \mathbf{X}^T_{te}), \tag{2.4}$$

$$\hat{\mathbf{y}}_{te} = g_{w^*}(\mathcal{E}_{ST}). \tag{2.5}$$

2.1.3.3 Multimodal Cyclic Translation Network

Multimodal Cyclic Translation Network (MCTN) is a neural model that learns robust joint representations by modality translations. Figure 2.3 shows a detailed description of MCTN for two modalities. Our method is based on the key insight that translation from a source modality $\mathbf{X}^S$ to a target modality $\mathbf{X}^T$ results in an intermediate representation that captures joint information between modalities $\mathbf{X}^S$ and $\mathbf{X}^T$, but using only the source modality $\mathbf{X}^S$ as input during test time.

To ensure that our model learns joint representations that retain maximal information from all modalities, we use a cycle consistency loss (Zhu et al., 2017) during modality translation. This method can also be seen as a variant of back-translation which has been recently applied to style transfer (Prabhumoye et al., 2018; Zhu et al., 2017) and unsupervised machine translation (Lample et al., 2018). We use back-translation in a multimodal environment where we encourage our translation model to learn informative joint representations but with only the source modality as input. The cycle consistency loss for modality translation starts by decomposing function f_θ into two parts: an encoder f_{θ_e} and a decoder f_{θ_d}. The encoder takes in $\mathbf{X}^S$ as input and returns a joint embedding $\mathcal{E}_{S\to T}$:

$$\mathcal{E}_{S\to T} = f_{\theta_e}(\mathbf{X}^S), \tag{2.6}$$

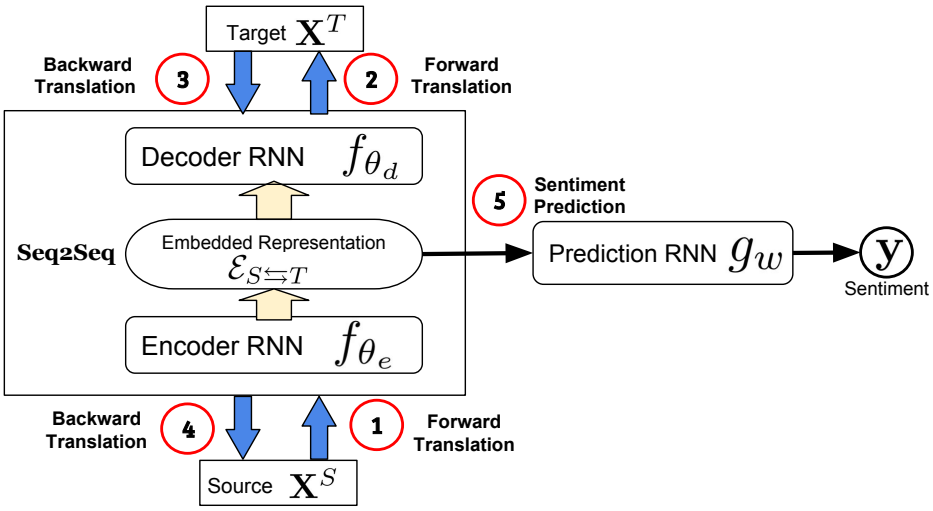


Figure 2.3: MCTN architecture for two modalities: the source modality $\mathbf{X}^S$ and the target modality $\mathbf{X}^T$. The joint representation $\mathcal{E}_{S \leftrightarrows T}$ is obtained via a cyclic translation between $\mathbf{X}^S$ and $\mathbf{X}^T$. Next, the joint representation $\mathcal{E}_{S \leftrightarrows T}$ is used for sentiment prediction. The model is trained end-to-end with a coupled translation-prediction objective. At test time, only the source modality $\mathbf{X}^S$ is required.

which the decoder then transforms into target modality $\mathbf{X}^T$:

$$\hat{\mathbf{X}}^T = f_{\theta_d}(\mathcal{E}_{S \to T}), \tag{2.7}$$

following which the decoded modality T is translated back into modality S:

$$\mathcal{E}_{T \to S} = f_{\theta_e}(\hat{\mathbf{X}}^T), \ \hat{\mathbf{X}}^S = f_{\theta_d}(\mathcal{E}_{T \to S}). \tag{2.8}$$

The joint representation is learned by using a Seq2Seq model with attention (Bahdanau et al., 2014) that translates source modality $\mathbf{X}^S$ to a target modality $\mathbf{X}^T$. While Seq2Seq model has been predominantly used for machine translation, we extend its usage to the realm of multimodal machine learning.

The hidden state output of each time step is based on the previous hidden state along with the input sequence and is constructed using a recurrent network. In more detail, for a single source input sample $\mathbf{X}_i^S$, $i \in [1, 2 \dots N]$, the recurring encoding happens at every single timestep (each one is corresponding to a word) in the sequential order, in which the previous hidden state (denoted as $\mathbf{h}$) is used as part of the encoding input for the very next timestep:

$$\mathbf{h}_i^{(\ell)} = \texttt{RNN}(\mathbf{h}_i^{(\ell-1)}, \mathbf{X}_i^{\mathbf{S},(\ell)}) \quad \forall \ell \in [1, 2, \dots, L], \tag{2.9}$$

where $\mathbf{X}_i^{\mathbf{S},(\ell)}$ is the encoded representation of $\mathbf{X}_i^S$ at timestep ℓ. The final encoder's output for that particular sample is the concatenation of all hidden states of the encoding RNN,

$$\mathcal{E}_{S \to T} = [\mathbf{h}_i^{(1)}, \mathbf{h}_i^{(2)}, ..., \mathbf{h}_i^{(L)}], \tag{2.10}$$

where L is the length of the source input $\mathbf{X}_i^S$. We eliminate the index i from the LHS of Equation 2.10 and in the following equations of this subsection to avoid cluttering.

The decoder maps the representation $\mathcal{E}_{S\to T}$ into the target modality $\mathbf{X}^T$. This is performed by decoding each token $\mathbf{X}_t^T$ at a time based on $\mathcal{E}_{S\to T}$ and all previous decoded tokens, which is formulated as (in terms of probability):

$$p(\mathbf{X}^T) = \prod_{\ell=1}^{L} p(\mathbf{X}_\ell^T | \mathcal{E}_{S\to T}, \mathbf{X}_1^T, ..., \mathbf{X}_{\ell-1}^T). \tag{2.11}$$

MCTN accepts variable-length inputs of $\mathbf{X}^S$ and $\mathbf{X}^T$, and is trained to maximize the translational condition probability $p(\mathbf{X}^T|\mathbf{X}^S)$. The best translation sequence is then given by

$$\hat{\mathbf{X}}^T = \arg\max_{\mathbf{X}^T} p(\mathbf{X}^T|\mathbf{X}^S). \tag{2.12}$$

We use the traditional beam search approach (**?**) for decoding.

To obtain the joint representation for multimodal prediction, we only use the forward translated representation during inference to remove the dependency on the target modality at test time. If the cyclic translation is used, we denote the translated representation with the symbol $\leftrightarrows$:

$$\mathcal{E}_{S\leftrightarrows T} = \mathcal{E}_{S\to T}. \tag{2.13}$$

$\mathcal{E}_{S\leftrightarrows T}$ is then used for sentiment prediction:

$$\hat{\mathbf{y}} = g_w(\mathcal{E}_{S\leftrightarrows T}). \tag{2.14}$$

2.1.3.4 Coupled Translation-Prediction Objective

Training is performed with paired multimodal data and labels in the training set $(\mathbf{X}_{tr}^S, \mathbf{X}_{tr}^T, \mathbf{y}_{tr})$. The first two losses are the forward translation loss $\mathcal{L}_t$ defined as

$$\mathcal{L}_t = \mathbb{E}[\ell_{\mathbf{X}^T}(\hat{\mathbf{X}}^T, \mathbf{X}^T)], \tag{2.15}$$

and the cycle consistency loss $\mathcal{L}_c$ defined as

$$\mathcal{L}_c = \mathbb{E}[\ell_{\mathbf{X}^S}(\hat{\mathbf{X}}^S, \mathbf{X}^S)] \tag{2.16}$$

where $\ell_{\mathbf{X}^T}$ and $\ell_{\mathbf{X}^S}$ represent the respective loss functions. We use the Mean Squared Error (MSE) between the ground truth and translated modalities. Finally, the prediction loss $\mathcal{L}_p$ is defined as

$$\mathcal{L}_p = \mathbb{E}[\ell_{\mathbf{y}}(\hat{\mathbf{y}}, \mathbf{y})] \tag{2.17}$$

with a loss function $\ell_{\mathbf{y}}$ defined over the labels.

Our MCTN model is trained end-to-end with a coupled translation-prediction objective function defined as

$$\mathcal{L} = \lambda_t \mathcal{L}_t + \lambda_c \mathcal{L}_c + \mathcal{L}_p, \tag{2.18}$$

where λ_t, λ_t are weighting hyperparameters. MCTN parameters are learned by minimizing this objective function

$$\theta_e^*, \theta_d^*, w^* = \arg\min_{\theta_e, \theta_d, w} [\lambda_t \mathcal{L}_t + \lambda_c \mathcal{L}_c + \mathcal{L}_p]. \tag{2.19}$$

Parallel multimodal data is not required at test time. Inference is performed using only the source modality $\mathbf{X}^S$:

$$\mathcal{E}_{S \leftrightarrows T} = f_{\theta_e^*}(\mathbf{X}^S), \tag{2.20}$$

$$\hat{\mathbf{y}} = g_{w^*}(\mathcal{E}_{S \leftrightarrows T}). \tag{2.21}$$

This is possible because the encoder $f_{\theta_e^*}$ has been trained to translate the source modality $\mathbf{X}^S$ into a joint representation $\mathcal{E}_{S \leftrightarrows T}$ that captures information from both source and target modalities.

2.1.3.5 Hierarchical MCTN for Three Modalities

We extend the MCTN in a hierarchical manner to learn joint representations from more than two modalities. Figure 2.4 shows the case for three modalities. The hierarchical MCTN starts with a source modality $\mathbf{X}^S$ and two target modalities $\mathbf{X}^{T_1}$ and $\mathbf{X}^{T_2}$. To learn joint representations, two levels of modality translations are performed. The first level learns a joint representation from $\mathbf{X}^S$ and $\mathbf{X}^{T_1}$ using multimodal cyclic translations as defined previously. At the second level, a joint representation is learned hierarchically by translating the first representation $\mathcal{E}_{S \to T_1}$ into $\mathbf{X}^{T_2}$. For more than three modalities, the modality translation process can be repeated hierarchically.

Two Seq2Seq models are used in the hierarchical MCTN for three modalities, denoted as encoder-decoder pairs $(f_{\theta_e}^1, f_{\theta_d}^1)$ and $(f_{\theta_e}^2, f_{\theta_d}^2)$. A multimodal cyclic translation is first performed between source modality $\mathbf{X}^S$ and the first target modality $\mathbf{X}^{T_1}$. The forward translation is defined as

$$\mathcal{E}_{S \to T_1} = f_{\theta_e}^1(\mathbf{X}_{tr}^S),\ \hat{\mathbf{X}}_{tr}^{T_1} = f_{\theta_d}^1(\mathcal{E}_{S \to T_1}), \tag{2.22}$$

and followed by the decoded modality $\mathbf{X}^{T_1}$ being translated back into modality $\mathbf{X}^S$:

$$\mathcal{E}_{T_1 \to S} = f_{\theta_e}^1(\hat{\mathbf{X}}_{tr}^{T_1}),\ \hat{\mathbf{X}}_{tr}^{S} = f_{\theta_d}^1(\mathcal{E}_{T_1 \to S}). \tag{2.23}$$

A second hierarchical Seq2Seq model is applied on the outputs of the first encoder $f_{\theta_e}^1$:

$$\mathcal{E}_{S \leftrightarrows T_1} = \mathcal{E}_{S \to T_1}, \tag{2.24}$$

$$\mathcal{E}_{(S \leftrightarrows T_1) \to T_2} = f_{\theta_e}^2(\mathcal{E}_{S \leftrightarrows T_1}),\ \hat{\mathbf{X}}_{tr}^{T_2} = f_{\theta_d}^2(\mathcal{E}_{(S \leftrightarrows T_1) \to T_2}). \tag{2.25}$$

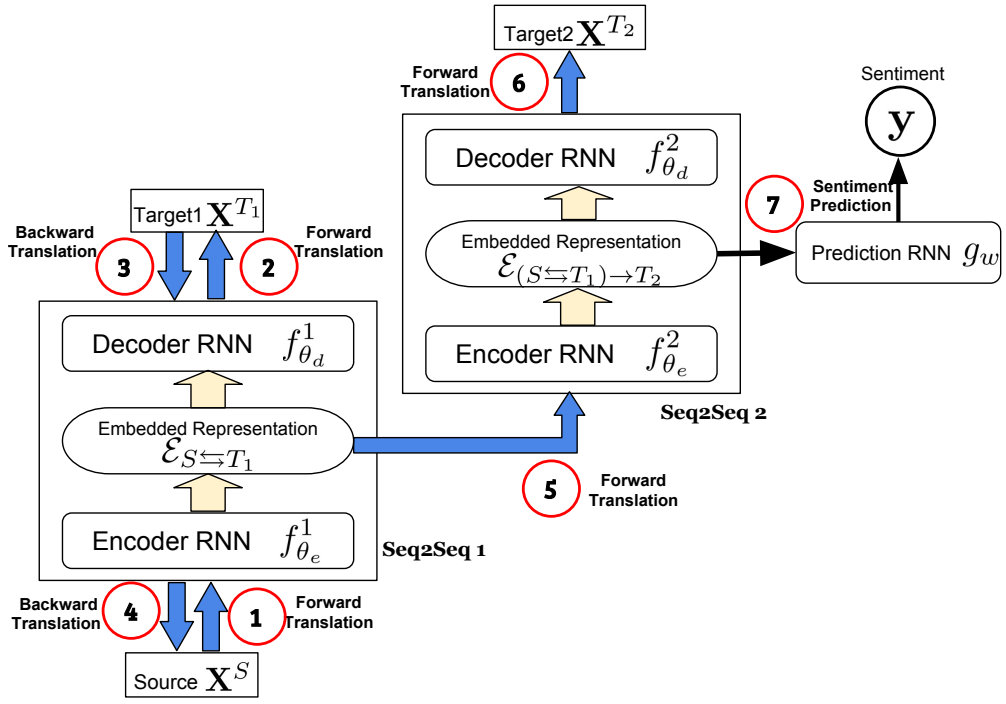


Figure 2.4: Hierarchical MCTN for three modalities: the source modality $\mathbf{X}^S$ and the target modalities $\mathbf{X}^{T_1}$ and $\mathbf{X}^{T_2}$. The joint representation $\mathcal{E}_{S\leftrightarrows T_1}$ is obtained via a cyclic translation between $\mathbf{X}^S$ and $\mathbf{X}^{T_1}$, then further translated into $\mathbf{X}^{T_2}$. Next, the joint representation of all three modalities, $\mathcal{E}_{(S\leftrightarrows T_1)\to T_2}$, is used for sentiment prediction. The model is trained end-to-end with a coupled translation-prediction objective. At test time, only the source modality $\mathbf{X}^S$ is required for prediction.

The joint representation between modalities $\mathbf{X}^S$, $\mathbf{X}^{T_1}$ and $\mathbf{X}^{T_2}$ is now $\mathcal{E}_{(S \leftrightarrows T_1) \rightarrow T_2}$. It is used for sentiment prediction via a recurrent neural network via regression method.

Training the hierarchical MCTN involves computing a cycle consistent loss for modality T_1, given by the respective forward translation loss $\mathcal{L}_{t_1}$ and the cycle consistency loss $\mathcal{L}_{c_1}$. We do not use a cyclic translation loss when translating from $\mathcal{E}_{S \leftrightarrows T_1}$ to $\mathbf{X}^{T_2}$ since the ground truth $\mathcal{E}_{S \leftrightarrows T_1}$ is unknown, and so only the translation loss $\mathcal{L}_{t_2}$ is computed. The final objective for hierarchical MCTN is given by

$$\mathcal{L} = \lambda_{t_1} \mathcal{L}_{t_1} + \lambda_{c_1} \mathcal{L}_{c_1} + \lambda_{t_2} \mathcal{L}_{t_2} + \mathcal{L}_p. \tag{2.26}$$

We emphasize that for MCTN with three modalities, *only* a single source modality $\mathbf{X}^S$ is required at test time. Therefore, MCTN has a significant advantage over existing models since it is robust to noisy or missing target modalities.

2.1.4 Experimental Setup

In this section, we describe our experimental methodology to evaluate the joint representations learned by MCTN[1]

2.1.4.1 Dataset and Input Modalities

We use the CMU Multimodal Opinion-level Sentiment Intensity dataset (CMU-MOSI) which contains 2199 video segments each with a sentiment label in the range $[-3, +3]$. To be consistent with prior work, we use 52 segments for training, 10 for validation, and 31 for testing. The same speaker does not appear in both training and testing sets to ensure that our model learns speaker-independent representations. We also run experiments on ICT-MMMO (Wöllmer et al., 2013) and YouTube (Morency et al., 2011) which consist of online review videos annotated for the sentiment.

2.1.4.2 Multimodal Features

Following previous work (Liang et al., 2018), GloVe word embeddings (Pennington et al., 2014), Facet (iMotions, 2017), and COVAREP (Degottex et al., 2014) features are extracted for the language, visual and acoustic modalities respectively. The detail for each modality is as follows.

Language: We used 300-dimensional Glove word embeddings trained on 840 billion tokens from the common crawl dataset (Pennington et al., 2014). These word embeddings were used to embed a sequence of individual words from video segment transcripts into a sequence of word vectors that represent spoken text.

Visual: The library Facet (iMotions, 2017) is used to extract a set of visual features including facial action units, facial landmarks, head pose, gaze tracking, and HOG features (Zhu et al.,

[1]Our source code is released at https://github.com/hainow/MCTN.

Dataset		CMU-MOSI			
Model	Test Inputs	Acc(↑)	F1(↑)	MAE(↓)	Corr(↑)
RF	$\{\ell, v, a\}$	56.4	56.3	-	-
SVM	$\{\ell, v, a\}$	71.6	72.3	1.100	0.559
THMM	$\{\ell, v, a\}$	50.7	45.4	-	-
EF-HCRF	$\{\ell, v, a\}$	65.3	65.4	-	-
EF-LDHCRF	$\{\ell, v, a\}$	64.0	64.0	-	-
MV-HCRF	$\{\ell, v, a\}$	44.8	27.7	-	-
MV-LDHCRF	$\{\ell, v, a\}$	64.0	64.0	-	-
CMV-HCRF	$\{\ell, v, a\}$	44.8	27.7	-	-
CMV-LDHCRF	$\{\ell, v, a\}$	63.6	63.6	-	-
EF-HSSHCRF	$\{\ell, v, a\}$	63.3	63.4	-	-
MV-HSSHCRF	$\{\ell, v, a\}$	65.6	65.7	-	-
DF	$\{\ell, v, a\}$	74.2	74.2	1.143	0.518
EF-LSTM	$\{\ell, v, a\}$	74.3	74.3	1.023	0.622
EF-SLSTM	$\{\ell, v, a\}$	72.7	72.8	1.081	0.600
EF-BLSTM	$\{\ell, v, a\}$	72.0	72.0	1.080	0.577
EF-SBLSTM	$\{\ell, v, a\}$	73.3	73.2	1.037	0.619
MV-LSTM	$\{\ell, v, a\}$	73.9	74.0	1.019	0.601
BC-LSTM	$\{\ell, v, a\}$	75.2	75.3	1.079	0.614
TFN	$\{\ell, v, a\}$	74.6	74.5	1.040	0.587
GME-LSTM(A)	$\{\ell, v, a\}$	76.5	73.4	0.955	-
MARN	$\{\ell, v, a\}$	77.1	77.0	0.968	0.625
MFN	$\{\ell, v, a\}$	77.4	77.3	0.965	0.632
LMF	$\{\ell, v, a\}$	76.4	75.7	0.912	0.668
RMFN	$\{\ell, v, a\}$	78.4	78.0	0.922	**0.681**
MCTN (Ours)	$\{\ell\}$	**79.3**	**79.1**	**0.909**	0.676

Table 2.1: Sentiment prediction results on CMU-MOSI. Best results are highlighted in bold. MCTN outperforms the current state-of-the-art across most evaluation metrics and uses only the language modality during testing.

2006). These visual features are extracted from the full video segment at 30Hz to form a sequence of facial gesture measures throughout time.

Acoustic: The software COVAREP (Degottex et al., 2014) is used to extract acoustic features including 12 Mel-frequency cepstral coefficients, pitch tracking and voiced/unvoiced segmenting features (Drugman and Alwan, 2011), glottal source parameters (Alku, 1992; Alku et al., 2002, 1997; Childers and Lee, 1991; Drugman et al., 2012), peak slope parameters and maxima dispersion quotients (Kane and Gobl, 2013). These visual features are extracted from the full audio clip of

Dataset		**ICT-MMMO**		**YouTube**	
Model	Test Inputs	Acc(↑)	F1(↑)	Acc(↑)	F1(↑)
RF	$\{\ell, v, a\}$	70.0	69.8	33.3	32.3
SVM	$\{\ell, v, a\}$	68.8	68.7	42.4	37.9
THMM	$\{\ell, v, a\}$	53.8	53.0	42.4	27.9
EF-HCRF	$\{\ell, v, a\}$	50.0	50.3	44.1	43.8
EF-LDHCRF	$\{\ell, v, a\}$	73.8	73.1	45.8	45.0
MV-HCRF	$\{\ell, v, a\}$	36.3	19.3	27.1	19.7
MV-LDHCRF	$\{\ell, v, a\}$	68.8	67.1	44.1	44.0
CMV-HCRF	$\{\ell, v, a\}$	36.3	19.3	30.5	14.3
CMV-LDHCRF	$\{\ell, v, a\}$	51.3	51.4	42.4	42.0
EF-HSSHCRF	$\{\ell, v, a\}$	50.0	51.3	37.3	35.6
MV-HSSHCRF	$\{\ell, v, a\}$	62.5	63.1	44.1	44.0
DF	$\{\ell, v, a\}$	65.0	58.7	45.8	32.0
EF-LSTM	$\{\ell, v, a\}$	66.3	65.0	44.1	43.6
EF-SLSTM	$\{\ell, v, a\}$	72.5	70.9	40.7	41.2
EF-BLSTM	$\{\ell, v, a\}$	63.8	49.6	42.4	38.1
EF-SBLSTM	$\{\ell, v, a\}$	62.5	49.0	37.3	33.2
MV-LSTM	$\{\ell, v, a\}$	72.5	72.3	45.8	43.3
BC-LSTM	$\{\ell, v, a\}$	70.0	70.1	45.0	45.1
TFN	$\{\ell, v, a\}$	72.5	72.6	45.0	41.0
MARN	$\{\ell, v, a\}$	71.3	70.2	48.3	44.9
MFN	$\{\ell, v, a\}$	73.8	73.1	**51.7**	51.6
MCTN (Ours)	$\{\ell\}$	**81.3**	**80.8**	**51.7**	**52.4**

Table 2.2: Sentiment prediction results on ICT-MMMO and YouTube. Best results are highlighted in bold. MCTN outperforms the current state-of-the-art across most evaluation metrics and uses only the language modality during testing.

each segment at 100Hz to form a sequence that represents variations in tone of voice over an audio segment.

2.1.4.3 Multimodal Alignment

We perform forced alignment using P2FA (Yuan and Liberman, 2008) to obtain the exact utterance time-stamp of each word. This allows us to align the three modalities together. Since words are considered the basic units of language we use the interval duration of each word utterance as one time-step. We acquire the aligned video and audio features by computing the expectation of their modality feature values over the word utterance time interval (Liang et al., 2018).

2.1.4.4 Evaluation Metrics

For parameter optimization on CMU-MOSI, the prediction loss function is set as the Mean Absolute Error (MAE): $\ell_p(\hat{\mathbf{y}}_{train}, \mathbf{y}_{train}) = |\hat{\mathbf{y}}_{train} - \mathbf{y}_{train}|$. We report MAE and Pearson's correlation r. We also perform sentiment classification on CMU-MOSI and report binary accuracy (Acc) and F1 score (F1). On ICT-MMMO and YouTube, we set the prediction loss function as categorical cross-entropy and report sentiment classification and F1 score. For all metrics, higher values indicate stronger performance, except MAE where lower values indicate stronger performance.

2.1.4.5 Baseline Models

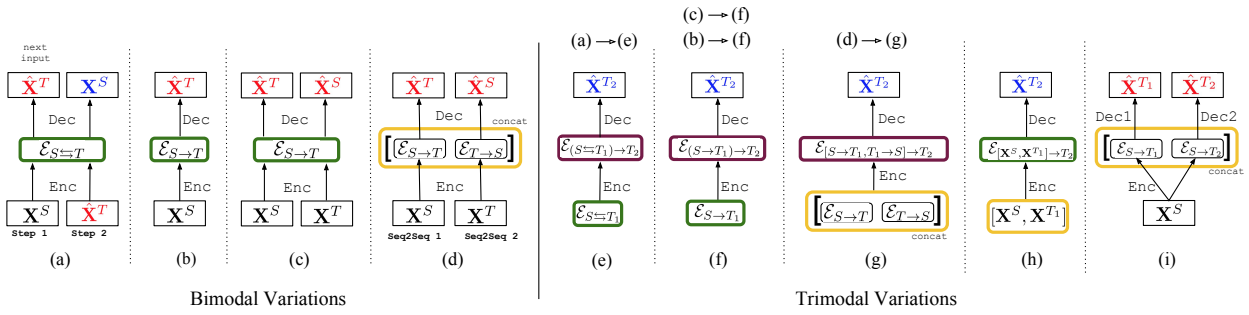


Figure 2.5: Variations of our models: (a) MCTN Bimodal with cyclic translation, (b) Simple Bimodal without cyclic translation, (c) No-Cycle Bimodal with different inputs of the same modality pair, and without cyclic translation, (d) Double Bimodal for two modalities without cyclic translation, with two different inputs (of the same pair), (e) MCTN Trimodal with input from (a), (f) Simple Trimodal for three modalities, with input as a joint representation taken from previous MCTN for two modalities from (b) or (c), (g) Double Trimodal with input from (d), (h) Concat Trimodal which is similar to (b) but with input as the concatenation of 2 modalities, (i) Paired Trimodal using one encoder and 2 separate decoders for modality translations. *Legend*: black modality is ground truth, red ("hat") modality represents translated output, blue ("hat") modality is target output from previous translation outputs, and yellow box denotes concatenation.

We compare to the following multimodal models: *RMFN* (Liang et al., 2018) uses a multistage approach to learn hierarchical representations (current state-of-the-art on CMU-MOSI). *LMF* (Liu et al., 2018) approximates the expensive tensor products in *TFN* (Zadeh et al., 2017) with efficient low-rank factors. *MFN* (Zadeh et al., 2018) synchronizes sequences using a multimodal gated memory. *EF-LSTM* concatenates multimodal inputs and uses a single LSTM (Hochreiter and Schmidhuber, 1997).

We also implement the Stacked, a.k.a. *EF-SLSTM* (Graves et al., 2013), Bidirectional, a.k.a. *EF-BLSTM* (Schuster and Paliwal, 1997), and Stacked Bidirectional (*EF-SBLSTM*) LSTMs, as well as the following baselines: *BC-LSTM* (Poria et al., 2017), *EF-HCRF* (Quattoni et al., 2007), *EF/MV-*

Dataset		CMU-MOSI			
Model	Translation	Acc	F1	MAE	Corr
MCTN Bimodal (2.5a)	$V \leftrightarrows A$	53.1	53.2	1.420	0.034
	$T \leftrightarrows A$	76.4	76.4	0.977	0.636
	$T \leftrightarrows V$	76.8	76.8	1.034	0.592
MCTN Trimodal (2.5e)	$(V \leftrightarrows A) \rightarrow T$	56.4	56.3	1.455	0.151
	$(T \leftrightarrows A) \rightarrow V$	78.7	78.8	0.960	0.650
	$(T \leftrightarrows V) \rightarrow A$	**79.3**	**79.1**	**0.909**	**0.676**

Table 2.3: MCTN performance improves as more modalities are introduced for cyclic translations during training.

LDHCRF (Morency et al., 2007), *MV-HCRF* (Song et al., 2012), *EF/MV-HSSHCRF* (Song et al., 2013), *MV-LSTM* (Rajagopalan et al., 2016), *DF* (Nojavanasghari et al., 2016), *SAL-CNN* (Wang et al., 2016), *C-MKL* (Poria et al., 2015), *THMM* (Morency et al., 2011), *SVM* (Cortes and Vapnik, 1995; Park et al., 2014) and *RF* (Breiman, 2001).

2.1.5 Results and Discussion

This section presents and discusses our experimental results.

2.1.5.1 Comparison with Existing Work

Q1: How does MCTN compare with existing state-of-the-art approaching for multimodal sentiment analysis?

We compare MCTN with previous models. From Table 2.1, MCTN using language as the source modality achieves new start-of-the-art results on CMU-MOSI for multimodal sentiment analysis. State-of-the-art results are also achieved on ICT-MMMO and YouTube (Table 2.2). It is important to note that MCTN only uses language during testing, while other baselines use all three modalities.

2.1.5.2 Adding More Modalities

Q2: What is the impact of increasing the number of modalities during training for MCTN with cyclic translations?

We run experiments with MCTN using combinations of two or three modalities with cyclic translations. From Table 2.3, we observe that adding more modalities improves performance, indicating that the joint representations learned are leveraging the information from more input modalities. This also implies that cyclic translations are a viable method to learn joint representations from multiple modalities since little information is lost from adding more modality

Dataset		CMU-MOSI			
Model	Translation	Acc(↑)	F1(↑)	MAE(↓)	Corr(↑)
MCTN Bimodal (2.5a)	$V \leftrightarrows A$	53.1	53.2	1.420	0.034
	$T \leftrightarrows A$	76.4	76.4	0.977	**0.636**
	$T \leftrightarrows V$	**76.8**	**76.8**	1.034	0.592
Simple Bimodal (2.5b)	$V \rightarrow A$	55.4	55.5	1.422	0.119
	$T \rightarrow A$	74.2	74.2	0.988	0.616
	$T \rightarrow V$	75.7	75.6	1.002	0.617
No-Cycle Bimodal (2.5c)	$V \rightarrow A,\ A \rightarrow V$	55.4	55.5	1.422	0.119
	$T \rightarrow A,\ A \rightarrow T$	75.5	75.6	**0.971**	0.629
	$T \rightarrow V,\ V \rightarrow T$	75.2	75.3	0.972	0.627
Double Bimodal (2.5d)	$[V \rightarrow A, A \rightarrow V]$	57.0	57.1	1.502	0.168
	$[T \rightarrow A, A \rightarrow T]$	72.3	72.3	1.035	0.578
	$[T \rightarrow V, V \rightarrow T]$	73.3	73.4	1.020	0.570

Table 2.4: Bimodal variations results on CMU-MOSI dataset. MCTN Bimodal with cyclic translations performs best.

translations. Another observation is that using language as the source modality always leads to the best performance, which is intuitive since the language modality contains the most discriminative information for sentiment (Zadeh et al., 2017).

In addition, we visually inspect the joint representations learned from MCTN as we add more modalities during training (see Table 2.5). The joint representations for each segment in CMU-MOSI are extracted from the best-performing model for each number of modalities and then projected into two dimensions via the t-SNE algorithm (van der Maaten and Hinton, 2008). Each point is colored red or blue depending on whether the video segment is annotated for positive or negative sentiment. From Figure 2.6, we observe that the joint representations become increasingly separable as more modalities are added when the MCTN is trained. This is consistent with increasing discriminative performance with more modalities (as seen in Table 2.3).

2.1.5.3 Ablation Studies

We use several models to test our design decisions. Specifically, we evaluate the impact of cyclic translations, modality ordering, and hierarchical structure.

For bimodal MCTN, we design the following ablation models shown in the left half of Figure 2.5: (a) MCTN bimodal between $\mathbf{X}^S$ and $\mathbf{X}^T$, (b) simple bimodal by translating from $\mathbf{X}^S$ to $\mathbf{X}^T$ without cyclic loss, (c) no-cycle bimodal which does not use cyclic translations but rather performs two independent translations between $\mathbf{X}^S$ and $\mathbf{X}^T$, (d) double bimodal: two Seq2Seq models with different inputs (of the same modality pair) and then using the concatenation of the

Dataset		CMU-MOSI			
Model	Translation	Acc(↑)	F1(↑)	MAE(↓)	Corr(↑)
MCTN Trimodal (2.5e)	$(V \leftrightarrows A) \rightarrow T$	56.4	56.3	1.455	0.151
	$(T \leftrightarrows A) \rightarrow V$	78.7	78.8	0.960	0.650
	$(T \leftrightarrows V) \rightarrow A$	**79.3**	**79.1**	**0.909**	**0.676**
Simple Trimodal (2.5f)	$(V \rightarrow T) \rightarrow A$	54.1	52.9	1.408	0.040
	$(V \rightarrow A) \rightarrow T$	52.0	51.9	1.439	0.015
	$(A \rightarrow V) \rightarrow T$	56.6	56.7	1.593	0.067
	$(A \rightarrow T) \rightarrow V$	54.1	54.2	1.577	0.028
	$(T \rightarrow A) \rightarrow V$	74.3	74.4	1.001	0.609
	$(T \rightarrow V) \rightarrow A$	74.3	74.4	0.997	0.596
Double Trimodal (2.5g)	$[T \rightarrow V,\ V \rightarrow T] \rightarrow A$	73.3	73.1	1.058	0.578
Concat Trimodal (2.5h)	$[V, A] \rightarrow T$	55.0	54.6	1.535	0.176
	$[A, T] \rightarrow V$	73.3	73.4	1.060	0.561
	$[T, V] \rightarrow A$	72.3	72.3	1.068	0.576
	$A \rightarrow [T, V]$	55.5	55.6	1.617	0.056
	$T \rightarrow [A, V]$	75.7	75.7	0.958	0.634
	$[T, A] \rightarrow [T, V]$	73.2	73.2	1.008	0.591
	$[T, V] \rightarrow [T, A]$	74.1	74.1	0.999	0.607
Paired Trimodal (2.5i)	$[T \rightarrow A, T \rightarrow V]$	73.8	73.8	1.022	0.611

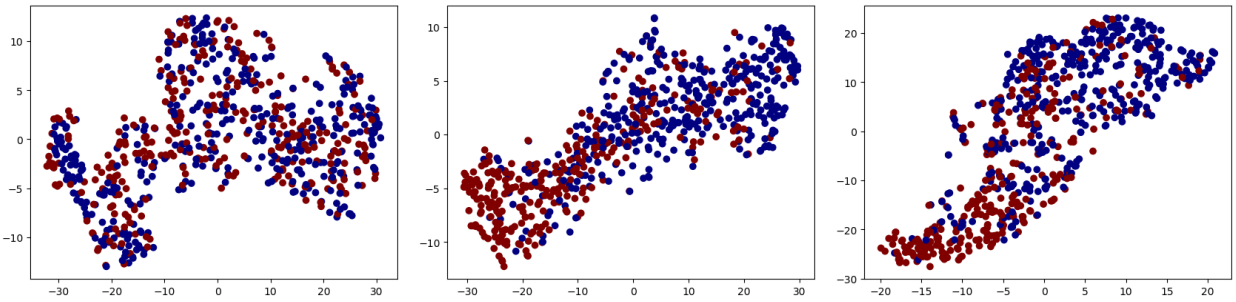
Table 2.5: Trimodal variations results on CMU-MOSI dataset. MCTN (hierarchical) with cyclic translations performs best.

joint representations $\mathcal{E}_{S \rightarrow T}$ and $\mathcal{E}_{T \rightarrow S}$ as the final embeddings.

For trimodal MCTN, we design the following ablation models shown in the right half of Figure 2.5: (e) MCTN trimodal which uses the proposed hierarchical translations between $\mathbf{X}^S$, $\mathbf{X}^{T_1}$ and $\mathbf{X}^{T_2}$, (f) simple trimodal based on a translation from $\mathbf{X}^S$ to $\mathbf{X}^{T_1}$ without cyclic translations, (g) double trimodal extended from (d) which does not use cyclic translations but rather performs two independent translations between $\mathbf{X}^S$ and $\mathbf{X}^{T_1}$, (h) concat trimodal which does not perform a first level of cyclic translation but directly translates the concatenated modality pair $[\mathbf{X}^S, \mathbf{X}^{T_1}]$ into $\mathbf{X}^{T_2}$, and finally, (i) paired trimodal which uses two separate decoders on top of the intermediate representation.

Q3: What is the impact of cyclic translations in MCTN?

The bimodal results are in Table 2.4. The models that employ cyclic translations (Figure 2.5(a)) outperform all other models. The trimodal results are in Table 2.5 and we make a similar observation: Figure 2.5(e) with cyclic translations outperforms the baselines (f), (g) and (h). The gap for the trimodal case is especially large. This implies that using cyclic translations is crucial for learning discriminative joint representations. Our intuition is that using cyclic translations: (1)



(a) MCTN Bimodal *without* cyclic translations

(b) MCTN Bimodal *with* cyclic translations

(c) MCTN Trimodal *with* cyclic translations

Figure 2.6: t-SNE visualization of the joint representations learned by MCTN. *Legend*: red: videos with negative sentiment, blue: videos with positive sentiment. Adding modalities and using cyclic translations improve discriminative performance and lead to increasingly separable representations.

encourages the model to enforce symmetry between the representations from source and target modalities thus adding a source of regularization, and (2) ensures that the representation retains maximal information from all modalities.

Q4: What is the effect of using two Seq2Seq models instead of one shared Seq2Seq model for cyclic translations?

We compare Figure 2.5(c), which uses one Seq2Seq model for cyclic translations with Figure 2.5(d), which uses two separate Seq2Seq models: one for forward translation and one for backward translation. We observe from Table 2.4 that (c) $>$ (d), so using one model with shared parameters is better. This is also true for hierarchical MCTN: (f) $>$ (g) in Table 2.5. We hypothesize that this is because training two deep Seq2Seq models requires more data and is prone to overfitting. Also, it does not learn only a single joint representation but instead two separate representations.

Q5: What is the impact of varying source and target modalities for cyclic translations?

From Tables 2.3, 2.4 and 2.5, we observe that language contributes most towards the joint representations. For bimodal cases, combining language with visual is generally better than combining the language and acoustic modalities. For hierarchical MCTN, presenting language as the source modality leads to the best performance, and a first level of cyclic translations between language and visual is better than between language and audio. On the other hand, only translating between visual and acoustic modalities dramatically decreases performance. Further adding language as a target modality for hierarchical MCTN will not help much as well. Overall, for the MCTN, language appears to be the most discriminative modality making it crucial to be used as the source modality during translations.

Q6: What is the impact of using two levels of translations instead of one level when learning from

three modalities?

Our hierarchical MCTN is shown in Figure 2.5(e). In Figure 2.5(h), we concatenate two modalities as input and use only one phase of translation. From Table 2.5, we observe that (e) $>$ (h): both levels of modality translations are important in the hierarchical MCTN. We believe that representation learning is easier when the task is broken down recursively: using two translations each between a single pair of modalities, rather than a single translation between all modalities.

2.1.6 Conclusion

This section investigated learning joint representations via cyclic translations from source to target modalities. During testing, we only need the source modality for prediction which ensures robustness to noisy or missing target modalities. We demonstrate that cyclic translations and Seq2Seq models are useful for learning joint representations in multimodal environments. In addition to achieving new state-of-the-art results on three datasets, our model learns increasingly discriminative joint representations with more input modalities while maintaining robustness to all target modalities.

2.2 Learning Robust Representation for Handwriting Recognition with Limited Data

Unlike the multimodal setting in the previous problem, we deal with yet another type of challenging data that is noisily scanned images of forms that are handwritten. Despite the advent of deep learning, especially in computer vision, the general handwriting recognition problem is far from solved. Previous works have achieved significant results using Hidden Markov Model (HMM) and Long Short Term Memory (LSTM). In this work, we design a novel approach to tackle the problem of offline handwritten form recognition in constrained settings. By using a generated synthetic dataset augmented with a Generative Adversarial Network-based image refinement technique and breaking down the problem into two consecutive modules of word segmentation and word classification, we show that our system outperforms other competitors in two modes: training with and without using any subset of the target dataset, using IAM-DB and an in-house dataset. Our approach can also be extended to the related problem of printed document recognition.

2.2.1 Introduction and Motivation

Another well-known yet unsolved problem that involves complex data is offline Handwriting Recognition, (HWR) (Plamondon and Srihari, 2000), in which we simply have access to an image of the final handwritten words and without any other cues such as in the online setting (Graves et al., 2009), where the model can adaptively learn and predict based on the progress of the writing on-the-way. Likewise, related to the previous sentiment analysis task (Section 2.1), this task requires the representation learning to map between 2 different modalities: given a scanned, noisy image of handwriting text (i.e. an image), we train a model able to recognize the content (i.e. text).

Despite the significant demand, there are few efficient methods able to tackle this problem due to the difficulty of designing a holistic solution suitable across various forms of input. The first challenge is to segment forms (i.e. images containing lines) properly to facilitate the recognition process. The most common method is to use a heuristics line-level segmentation (Graves et al., 2009; Liwicki et al., 2007). However, this is often impractical since words and characters are not usually handwritten along straight lines. The second challenge is to build a model capable of recognizing and generalizing diverse handwriting styles. Furthermore, in some resource-constrained settings where we have limited access to real data, it is infeasible to manually build large-scale handwriting recognition datasets such as IAM-DB (Marti and Bunke, 2002), or SD19 (Grother and Hanaoka, 1995). It therefore becomes necessary to find a powerful solution that does not require a large quantity of real data. This problem is ubiquitous in practice, in that we only have access to limited data with inherent noise. Typically in such settings, people rely on commercial systems which are prohibitively expensive, or open APIs such as Google Cloud Vi-

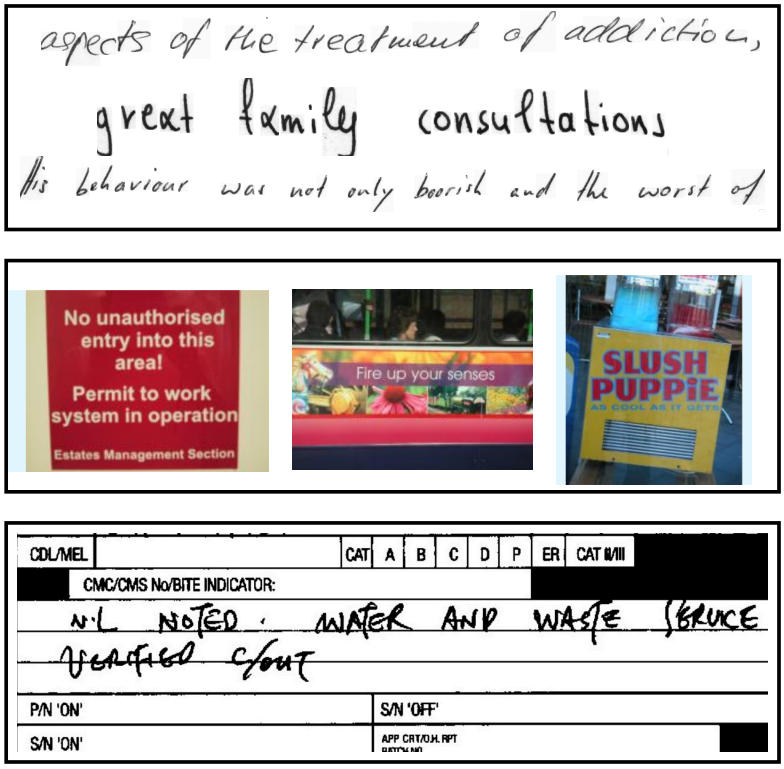


Figure 2.7: Some samples from 3 different problems. *Top*: three lines of IAM (Marti and Bunke, 2002) dataset which has handwritten text on blank background; most solutions segment them into lines without clarifying segmentation quality. *Middle*: ICDAR (Karatzas et al., 2015) dataset for scene-text recognition which has printed text with a random background. *Bottom*: our BHD dataset which combines the difficulties of the other two: multi-style, unaligned handwritten text in the whole form (not lines) and noisy background.

sion[2] or Tesseract (Smith, 2007) which typically perform poorly as they are mainly designed for *printed* text and for dealing with many languages with a single model.

Furthermore, our dataset is much more difficult than IAM-DB or SD19 as illustrated in Figure 2.7. First, it has limited and noisy data and annotation. Second, it combines the difficulties of the classical HWR datasets and scene-text detection and recognition ones. As a result, we modularize the problem into two stages in order to make it more tractable to train two separate deep models. In the first stage we employ a object detection model, such as R-FCN (Dai et al., 2016), to detect words from the background with various types of noise. The resulting segments are fed into a recognition model in the second stage which can be a word-based or a character-based model.

[2] https://cloud.google.com/vision/

2.2.2 Related Work

For offline HWR, there have been many achievements using the classical HMM-based models (Bahlmann and Burkhardt, 2004; Bunke and Varga, 2007; Fischer et al., 2012; Hu et al., 2000). Later, with the advent of deep learning, Recurrent Neural Network based approaches, such as using LSTM (Hochreiter and Schmidhuber, 1997), gained new successes in this setting (Kang et al., 2018; Puigcerver, 2017; Voigtlaender et al., 2015). Following this line, there have been also some other solutions that also employ convolutional neural network (CNN) such as in (Dutta et al., 2018), and using CNN plus language-based features (Krishnan et al., 2016; Poznanski and Wolf, 2016). However, in comparison to their settings, our variable-sized forms are more challenging for they include horizontal lines running across the document which contribute to noise since the text doesn't necessarily conform to these lines. Furthermore, the content is mixed with other random noises such as signatures, stamps, or other unrecognized marks caused by scanners or inks. Despite such difficult inputs, our model can directly process whole forms properly, in contrast to these existing solutions that rely on heuristic methods for line-level segmentation.

A closely related problem to our method is segmentation for which there have been some heuristic (Plamondon and Srihari, 2000) or HMM-based (Zimmermann and Bunke, 2002) methods. Our model instead relies on deep segmentation frameworks which are usually employed for object detection tasks (Dai et al., 2016; He et al., 2017; Klambauer et al., 2017; Lin et al., 2017; Redmon and Farhadi, 2018). Unlike those methods that learn to predict a regression bounding box and detect an object at the same time, in our segmentation phase, we reduce the task to a more tractable problem of only predicting a bounding box covering a word and leave the recognition job to a downstream task. We retain the order of the words while doing this so as to ensure that sentence or document-level meaning is retained.

2.2.3 Effective Representation In the Face of Noisy and Limited Data

Our inputs are rectangular images of varying sizes containing handwritten sentences, often in unaligned lines and with lots of noise and other irrelevant content such as stamps, signatures, and other types of random noise. Our goal is to recognize those relevant sentences and output the corresponding texts for further data analysis purposes.

2.2.3.1 Choice of Two-phase Model

As mentioned above, we design a two-phase approach (Figure 2.8) that segments the entire form into words (in the presence of noisy content) while maintaining their original order and recognizing each word individually. There are many reasons for this approach. First, we have very few annotated samples, thus the generalizability of our model is benefited from the inductive bias of the two-stage approach. Second, the difficulties of the forms are unusual. Due to unaligned texts, it is impossible to segment forms into lines without affecting the content as in other HWR

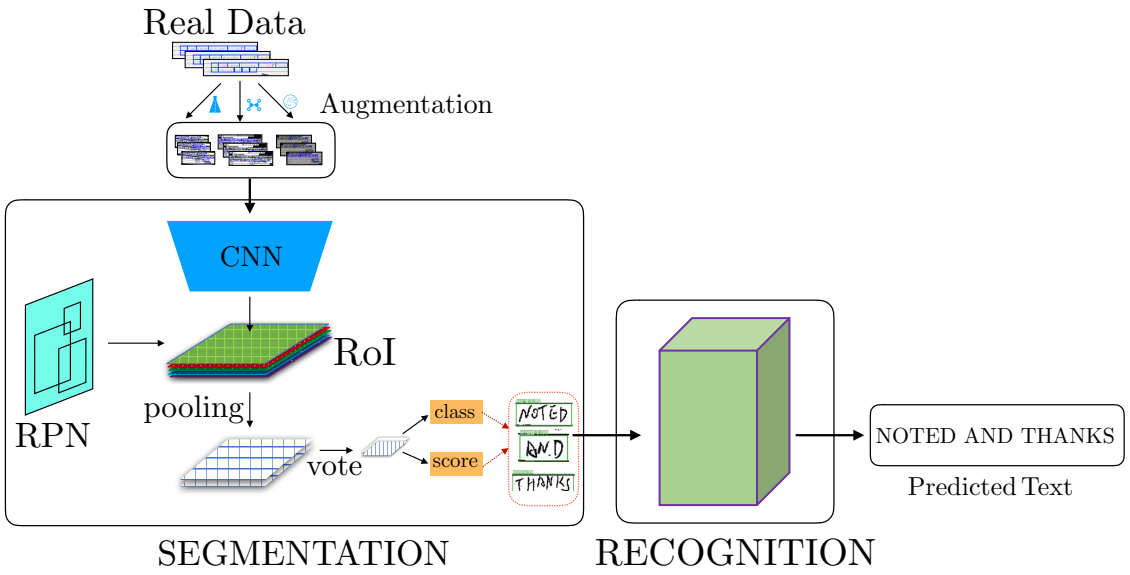


Figure 2.8: Our model uses data augmentation to train a segmentation module (locating words amidst background noise) and word recognition (word or character-based) models.

methods. Furthermore, like scene-text recognition datasets, our forms have many types of noise (Figure 2.7 and 2.9). Third, this approach is interpretable and easier to train and debug. Finally, it becomes easier to perform parallel training of the two stages across limited resources, allowing for better quality control and modularity in design.

2.2.3.2 Word Segmentation

Instead of trying to predict the correct bounding boxes and recognize the words inside simultaneously, the word segmentation phase only focuses on drawing correct bounding boxes at the word level and leaves the recognition job as a downstream task. We choose this design for the following reasons. First, word-level segmentation is used since separating spaces among words (as opposed to characters) is much more feasible in practice (especially in cursive handwriting). Second, as explained previously, line-level segmentation is not preferred since in our setting words are often not aligned horizontally.

In terms of architecture, since HWR is different from object detection where detection is only a proxy, we explore multiple options like R-FCN (Dai et al., 2016), Faster R-CNN (Girshick, 2015) and YOLO-v3 (Redmon and Farhadi, 2018) to identify which kind of architecture is most suited for our HWR pipeline. Although the core components of those detection methods remain unchanged, it is worth noting two important changes in adapting such methods. First, given word segmentation is an intermediate step, we simplify this phase by limiting the number of classes to only 5 (Figure 2.9), with the main goal being extracting the text out of the forms without having to recognize its content. Second, based on the nature of our dataset, we change the segmentation input to

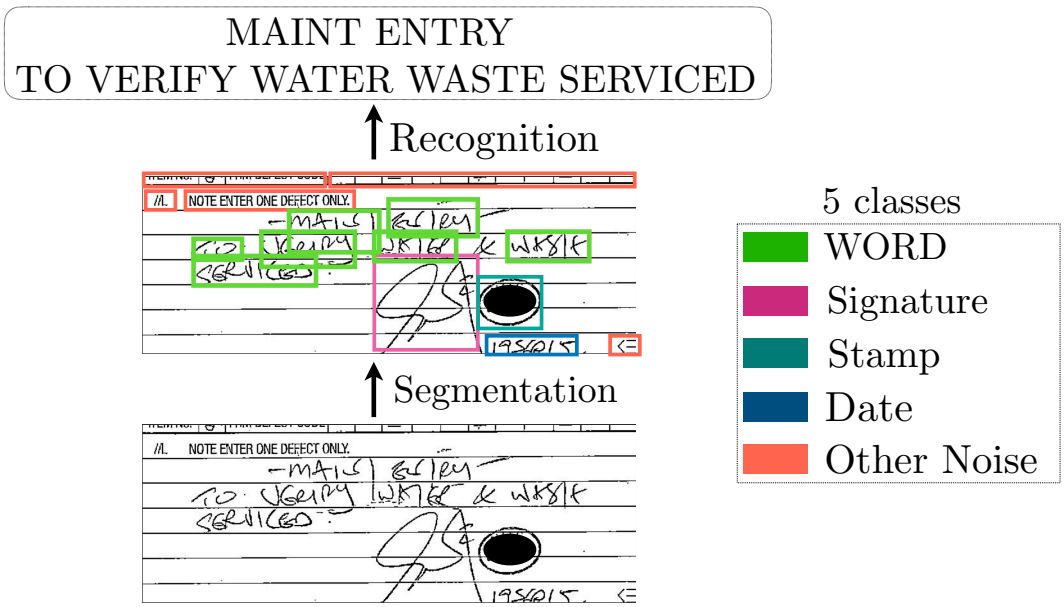


Figure 2.9: Illustration of an annotated example with 5 classes where class WORD is the main focus. Our model can handle noisy forms by localizing unaligned texts, and filtering out other types of noise to recognize the sentence(s) in the correct order of words. We hide stamps' contents for security reasons.

grayscale images with only 1 channel. As a result of these two adjustments, our segmentation phase is much easier and faster to train compared with their original uses.

2.2.3.3 Word Recognition

For each form, this module takes the bounding boxes (as images) from the Word Segmentation module as inputs, and outputs a word for each bounding box. Based on the coordinates given by the Word Segmentation module, we are able to reconstruct the entire sentence from individual words. And because of the complications of the input forms, we experiment with 3 different models namely Word Model, Character Model, and CTCSeq2Seq Model, as detailed below.

a) Word Model

The word model is a CNN-based image classification network that uses an augmented Resnet-18 (He et al., 2016) to predict words from a predefined word vocabulary. Furthermore, due to the low resolution of our input images, we adjust Resnet-18 to only have a stride size of 1 instead of 2 in the residual blocks. This model is simple but is only capable of predicting words within the predefined vocabulary of 998 words.

b) Character Model

This model shares its architecture with the Word Model, which enables the benefits of initializing weights from a pre-trained Word Model, except that it uses a CTC loss (Graves et al.,

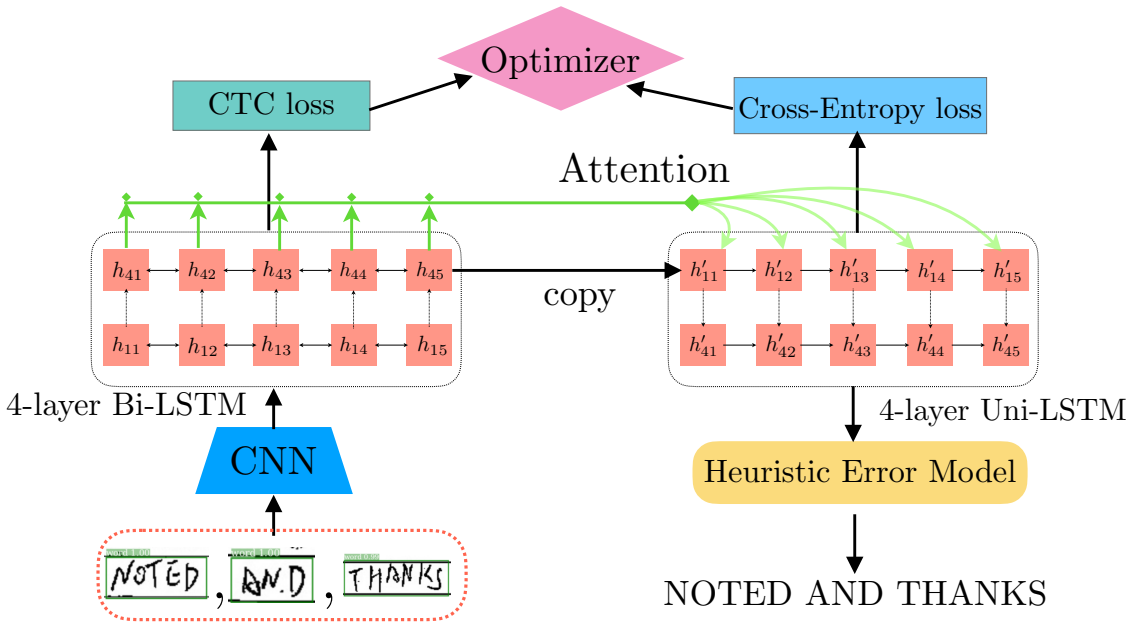


Figure 2.10: Our CTCSeq2Seq model contains 3 core modules: Feature Extraction which is CNN-based, Encoder, and Decoder that combined form a Seq2Seq model. The encoder uses *CTC loss* which helps with alignments of the frames to the outputs.

2009; Liwicki et al., 2007; Voigtlaender et al., 2016) instead of cross-entropy loss. For this reason, it predicts a sequence of characters instead of a single word at a time. Furthermore, the last fully-connected layer in Resnet-18 is replaced with a convolutional layer to reshape the output from $H \times W \times D$ to $1 \times W/2 \times C$, where C is the cardinality of the character prediction space.

By using CTC, this model has two advantages over Word Model. First, CTC largely reduces the prediction space from 998 words to 35 alpha-numeric characters (our dataset does not have the letter "Z"), making it agnostic to word vocabulary size. Second, it enables the model to predict unseen words.

c) CTCSeq2Seq Model

Our motivation for this model is to learn the embedded latent representation of images that can be decoded into text. As shown in Figure 2.10, the model can be broken down into 3 main blocks: Feature Extraction, Encoder, and Decoder. The model loss is the weighted sum of CTC loss (Encoder) and softmax cross-entropy loss (Decoder). Except for those 3 main modules, there is an edit-distance-based error module that corrects a predicted out-of-vocabulary word within a maximum of 2 wrong characters compared to a known word.

c.1) Feature Extraction: This module accepts variable-sized input images, each of which has a single word. It firstly resizes inputs to the same height but not necessarily to the same width. Next, it slices each one into small patches of equal widths (as illustrated in Figure 2.12). Finally, it extracts CNN-based features out of the patches using a custom VGG (Simonyan and

Zisserman, 2014).

c.2) Encoder: For our encoder, we use a 4-layer bidirectional LSTM that takes inputs from the Feature Extraction module. Since each input word is segmented into many sequential equal-height patches, the LSTM can model their relations into a hidden representation. Another key feature of this module is to have a CTC loss to enforce reconstruction of the original characters so that the embedded representation is learned effectively.

c.3) Decoder: This module is a 4-layer unidirectional LSTM that consumes the hidden representation from the Encoder and has an attention module (Luong et al., 2015) which calculates the weighted average of each output with the entire input sequence. This mechanism helps the model learn to focus on more important patches.

In addition, this module uses the softmax cross-entropy loss normalized by the length of the input, since we have variable-length sequences of patches. Finally, it also predicts among 35 alpha-numeric characters, the same as Character Model (Section 2.2.3.3) which also ignores punctuation in the datasets.

2.2.4 Experiments

Dataset	Type	Train	Valid	Test
Segmentation	Real	2,358	-	1,362
	+DA	40,159	-	-
Recognition	Real	6,639	3,400	1,249
	+DA	660,000	-	-
Pipeline	Real	-	-	1,362

Table 2.6: Statistics of BHD dataset. We have 2 types of data (i) Real and (ii) +DA: real images with data augmentation. For each model, we only have a single test set from real forms, and the one used for Pipeline evaluation is shared with Segmentation. Data augmentation is a key preprocessing step to get more samples and styles for training deep models.

2.2.4.1 Dataset

Our in-house BHD dataset, as shown in Table 2.6, comprises maintenance logbooks in which there are many aerospace terms or abbreviations that do not appear in the normal English vocabulary. Each image is grayscale and may contain from 3 to 50 bounding boxes. Moreover, in addition to the presence of unusual aerospace terms, there are many arbitrary part numbers (*e.g.*, "W308003-12239-22"). As mentioned earlier, our forms contain multiple horizontal lines, with signatures, stamps, dates, and other types of noise, making our task even more challenging. Finally, to create

word vocabulary, we use `tf-idf` to retrieve the first 1000 words from digitized maintenance logbooks, then remove 2 outliers to finally have 998 words.

Furthermore, our manual inspection of the BHD dataset reveals that in several cases the strokes from adjacent words are connected to each other, while in other cases, the characters in a word are quite far apart, which tempts any object detection model to confuse multiple words with just one. This makes BHD more challenging than ICDAR and other scene-text detection and recognition datasets.

2.2.4.2 Training Data Augmentation

Because we have limited data, and our model contains deep neural networks that are typically data-hungry, data augmentation is an important technique to increase the effective size of BHD prior to training and to improve the generalization capability of our models. In particular, we use two data augmentation techniques for both segmentation and recognition tasks. First, we use several types of noise including pepper, stroke, and Gaussian noises. Second, we employ local image transformations that are erosion, dilation, and flipping.

2.2.4.3 Evaluation Metrics

Segmentation: We use the canonical MaP metric (Everingham et al., 2010) to evaluate segmentation performance against our annotation in the BHP real-form test dataset.

Recognition: We use word accuracy (WA) and Character Error Rate (CER) to evaluate our recognition models. While WA simply calculates the average number of predicted words that exactly match with ground truths, CER is calculated as CER $= (\mathrm{D}(w_{gt}, w_{predict}) \times 100) \,/\, |w_{gt}|\ (\%)$, where $\mathrm{D}(w_{gt}, w_{predict})$ is the minimum Damerau-Levenshtein edit distance (Damerau, 1964) between the ground-truth word w_{gt} and predicted word $w_{predict}$, and $|w_{gt}|$ is the number of characters in w_{gt}.

Full Pipeline: Our pipeline takes the form of input and outputs a sequence of predicted words. Therefore we use Word Error Rate (WER) and CER to evaluate performances. For WER, we treat every word as a character. For CER, we concatenate the sequence of words by inserting a space between every two words and treating the concatenated sequence as the predicted string.

2.2.4.4 Baselines

Since different models require different sets of annotations (*e.g.* many HWR models expect noise-free input), we cannot fairly compare our full pipeline performances with many SoTA methods for HWR. As a result, the only close HWR pipeline we compare our model with is Convolve-Attend-Spell (Kang et al., 2018) (after it is fine-tuned on the full-pipeline dataset) which has the capability of accepting the entire form as an input and to some extent is also robust to noise.

However, we can compare each phase of our pipeline with segmentation and recognition baselines developed for scene-text detection. For segmentation, we use EAST (Zhou et al., 2017), PixelLink(Deng et al., 2018) and CRAFT (Baek et al., 2019). In order to have a fair comparison, we fine-tune EAST and PixelLink[3] (trained on ICDAR 2015 (Karatzas et al., 2015)) and only compare on the *word* class, which is the ultimate goal. For recognition, we use MORAN (Luo et al., 2019) which is pre-trained on synthetic images (Gupta et al., 2016; Jaderberg et al., 2014) and subsequently fine-tuned on BHD recognition training data. And last, for the full pipeline, we combine PixelLink and MORAN, for which the full training codes are available.

2.2.5 Results and Discussion

We compare the performances of our approach to the baselines for the full pipeline, segmentation, and recognition. We also perform an ablation study on the impact of segmentation on the full pipeline.

Segmentation	Recognition	WER(↓)	CER(↓)
Convolve-Attend-Spell		38.9	24.1
PixelLink	MORAN	80.7	47.4
R-FCN	Word (Ours)	31.5	22.9
	CTCSeq2Seq (Ours)	**30.1**	**18.5**

Table 2.7: Full pipeline performance of our best model compared to the baselines with the following components: Convolve-Attend-Spell (Kang et al., 2018), PixelLink (Deng et al., 2018), MORAN (Luo et al., 2019), R-FCN (Dai et al., 2016). Our model significantly outperforms all the baselines in both WER and CER metrics.

2.2.5.1 Full Pipeline Results

The full pipeline results are shown in Table 2.7. We observe that R-FCN (Dai et al., 2016) in conjunction with CTCSeq2Seq (both of which are trained on the +DA dataset) yields the best performance, and significantly outperforms the baseline models.

Furthermore, Figure 2.11 illustrates some qualitative results. The R-FCN is able to filter out several types of noise in each form and pick out the correct bounding boxes with almost 100% confidence for all words. Furthermore, our CTCSeq2Seq is able to detect words and characters of various styles, orientations, and intensities. However, the baseline one makes lots of mistakes in word localization, which are compounded in the second phase of recognition.

[3]The same cannot be done for CRAFT due to its code's unavailability.

	EAST	CRAFT	PixelLink	R-FCN	Faster-RCNN	YOLO-v3
AP (↑)	38.9	12.8	81.6	89.0	**89.1**	86.0

Table 2.8: AP score comparison on the *word* class (IoU=50%). Our three models significantly outperform the baselines.

2.2.5.2 Segmentation Results

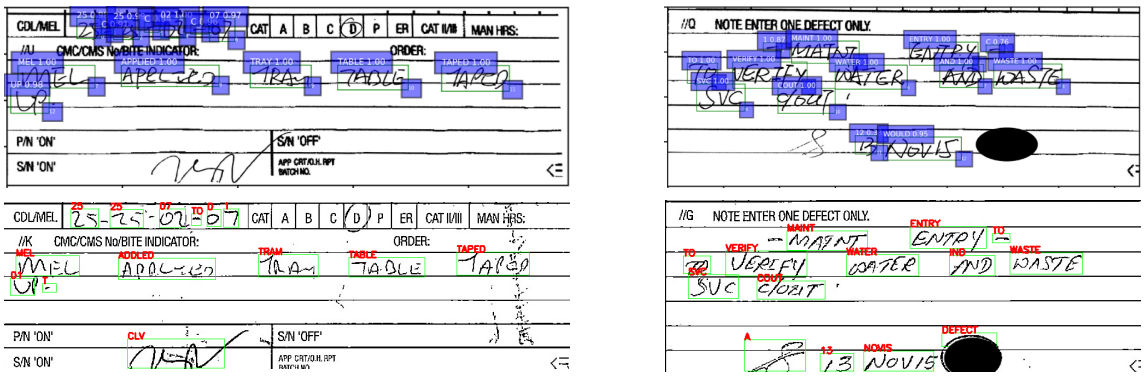


Figure 2.11: Full pipeline qualitative results of our model R-FCN (Dai et al., 2016) + CTCSeq2Seq (top) and the baseline PixelLink (Deng et al., 2018) + MORAN (Luo et al., 2019) (bottom). Ours performs much better in both locating words and recognizing them.

As shown in Table 2.8, our three segmentation models clearly outperform all baseline methods, especially on EAST and CRAFT. While EAST fails to split large bounding boxes, leading to a low recall (18.4%), CRAFT's pre-trained model mistakes printed words for handwritten text and therefore has a low precision (21.4%). Finally, since PixelLink is trained on BHD, it can achieve a decent score of 81.6% AP.

Additionally, considering only our models, Table 2.9 shows that data augmentation leads to improvements on AP for R-FCN (especially for rare categories like *Signature* or *Date*). R-FCN with position-based scores is particularly effective in tackling translation variance (Dai et al., 2016) for handwriting recognition where the Region-of-Interest (RoI) is fairly small (as seen in Figure 2.11).

2.2.5.3 Recognition Results

As demonstrated in Table 2.10, our Word Model achieves similar performances to the best performer MORAN in both WA and CER given ground-truth bounding boxes. Even being initialized with Word Model's pre-trained weights, the Character Model under-performs the other two by a huge margin. We suspect the reason is that CTC is hard to train, and may require more training data or more complex techniques.

Class	R-FCN (Real)	R-FCN (+DA)	Faster R-CNN (+DA)	YOLO-v3 (+DA)
Word	85.8	89.0	**89.1**	86.0
Signature	67.9	**78.2**	43.3	40.8
Stamp	86.6	**89.9**	10.7	84.2
Date	70.1	**82.9**	24.7	62.9
Noise	18.2	17.4	**27.3**	15.2
Average	65.7	**71.3**	39.0	57.8

Table 2.9: AP scores for Segmentation models R-FCN (Dai et al., 2016), Faster R-CNN (Klambauer et al., 2017), and YOLO v3 (Redmon and Farhadi, 2018). R-FCN significantly outperforms others in most classes with augmented training data (IoU=50%).

Model	Dataset	WA (↑)	CER(↓)
MORAN	Real	91.7	3.4
	+DA	**96.4**	**1.5**
Word (Ours)	Real	76.1	20.4
	+DA	**96.1**	**2.6**
Character (Ours)	Real	5.0	62.8
	+DA	76.3	9.7
CTCSeq2Seq (Ours)	Real	87.1	7.8
	+DA	94.9	3.2

Table 2.10: Comparison on recognition models (on Recognition dataset) given ground-truth bounding boxes. Our Word Model and MORAN (Luo et al., 2019) perform the best compared to others.

2.2.5.4 Ablation Study

We study how different segmentation models affect pipeline performance on the same recognition model. As shown in Table 2.11, our models perform much better than the baselines, and CTCSeq2Seq is the best recognition model. As shown in Figure 2.12, CTC loss combined with attention module significantly helps with character recognition, making the CTCSeq2Seq the best choice for our full pipeline.

And interestingly, CER increases much more than WER when replacing R-FCN with Faster R-CNN. Our empirical analysis reveals that R-FCN tends to give predictions with higher confidence scores and in difficult cases, it predicts more bounding boxes than Faster R-CNN in the segmentation phase. Finally, given ground-truth bounding boxes, both WER and CER decrease but only to a limited extent. This suggests that the segmentation module is not the bottleneck of

Recognition	Segmentation	WER(↓)	CER(↓)
MORAN	Ground Truth	49.2	25.7
	PixelLink	80.7	47.4
Word (Ours)	Ground Truth	15.1	9.5
	R-FCN	18.3	13.2
	Faster R-CNN	19.1	21.0
CTCSeq2Seq (Ours)	Ground Truth	**14.1**	**8.2**
	R-FCN	18.9	12.3
	Faster R-CNN	19.8	19.5

Table 2.11: Impact of different Segmentation methods on the full pipeline (on Pipeline dataset). Our models clearly outperform the baselines, and CER is much higher if we replace R-FCN by Faster R-CNN.

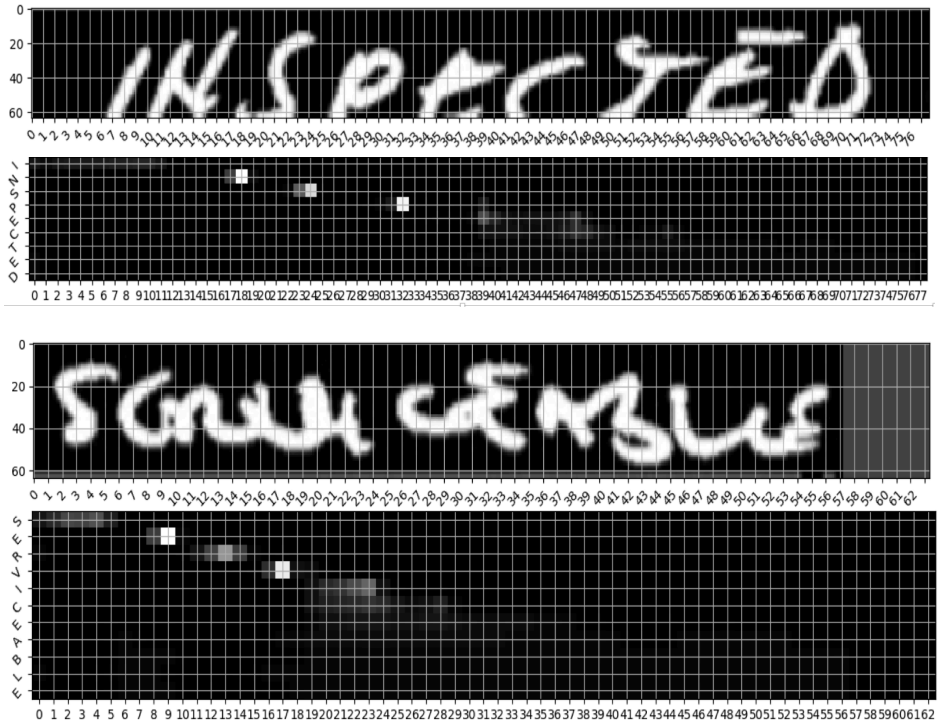


Figure 2.12: Attention map results of CTCSeq2seq model for 2 words: INSPECTED and SERVICEABLE. The upper image is raw input the lower one is the corresponding attention map. Brighter squares indicate higher weights (focusing more in decoding). After the first several characters are recognized, the model can infer the rest of the characters without relying on encoder information.

whole pipeline system, and we should focus more on the recognition module to increase pipeline performance.

2.2.6 Conclusion

In this part, we focused on HWR for noisy and challenging maintenance logs, a previously overlooked domain in this field. We presented a two-stage approach that can process the entire forms directly without the need of segmenting them into lines. Our experimental results show that our approach significantly outperforms the HWR and scene-text detection and recognition baselines on the full pipeline while achieving high accuracies on the individual phases of word segmentation and recognition.

2.3 Learning Long-Document Representation with Position-Aware Multimodal Attention

Despite several successes in document understanding, the practical task for long document understanding is largely under-explored due to several challenges in computation and how to efficiently absorb long multimodal input. Most current transformer-based approaches only deal with short documents and employ solely textual information for attention due to its prohibitive computation and memory limit. To address those issues in long document understanding, we explore different approaches in handling 1D and new 2D position-aware attention with essentially shortened context. Experimental results show that our proposed models have the advantages for this task based on various evaluation metrics. Furthermore, our model makes changes only to the attention and thus can be easily adapted to any transformer-based architecture.

2.3.1 Introduction

The task of document understanding has recently gleaned many successes (Appalaraju et al., 2021; Xu et al., 2020b, 2021). This task requires multimodal input that makes it heavier than the text-only ones, resulting in most models only being capable of dealing with short documents, i.e. having up to 512 tokens. However, there exist long documents almost everywhere, e.g. contracts, scientific papers, newsletters, or Wikipedia articles, which are typically longer than 1,000 words. To automatically summarize and understand such long documents urges long document understanding to become an important task in both natural language processing and artificial intelligence.

Long document understanding faces several big challenges. 1) Recent document understanding approaches heavily rely on transformer (Vaswani et al., 2017). However, the transformer suffers from quadratic attention that usually limits the input to 512 words. Therefore, the correlation across long paragraphs/pages is yet to be learned. 2) Understanding long documents requires the power to model all long information available, not only just in text but also in other modalities such as spatial information. For example, LayoutLM (Xu et al., 2020b) showed that short document understanding is largely improved by additionally embedding spatial into text information. How to efficiently make use of spatial information for long document understanding, however, is still an open and challenging problem regarding computation cost and adaptability.

Given the fact that long documents frequently appear in practice as well as in many datasets as shown in Figure 2.13, it is reasonable to assume that useful information is spanned across their lengths. Especially current OCR technology, which is essential for data preprocessing, only supports extracting spatial information on a single-page basis, without the knowledge of other pages. This behavior poses yet another big challenge in dealing with long documents, which requires a proper method to connect information across pages for all input modalities.

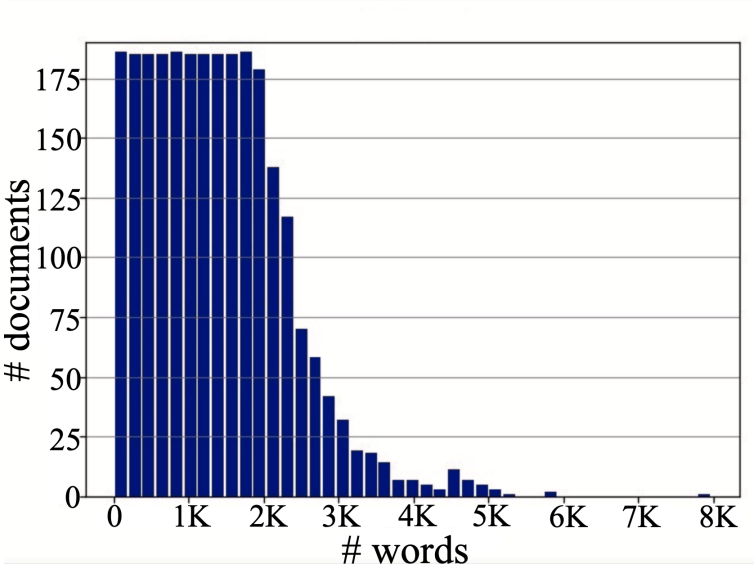


Figure 2.13: Distribution of document length in RVL-CDIP (Harley et al., 2015), a subset of IIT-CDIP used predominantly in the document understanding pretraining tasks. Most of them are longer than 512 words, the limit most current document understanding models accept. We argue that helpful content should span across entire documents and that only accepting 512 words would degrade the performance.

In this work, we discover new approaches to dealing with long document understanding, which addresses the aforementioned challenges. We carefully preprocess OCR data to establish the proper linkages across pages. Then we explore approaches for directly reducing the heavy attention cost while achieving high performance, flexibly using the typical 1D (textual) and/or novel, 2D (spatial) reduced contextual information, without the need of adding more components into the already-heavy transformer (Appalaraju et al., 2021; Nguyen et al., 2021), employing additional pretraining tasks for better representation learning (Huang et al., 2022; Li et al., 2021) or employing complicated new encoding techniques (Hong et al., 2022; Wang et al., 2022). Despite being simple, we show through experiments that both 1D and 2D information can enhance the practicality of transformer-based models while achieving the needed power of handling long documents without introducing any new pretraining tasks other than the popular one: masked language modeling.

Our contributions In summary, we have three following contributions. 1) We newly motivate the simplistic, flexible use of spatial input into self-attention, making it plug-able to transformer-based and other architectures using attention. 2) We are able to tackle the document understanding task with input data up to 4096 words with several attention configurations. 3) Experimental results prove the advantages of our approaches on various long-document datasets in comparison to short models for both 1D and 2D contextual information.

2.3.2 Related Work

Transformer Attention For Long Documents There are several methods that address the quadratic cost of the transformer attention and some of them narrow the focus on long documents for their practicality. Longformer (Beltagy et al., 2020) uses sliding windows to reduce the context, only retrains some sparse global connections. Similarly, ETC (Ainslie et al., 2020) embeds relative positions and adds contrastive predictive encoding. Bigbird (Zaheer et al., 2020) adds a few random connections on top of the sliding windows and sparse global connections and then arranges a long context into a few blocks to reduce the number of intermediate matrix rearrangement and calculation steps. Likewise, less global and more local attentions are learned for higher dimensions to achieve good results (Parmar et al., 2018). Our model similarly uses sliding windows to effectively handle long documents but differs in that it addresses the complication of multimodal, instead of text-only, data and exploits layout input along with the typical text input flexibly and directly into attention and thus enhancing the attention more power and flexibility in dealing with different data types.

While being orthogonal to our work due to difference in approaches, it is worth highlighting some other work contributing to efficient attention for transformer such as substituting softmax with low-rank kernels (Katharopoulos et al., 2020), using Random Fourier features (Peng et al., 2021; Wacker, 2022), extracting random, orthogonal features (Choromanski et al., 2020), or approximating using nested functions (Ma et al., 2021). Some other works try working around the attention and approaching transformer in a different angle such as applying Recurrent Transformer (Dai et al., 2019) with segment reordering objective to pretraining models (Ding et al., 2020), or discarding completely the use of the expensive attention (Bello, 2021) by encoding the local contexts into fixed vectors, preserving the spatial relation while bringing down the computational cost.

Multimodal Document Pretraining Document understanding largely inherits from multimodal pretraining (Chen et al., 2020; Li et al., 2020; Luo et al., 2020) with the successes from LayoutLM (Xu et al., 2020a,b). Docformer (Appalaraju et al., 2021) and StructuralLM (Li et al., 2021) developed the task further by introducing a new two-pronged approach: having new pretraining tasks and suitable changes to the processing or embedding. Similarly, LayoutLMv3 (Huang et al., 2022) introduces two new, additional pretraining tasks on top of masked language modeling (MLM) to enrich the representation learned by the models. Yet another approach is to focus on encoding the spatial information properly with either relative spatial encoding (Hong et al., 2022) or having separate encoding flows for textual and spatial input, then flexibly fusing them (Wang et al., 2022). Unlike all of those approaches, our solution has a different focused motivation that is long documents, only employs MLM as the only pretraining objective, and tackles the attention directly–by efficiently handling the shortened contexts based on textual/spatial information to deal with long contexts–instead of resorting to further embedding and/or encoding all information properly, resulting in a more simple and lightweight solution that can be adapted easily for

any architecture using the attention mechanism.

Finally, Skim-Attention (Nguyen et al., 2021) probably has the most related motivation for long documents, although we have a more memory-efficient, and faster way of handling layout input directly into attention and not from after the embedding like theirs, and consequently support longer input (4096 vs. 2048).

2.3.3 Our Model

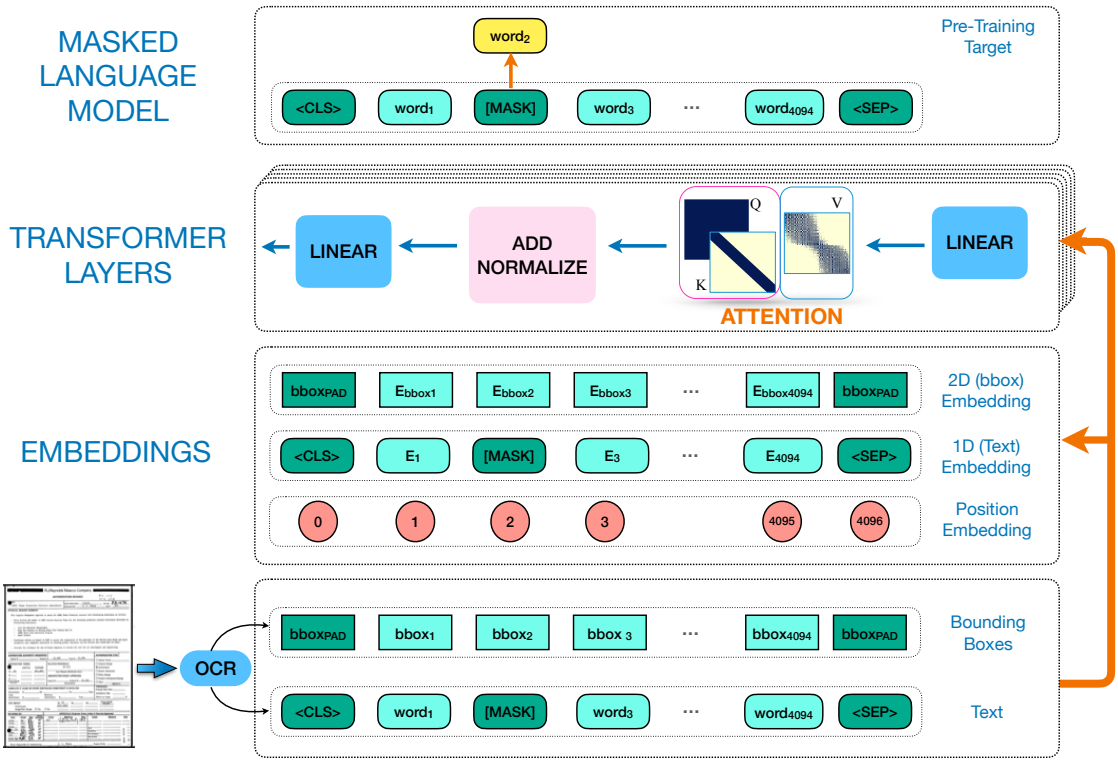


Figure 2.14: Our pre-trained model architecture. Unlike other models for this task, we keep a simple approach by only employing a single MLM pretraining objective and do not employ extra overhead in multimodal embedding or encoding methods. Instead, we tackle the attention module directly and make necessary changes to deal with our focus on long documents, by flexibly using 1D and 2D input.

The structure of this section is as follows. We will first introduce our MLM pretraining model with an emphasis on the novel attention that employs the direct flexible use of either textual (1D) or spatial (2D) information. Next, we explain the post-processing of OCR and its crucial importance in MLM for document intelligence. Then we explain different attention configurations based on 1D and 2D inputs. Finally, we enumerate the models associated with those attention modules.

2.3.3.1 Pretrain Model Architecture

To keep our solution simplistic and easy for studying the effects of each approach being proposed, we only employ Masked Language Model (MLM) architecture as in other document intelligence work, e.g. Xu et al. (2020a,b). However, we discover new attention approaches in MLM to enable its capability of handling long documents. In more detail, different from a typical MLM predominantly used in natural language processing, we have multimodal–instead of text-only–input, which inevitably makes the model heavier and hence cannot deal with long documents without proper changes, as we propose below.

First, we use the sliding-window inspired from Beltagy et al. (2020), given its lightweight and elegance in limiting the context window, making it significantly more memory friendly. Second, we introduce new spatial-based attention masks, in which each context window to a bounding box is determined by calculating its spatial neighbors, instead of the given neighboring words. Likewise, our model not only uses spatial input in the embedding but also in attention directly with preserved spatial correlation. The illustration of our MLM model is shown in Figure 2.14. Additionally, Section 2.3.3.3 will elaborate on the establishment and usage of these new distance masks in comparison with others.

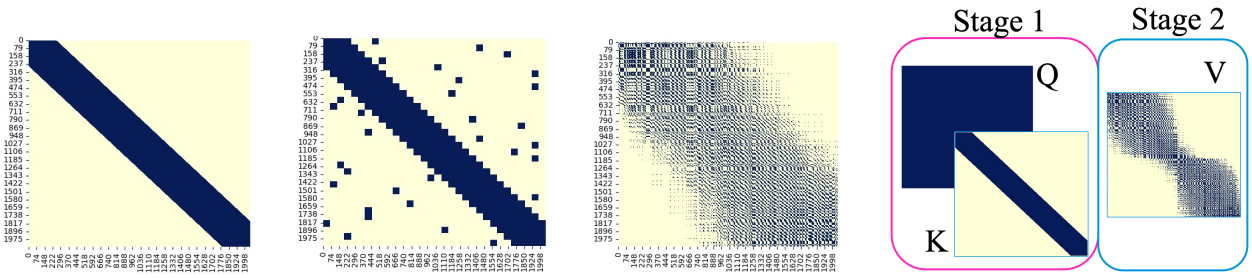
2.3.3.2 Post-OCR Processing

The task of document intelligence relies heavily on the quality of the OCR pre-processing as the first data processing. As a result, how to present the post-OCR data properly to the model is very important, as any mistake in this phase will be compounded later in the model. Especially in the case of long documents, this processing is more crucially important. While long documents have multiple pages, current OCR engines only generate single-page results, without any connections among pages. More current models are "short" models that support up to 512 tokens, and thus typically make use of the very first page's OCR results, discarding the rest of the valuable information. As a result, the further need for post-processing is usually unnecessary in those models.

Unlike those short models, to make our model capable of tackling long documents, we process and normalize the post-OCR data to establish the connections for all input components among the pages. For example, the bounding boxes on page n need adjusting the coordinates to include the previous $n-1$ pages.

2.3.3.3 Different Attention Masks

We are motivated by the fact that in rich documents with multimodal contents, the relationship of words not only follows the consecutive, sequential nature of texts but also in the boxes or sections organized in many complicated forms, in which spatial input offers essential information in addition to text. Furthermore, we argue that in dealing with long documents, we should not put



(a) SW (b) SW+RAND (c) DISTANCE (d) DISTANCE+SW

Figure 2.15: Visualization of our models' different types of attention mask for real samples from RVL-CDIP dataset (Harley et al., 2015) with limit length of 2048 and context size 512 (for both textual and spatial cases). Fig 2.15a is sliding window (SW), Fig 2.15b is sliding window in blocks with 1-per-block random blocks (SW+RAND), Fig 2.15c is a spatial-based distance mask, and Fig 2.15d is the combination of sliding window and distance modes. *Legend*: Attention mask may only have values of 0 and 1, which are represented as the light-yellow background and dark-blue foreground colors, respectively.

extra overhead on the already-heavy transformer-based models in both computation and memory perspectives. As a result, we employ neither additional embedding techniques nor complicated encoding or fusing methods as in many other approaches (see Section 2.3.2 for more information), and instead focus on making the attention, the main cost of those models, lightweight and effective by having a shortened yet flexible context information of textual and/or spatial input.

In the following, we begin to describe the original transformer attention mechanism and the different approaches that we propose specifically for long multimodal documents, using 1D and 2D input data.

Original Attention Masks For the original transformer-based architectures (Vaswani et al., 2017), in each of their layers, the attention score is calculated by two main steps, as formulated in Equations (2.27) and (2.28),

$$\text{score}(\mathbf{Q}, \mathbf{K}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right) \tag{2.27}$$

$$\text{attn\_score}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{score}(\mathbf{Q}, \mathbf{K}) \cdot \mathbf{V}, \tag{2.28}$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ stand for the learnable Query, Key, and Value matrices respectively. Given the lengths of these three matrices are all N, which is also the input length, the complexity of each step is $\mathcal{O}(N^2)$.

We usually refer to this attention mechanism as full attention because each single input token attends to all N available tokens including itself, which makes it impractical in terms of both computation and memory in the cases of long documents. As a result, proper changes have to be made as described below in our proposed attention approaches.

Sliding-Window Masks (Figure 2.15a) We use the sliding-window approach as inspired from Beltagy et al. (2020), which limits the context for each token from N down to a smaller M, e.g. $N = 4096, M = 512$, and so the complexity is essentially reduced to $\mathcal{O}(NM)$.

$$\mathbf{K}_w = \text{get\_window}(\mathbf{K}) \tag{2.29}$$

$$\text{score}(\mathbf{Q}, \mathbf{K}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}_w^T}{\sqrt{d_k}}\right) \tag{2.30}$$

$$\mathbf{V}_w = \text{get\_window}(\mathbf{V}) \tag{2.31}$$

$$\text{attn\_score}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{score}(\mathbf{Q}, \mathbf{K}) \cdot \mathbf{V}_w. \tag{2.32}$$

Using that intuition, the calculations are now changed to Equations from 2.29 to 2.32, with the added `get_window` steps in Equations (2.29) and (2.31). This change is simplistic because while significantly reducing the heavy blueprint of the full attention, it retains a consistent pattern of token arrangement for fast implementation [4].

Sliding-Window plus Random Token Masks (Figure 2.15b) On top of sliding windows, we add a few random tokens to establish more connections to the attention, as done similarly by Zaheer et al. (2020). This operation essentially makes changes only to Equations 2.29 and 2.31, and replace them with Equations 2.33 and 2.34, respectively.

$$\mathbf{K}_w = \text{get\_sliding\_and\_rand\_window}(\mathbf{K}) \tag{2.33}$$

$$\mathbf{V}_w = \text{get\_sliding\_and\_rand\_window}(\mathbf{V}). \tag{2.34}$$

In more detail, the sliding-window contexts are enhanced by some random contexts added. While certainly being an extra overhead, the number of those random connections is limited to only a few, maintaining the practicality of the model in the face of long documents [5].

$$\mathbf{K}_w = \text{get\_2D\_spatial\_window}(\mathbf{K}) \tag{2.35}$$

$$\mathbf{V}_w = \text{get\_2D\_spatial\_window}(\mathbf{V}). \tag{2.36}$$

Spatial Distance Masks (Figure 2.15c) Different from previous attention types, the M contextual neighbors of each token are decided by spatial (2D) information instead of textual (1D)

[4] To enable fast calculations in Equations (2.30) and (2.32) with now-changed matrix shapes, one has to extract and chunk the context for all tokens in a way that can exploit fast matrix multiplication (e.g. by using `einsum`).

[5] Due to the introduction of those random tokens, the consistent pattern of sliding windows and hence their fast implementation are largely affected. We divide the original sequence length into blocks (e.g. 512 to 8 equal blocks of length 64), to facilitate grouping and chunking, as well as to lessen the computational steps (have much fewer sliding windows) and only use 3 blocks, by default, for random connections

information. In the final result, however, the spatial attention mask has the same shape as sliding windows (if they both have the same number of contextual neighboring tokens). This process comprises a couple of steps.

First, we calculate the centers of all bounding boxes. Second, we fit the kNN algorithm to the sequence of those points based on L2 distance, resulting in a 2D distance matrix, in which each token now spatially attends to M neighboring tokens. In summary, we replace Equations 2.29 and 2.31 with Equations 2.35 and 2.36. The resulting masks consequently have a non-consecutive neighboring relationship, unlike in the traditional text-based contexts. More illustrations of those distance-based masks for real examples are also shown in Figure 2.16. And because of its importance, in the following, we detail the implementation notes for those new attention masks.

Implementation of Distance Masks In terms of efficient implementation, there are certain considerations to enable the practical use of those newly proposed distance masks, which consume more computation and memory cost compared to the normal sliding window mechanism.

First, identifying spatial neighbors for each token usually takes quadratic time, which is a great deterrent to our solution. So we choose to use `scikit-learn`'s kNN library [6] for its well-regarded efficiency and speed.

Second, "where to create distance masks: in dataset loader or in model computation" is a key problem. We choose to create distance masks in the dataset loader for the following reasons. On one hand, the main obstacle to applying long-document attention methods is that the transformer-based models are inherently heavy. If placing the quadratic computation of those distance masks in the main model phase, the model will be significantly slower (in proportion to document lengths) and the risk of out-of-memory will be much higher (given the limitation of GPU memory). On the other hand, by preemptively computing the distance mask in the dataset loader, e.g. using Pytorch Dataloader [7] and exploiting its data buffering mechanism, the data loading will not be slower by running multiple loader processes simultaneously.

Finally, for the sliding-window attention, we inherit the implementation from Huggingface [8], then implement our distance-based solution on top of it.

2.3.3.4 Pretrain Model Variants

We build out MLM pretraining architecture with various attention mechanisms for long documents as described in Section 2.3.3.3 and compare their performances in several tasks. Since this change is only made directly to the attention, our method can be used off-the-shelf for transformer-based architecture with multimodal input.

[6] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

[7] https://pytorch.org/docs/stable/_modules/torch/utils/data/dataloader.html#DataLoader

[8] https://huggingface.co/transformers/model_doc/longformer.html

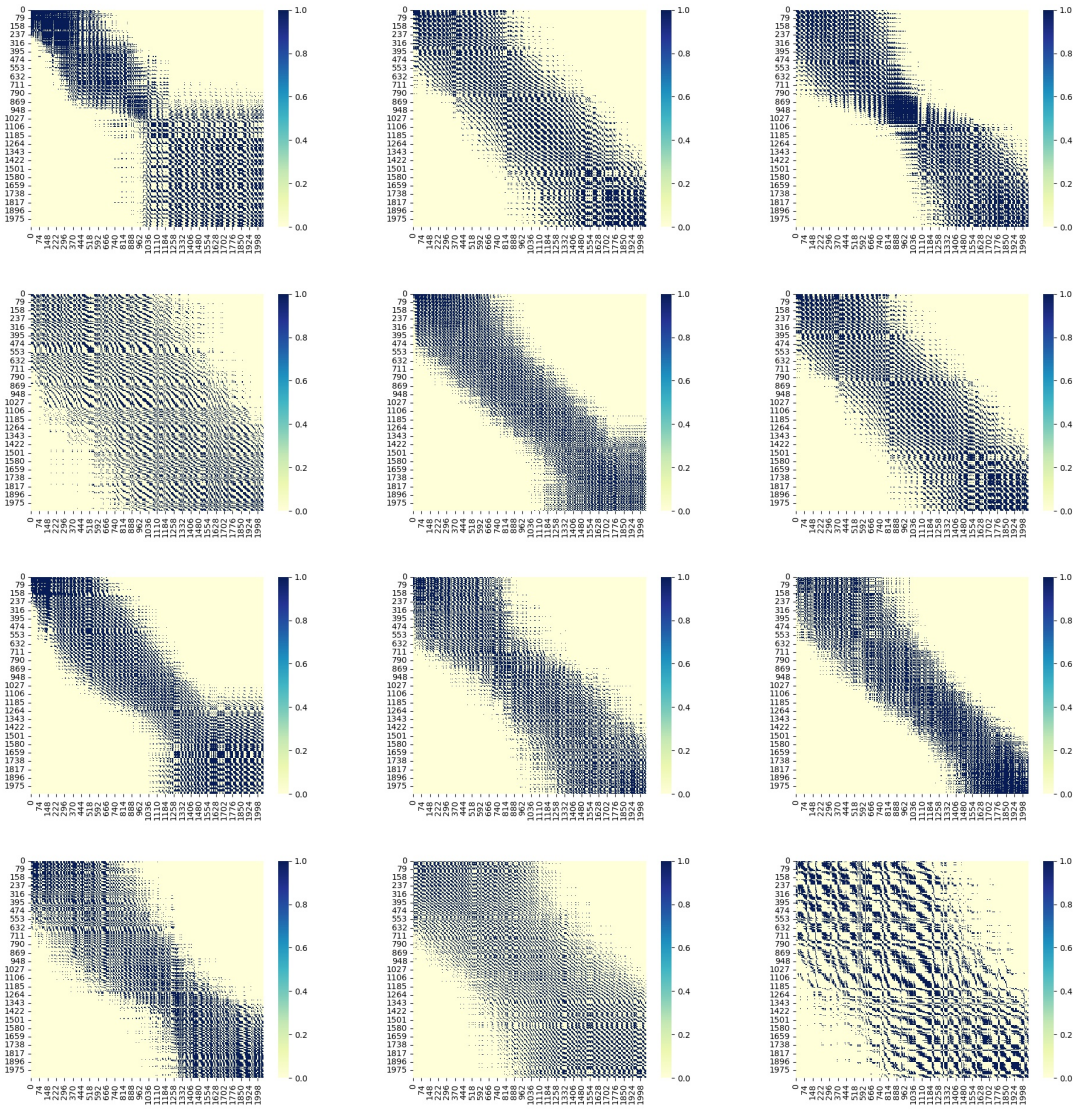


Figure 2.16: More illustrations of distance masks from RVL-CDIP samples with the limit length of 2048 and 512 neighbors each.

SW Model This model directly uses Sliding-Window (SW) masks for attention, which significantly reduces the computation and was shown to be effective for long documents in text-based tasks (Figure 2.15a).

SW+Rand Model This model uses blocked Sliding-Window plus some random blocks on top (Figure 2.15b).

Distance Model This model uses Spatial Distance Masks, with all neighboring contexts being preemptively computed using kNN, and is implemented in the data loading instead of transformer encoding phase, not to slow down the main process (Figure 2.15c).

Distance+SW Model. In this model, we combine the spatial and textual attention masks together in a single attention pass. In detail, it is done via two steps, as shown in Equations (2.29–2.32), with Equation (2.29) now being replaced by Equation (2.35). This is a possible adjustment since these two steps are separated and both preserve the logic and shapes of matrices in their calculation. Our motivation and intuition are to combine the benefits of both textual and spatial information in a single attention pass (Figure 2.15d).

2.3.4 Experiments

In this section, we describe our experimental methodology to evaluate our proposed approach of flexible attention using different contextual input information.

2.3.4.1 Tasks and Datasets

Pretraining We use **IIT-CDIP Test Collection 1.0** [9] dataset for our MLM pretraining task. This is a large-scale dataset with over 6M multi-page documents and around 11M pages in total (each page is stored as a scanned image and is preprocessed by an OCR engine).

Document Classification This document classification task uses **RVL-CDIP** (Harley et al., 2015) dataset, which is a subset of the pretraining dataset IIT-CDIP. It comprises 16 classes and each class equally has 25K grayscale images. All of these 400K images in combination are split into 320K images for training and 40K images each for validation and testing. For more statistics on this dataset, the document length distribution is shown in Figure 2.13.

Sequence Labeling There are two datasets for this task, namely Kleister-NDA and FunSD.

1) FunSD (Guillaume Jaume, 2019)[10] This is a lightweight dataset that has 199 noisy scanned forms, which contain around 31K words and 9.7K entities with 7 given token classes. Although it is not a long-document dataset (all documents have < 512 words), it is a popular dataset used by many document intelligence models and is also useful for ablation studies on how long document models perform on a short document dataset, as we show in Section 2.3.4.6.

2) Kleister-NDA (Graliński et al., 2020; Stanisławek et al., 2021)[11] This dataset has 540 documents in total (254 train, 83 validation, and 203 test) with 2,160 entities annotated and an average of 2,540 words per document. Due to the difficulty in reproducibility with unclear results post-processing, this task is cast similarly to FunSD with 4 classes. Consequently, we report the evaluation results of our models along with all other methods' reproduced outcomes using the same preprocessing steps and metrics, in order to maintain fair comparisons.

[9] https://ir.nist.gov/cdip/
[10] https://guillaumejaume.github.io/FUNSD
[11] https://github.com/applicaai/kleister-nda

Parameter Name	Value
do_lower_case	true
fp16	true
fp16_backend	amp
gradient_accumulation_steps	4
max_seq_length	4096
max_2d_position_embeddings	1024
max_steps	1000000
model_name_or_path	allenai/longformer-base-4096
dataloader_num_workers	64
tasks	mask_lm
optimizer	transformers_AdamW
learning_rate	5e-5
warmup_ratio	0.1
weight_decay	0.01
whole_word_masking	false
add_prefix_space	true
attention_window	512

Table 2.12: Main pretrain hyperpameters on the MLM pretraining task for the ITT-CDIP large-scale dataset. There are 3 variants share this set of parameters that are Ours SW, Ours DISTANCE and Ours DISTANCE+SW models. All of them use the pretrained weights from Longformer-base (Beltagy et al., 2020) model.

2.3.4.2 Pretraining

Pretrain Data Preprocessing To pretrain the models, we retain the same OCR engine for generating and aligning layout and text information from LayoutLM (Xu et al., 2020b). The task is also the same, which is Masked Language Modeling (MLM). To deal with long documents, we have to implement the additional sliding-window, random-block and distance-based masks.

Pre-trained Model Implementation Our solution only makes changes to the attention module, in which uses can choose to use any types of attention masks from the 4 variants illustrated in Figure 2.15.

For the SW and SW+RAND models which are also our new pre-trained models, we implement the layout-related part on top of the original BigBird [12] and Longformer [13] implementations from Huggingface's transformers, respectively. Otherwise the distance-based masks, which are employed in DISTANCE and DISTANCE+SW models, are newly implemented as a pluggable module.

[12] https://huggingface.co/transformers/model_doc/bigbird.html
[13] https://huggingface.co/transformers/model_doc/longformer.html

Training MLM We pre-train the task on the IIT-CDIP datasets, using a single-node multi-GPU mode. Each job was run on a server with 8 V100 Nvidia GPUs, each of which has 32GB memory and fast processors. For text-only models, please refer to LayoutLM's github [14].

For SW model, we use the public pre-trained weights from Lomgformer (Beltagy et al., 2020). Other of our models employ the same set of parameters, except for the pretrained weights, in which SW+Rand model uses the weights from Bigbird (Zaheer et al., 2020) and the last two models having distance masks (Distance and Distance+SW models) use the same pretrained weights as SW model, as demonstrated in Table 2.12.

It is also worth noting that the pretrained weights from Longformer and Bigbird models are useful even for the models using distance masks because those two model families support documents with length 4096, so the position embeddings are helpful. For speed and memory tradeoff, we limit the context for distance masks to only 128 (vs. 512 in textual contexts), without sacrificing much performances, as reported in Section 2.3.4.5.

2.3.4.3 Implementation Notes on Pretraining and Finetuning

Pretraining Notes Although not reported in the main content, we note some lessons learned from the pretraining task. As we observe, the Ours SW model consistently achieves the best results, while consuming the least GPU memory. For the base model, it only consumes about 7 GB GPU memory and Ours Distance+SW that uses sliding-window attention on its half processing also consumes about 9 GB memory. Both models, as a result, can be deployed well on a broad range of GPUs in the market.

Unlike those conveniences, Ours SW+Rand and Ours Distance do not share the same advantages. In fact, they consumes about more than 30GB GPU memory each, limiting their practicality. We hypothesize the main reason for such drawbacks is that they have random, inconsistent patterns, and hence there is no efficient way to take advantage of fast memory-efficient and fast matrix operations.

Finally, although showing promising practical behaviors, all baselines and our models, and almost any transformer-based ones are certainly not lightweight models. And although there are advancements in compressing those heavy models (e.g. (Frankle and Carbin, 2018; Touvron et al., 2021), there seems to be a considerable way to go for making these model run on mobile devices in the near future.

Finetuning Notes As described in the main content, after pretraining, the saved models are the backbone for the respective fine-tuning model types. For that reason, the parameters are mostly shared with their pretraining counter-part models, e.g. Table 2.12 for Ours SW models. Generally, we keep the same optimizer and batch size of 32 (combined across all used parallel GPUs).

For **RVL-CDIP** in the document classification task, we use the `SequenceClassification` model type. On top of the pretrain skeleton, we add a small classifier with 2 fully-connected layers

[14] https://github.com/microsoft/unilm

and a drop-out layer in between. The final output is the single class for the whole sequence/document.

For **FunSD** and **Kleister-NDA** datasets, we instead use the `TokenClassification` model type, which is designed to classify all-document entities. The similar classifier is added to the pretrained skeleton, now with a different usage in which each token/entity is to be classified into 1 of the number of given classes.

What's more, to preprocess these two datasets, we have to ingest all available document tokens. Likewise, with documents longer than the maximum lengths, we need to cut those documents, and recursively treat the overflowing parts in the same way. In terms of implementation, unlike FunSD that is lightweight, we always want to avoid loading the whole dataset into the memory but rather take advantage of the data buffering in feeding to the models. As a result, we pre-process all data first, save them to disks and only load the respective parts when needed.

Additional Information for Kleister-NDA It is worth noting that the evaluation of it is tricky if using the provided official GEval evaluation script (Graliński et al., 2020)[15]. In detail, given the predited tokens, one has to retrieve the associated texts in a group. For example, the beginning of an entity group usually starts with a class beginning with "B-", followed by a series of "I-" tokens. However, there is no guarantee that the prediction will always return a group having this meaningful pattern, let alone many other complicated cases that can happen. Such complications make the post-processing of the prediction– before feeding to GEval–very difficult and importantly, not easily reproducible. In fact, amongst recent papers that report performance on this dataset (e.g. in Appalaraju et al. (2021); Xu et al. (2020a)), there is reference code with which for us to compare.

Consequently, we treat this dataset the same as FunSD, given their similarity in annotation. In addition, because this dataset is larger and much more difficult (due to decoying texts) compared to FunSD, we analyze the train dataset and employ the weighted loss based on the distribution the given labels. As a result, our method is more transparent and reproducible.

2.3.4.4 Baselines

We pretrain our 4 model variants (Figure 2.15) with the MLM objective and then compare them with the following baseline groups:

Text: This group consists of models that only accept text input including BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and other long models including Bigbird (Zaheer et al., 2020) and Longformer (Beltagy et al., 2020) [16].

Text+Layout: This group contains models that accept both text and layout information, including LayoutLM (Xu et al., 2020b) variants.

[15] https://github.com/applicaai/kleister-nda

[16] Our SW and SW+Rand models share the similarity with those last two ones, with the difference of handling multimodal input for document intelligence.

Type	Model	SeqLen	Acc (%) ↑
Text	BERT-base	512	89.81
	RoBERTa-base	512	90.06
	BERT-large	512	89.92
	RoBERTa-large	512	90.11
	Bigbird-base	4096	93.48
	Longformer-base	4096	93.85
	Bigbird-large	4096	93.34
	Longformer-large	4096	93.73
Text+Layout	LayoutLM-base	512	91.88
	LayoutLM-large	512	91.90
	Ours SW	4096	94.50
	Ours SW+RAND	4096	**95.25**
	Ours DISTANCE	4096	94.79
	Ours DISTANCE+SW	4096	94.69

Table 2.13: Classification accuracy for RVL-CDIP. For this long-document dataset, the models capable of using 4096 words uniformly beat other models and layout information helps with the task compared with using Text input. All our long models show their advantages on this long dataset.

2.3.4.5 Results and Discussions

Document Classification As shown in Table 2.13, long models (SeqLen[17] 4096) clearly outperform short ones in both baseline groups, with or without layout information added to the input. Furthermore, all our 4 model variants outperform all the baselines.

This result concurs with our observation that long documents have valuable information spanned across the length. And importantly, our models show advantages of handling long multimodal input, and hence are more practical with real data that are usually longer than 512 tokens.

Sequence Labeling with Kleister-NDA[18] Comparing the "base" versions (separated from their "large" counterparts), Table 2.14 shows that most of our models, which are also the "base" ones, clearly have better scores. Particularly, our SW model is the best performer.

Furthermore, our DISTANCE+SW is not performing equally well. Our hypothesis is that the OCR engine cannot understand the decoying annotation in this dataset, and thus generates spatial results that do not correlate well with the text. Consequently, the combination of textual and spatial information does not result in the benefits of those two.

[17] SeqLen is short for Sequence Length.

[18] The results are from the validation split due to no annotation for the test split provided in the dataset.

Type	Model	SeqLen	F1 ↑
Text	BERT-base	512	47.06
	BERT-large	512	52.66
	Longformer-base	4096	61.78
	Bigbird-base	4096	46.98
Text+Layout	LayoutLM-base	512	55.69
	LayoutLM-large	512	61.95
	Ours SW	4096	**64.06**
	Ours SW+RAND	4096	58.92
	Ours DISTANCE	4096	57.01
	Ours DISTANCE+SW	4096	44.70

Table 2.14: Results on Kleister-NDA. Although this dataset is challenging, long models still show advantages over short ones.

2.3.4.6 Ablation: Long Models on Short Dataset

The purpose of this study is to explore how long models perform on short documents, which also appear in practice, to see whether they can generalize their performance to shorter data.

Table 2.15 shows that on FunSD, we see again that layout information generally helps in the case of multimodal input. However, long models do not perform well compared to short ones, although the gap between the best of ours and the baselines are not very far away (77.1 vs. 79.0). The main reason is that long models essentially have much more parameters than short ones. And not only is FunSD short, it is also very small. As a result, the limited phase of fine-tuning on only 199 samples can hardly tune parameters well for good results. Especially, since all documents are short, most long input to the model is zero padding and thus not enough for contributing for better scores.

Another reason is that long models have their embedding representations trained for the length of 4096 tokens and hence are hard to adapt to 512-token input with just a few fine-tuning steps. As a result, analyzing the data well to design suitable pretraining and fine-tuning models is very important.

The next 2 studies will explore the implications of the newly-added spatial attention masks in our models.

2.3.4.7 Ablation: Different-Length Documents

This study aims to explore how the models work if we do not cut any information from documents (the models take input up to their maximum length limit). Out of 40K test samples in RVL-CDIP, there are 9268 samples with length ≥ 512, 2312 with length ≥ 1024, and only 106 with length ≥ 2048.

Type	Model	SeqLen	F1 ↑
Text	BERT-base	512	60.3
	RoBERTa-base	512	66.5
	BERT-large	512	65.6
	RoBERTa-large	512	70.7
	Bigbird-base	4096	45.8
	Longformer-base	4096	71.4
	Bigbird-large	4096	46.8
	Longformer-large	4096	73.5
Text+Layout	LayoutLM-base	512	78.7
	LayoutLM-large	512	**79.0**
	Ours SW	4096	69.9
	Ours SW+RAND	4096	77.1
	Ours DISTANCE	4096	64.0
	Ours DISTANCE+SW	4096	61.8

Table 2.15: Results on FunSD dataset. As usual, layout information is helpful in boosting performance. However, long models do not perform well compared with short models on this small, short-document dataset.

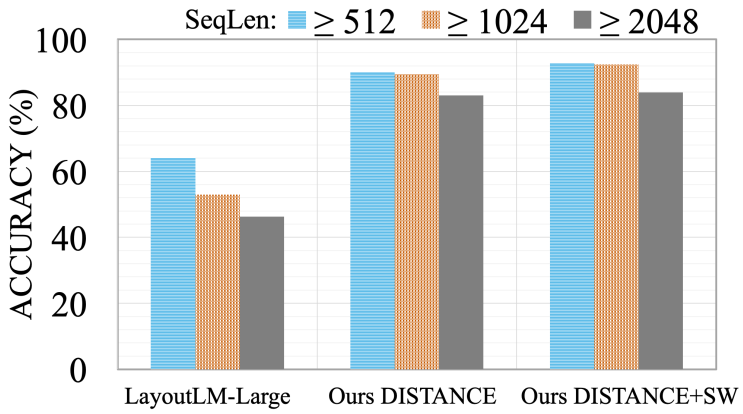


Figure 2.17: RVL-CDIP performance on different document types based on their original lengths (i.e. without purging) with LayoutLM (with the best "large" version) and our spatial models (DISTANCE and DISTANCE+SW). Our models are consistently better.

Figure 2.17 shows the consistent observation that our models are much better than LayoutLM, and yet perform slightly worse as the original document length increases. There could be several possible reasons for this behavior: the models are not well pre-trained and/or fine-tuned, many long documents have lots of confusing parts, or there are many noises in OCR results.

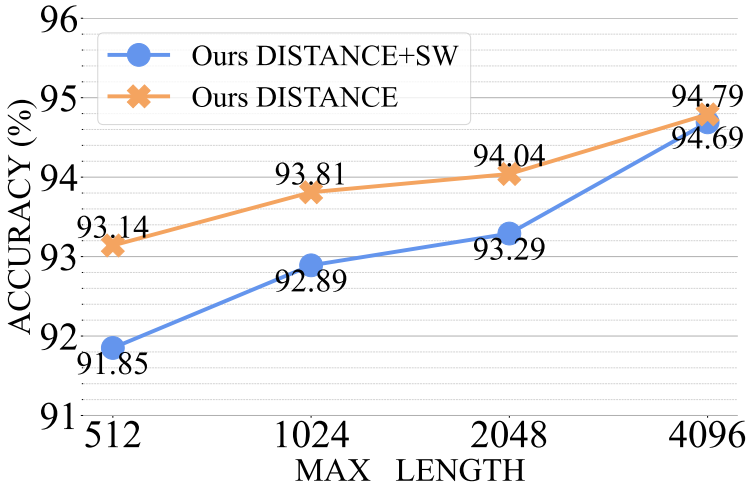


Figure 2.18: RVL-CDIP performance on different maximum lengths using our DISTANCE and DISTANCE+SW models. For each of lengths 512, 1024, 2048, and 4096, the test set contains the same 40K samples. A longer maximum length gives better results.

2.3.4.8 Ablation: Different Max Input Lengths

Given the pre-trained models that can accept input up to 4096 tokens, we finetune them with the input of different maximum lengths, i.e. excess will be purged. As a result, we use all 40K test samples in RVL-CDIP for this study.

As shown in Figure 2.18, our models are better and better as more tokens are absorbed, thus once again confirming our intuition that valuable information is spanned across the length. As a result, if the model capacity permits, we should not limit the capacity to 512 tokens as in most current models in the literature.

2.3.4.9 Further Discussion on Spatial Masks

As seen in the above experimental results, direct usage of 2D layout context information in the transformer attention has some advantages. However, its performance does not match the typical usage of 1D textual information. This might be discouraging at first since introducing spatial masks brings heavier computation compared to textual ones. We hypothesize the drawbacks are due to some objective limitations. First, the kNN suffers some inaccuracy compared with normal (and slow) calculations. Second, the performance of the whole pipeline heavily depends on OCR quality, e.g. in Kleister-NDA with decoy design, OCR results are not well aligned with the text. Consequently, we conjecture that with future development in OCR technologies, the use of spatial masks would be more and more helpful in practice.

2.3.5 Conclusion and Discussion

We propose a versatile solution for long document understanding, in which the shortened context can be used in the form of textual and/or layout input for the attention mechanism in a flexibly pluggable manner. We keep our approach simple by not putting extra overhead on complicated embedding or encoding methods. Despite its simplicity, our solution has shown promising experimental results on document understanding tasks with long, multimodal input. In the future, we will further reduce the memory consumption of models with given multimodal input and speed up the pretraining. Similar to LayoutLM, pretraining usually takes 80 hrs/epoch with 8 V100 GPUs. Thus there are certainly lots of room for improvement to make these models more efficient and practical.

Chapter 3

Scalability of Representation Learning

As mentioned in the introduction (Chapter 1), in the era of big data, one has to take into account the size of the data itself, which is certainly a big factor affecting not only the efficiency of representation learning but importantly its scalability. Consequently, a practical solution is one that can be applied on a large scale so that enterprises of any size can be beneficial from it. That is the second target of this dissertation.

There are many schools for approaching this challenging problem. One is to design efficient distributed, big data-oriented systems that bring a superpower to ingest and process huge data at a high speed. This thesis, however, chooses another approach that is essentially system-agnostic. In more detail, it instead tries to approach the big data problem by tackling directly the learning model itself, by approximating the heavy components of learning, typically in the form of matrices or their related features.

The following sections will be the approximation techniques for Gaussian Processes, implicit matrix trace estimation, and tasked-based Mixture-of-Experts for transformer-based models. It is worth remarking, however, that the techniques that will be introduced in this chapter do not only serve for scalability issues but also efficiency ones simultaneously. For instance, along with helping with scalability, our task-based Mixture-of-Experts architecture concurrently helps the model learn a better representation for multitask data, in which similar tasks' representations are routed to the same task adapters and vice versa. Furthermore, there is no restriction on using the scalability techniques in this chapter for improving the tasks in the previous chapter. For example, one could use our trace estimation work for studying Hessian information of neural networks, in order to design better optimization techniques for training the models in Chapter 2.

3.1 Sparse Spectrum Approximation of Gaussian Processes

This section introduces an approximation method for an important problem in machine learning, Gaussian Processes (GP), which has a lot of applications in practice. Despite that fact, training GP is hard especially given big datasets due to its overly expensive cost. In this work, we introduce a new scalable approximation for GP with provable guarantees which hold simultaneously over its entire parameter space. Our approximation is obtained from an improved sample complexity analysis for sparse spectrum Gaussian processes. In particular, our analysis shows that under a certain data disentangling condition, an SSGP's prediction and model evidence (for training) can well-approximate those of a full GP with low sample complexity. We also develop a new auto-encoding algorithm that finds a latent space to disentangle latent input coordinates into well-separated clusters, which is amenable to our sample complexity analysis. We validate our proposed method on several benchmarks with promising results supporting our theoretical analysis. [1]

3.1.1 Problem and Motivation

GP (Rasmussen and Williams, 2006) is a popular probabilistic kernel method for regression that has found applications across many scientific disciplines. Examples of such applications include meteorological forecastings, such as precipitation and sea-level pressure prediction (Ansell et al., 2006); sensing and monitoring of ocean and freshwater phenomena such as temperature and plankton bloom (Cao et al., 2013; Dolan et al., 2009); traffic flow and mobility demand predictions over urban road networks (Chen et al., 2012, 2013b; Low et al., 2015); flight delay predictions (Hensman et al., 2013; Hoang et al., 2015, 2016); and persistent robotics tasks such as localization and filtering (Xu et al., 2014). The broad applicability of GPs is in part due to their expressive Bayesian non-parametric nature which provides a closed-form prediction (Rasmussen and Williams, 2006) in the form of a Gaussian distribution with formal measures of predictive uncertainty, such as entropy and mutual information criteria (Krause and Guestrin, 2007; Srinivas et al., 2010; Zhang et al., 2016). Such expressiveness makes GPs not only useful as predictive methods but also a go-to representation for active learning applications (Hoang et al., 2014a,b; Krause and Guestrin, 2007; Zhang et al., 2016) or Bayesian optimization (Hoang and Kingsford, 2020; Hoang et al., 2018; Snoek et al., 2012; Zhang et al., 2017) that need to optimize for information gain while collecting training data.

Unfortunately, the expressive power of a GP comes at a cost of poor scalability (i.e., cubic time (Rasmussen and Williams, 2006)) in the size of the training data (see Section 3.1.2.1 below), hence limiting its use to small datasets. This prevents GPs from being applied more broadly to modern settings with increasingly growing volumes of data (Hensman et al., 2013; Hoang et al., 2015, 2016). To sidestep this limitation, a prevalent research trend is to impose sparse structural

[1] Our experimental code is released at https://github.com/hqminh/gp_sketch_nips.

assumptions (Quiñonero-Candela and Rasmussen, 2005; Quiñonero-Candela et al., 2007) on the GP's kernel matrix to reduce its multiplication and inversion cost, which comprises the main bulk of the training and inference complexity. This results in a broad family of sparse Gaussian processes (Hensman et al., 2013; Hoang et al., 2017, 2015; Lázaro-Gredilla et al., 2010; Seeger et al., 2003; Titsias, 2009) that are not only computationally efficient but also amenable to various forms of parallelism (Chen et al., 2013a; Low et al., 2015) and distributed computation (Allamraju and Chowdhary, 2017; Gal et al., 2014; Hoang et al., 2019a, 2016, 2019b), further increasing their efficiency.

Despite such advantages, the sparsification components at the core of these methods are heuristically designed and do not come with provable guarantees that explicitly characterize the interplay between approximation quality and computational complexity. This motivates us to develop a more robust, theoretically-grounded approximation scheme for GPs that is both provable and amenable to the many fast computation schemes mentioned above. More specifically, our contributions include:

1. An analysis of a new approximation scheme that generates a sparse spectrum approximation of a GP with provable bounds on its sample complexity, which practically becomes significantly small when the input data exhibits a certain clustering structure. Furthermore, the impact of the approximation on the resulting training and inference qualities is also formally analyzed (Section 3.1.3.1).

2. A data partitioning algorithm inspired by the above analysis, which learns a cluster embedding that reorients the input distribution while ensuring reconstructability of the original distribution (Section 3.1.3.3). We show that using sparse spectrum Gaussian processes (SSGP) (Hoang et al., 2017; Lázaro-Gredilla et al., 2010) on the embedded space requires fewer samples to achieve the same level of the approximation quality. This also induces a linear feature map which enables efficient training and inference of GPs.

3. An empirical study on benchmarks that demonstrates the efficiency of the proposed method over existing works in terms of its approximation quality versus computational efficiency (Section 3.1.4).

3.1.2 Related Work

3.1.2.1 Gaussian Processes (GPs)

A Gaussian process (Rasmussen and Williams, 2006) defines a probabilistic prior over a random function $g(\mathbf{x})$ defined by the mean function $m(\mathbf{x}) = 0$[2] and kernel function $k(\mathbf{x}, \mathbf{x}')$. These functions induce a marginal Gaussian prior over the evaluations $\mathbf{g} = [g(\mathbf{x}_1) \dots g(\mathbf{x}_n)]^\top$ on an arbitrary finite subset of inputs $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$. Let $\mathbf{x}_*$ be an unseen input whose corresponding output $g_* = g(\mathbf{x}_*)$ we wish to predict. The Gaussian prior over $[g(\mathbf{x}_1) \dots g(\mathbf{x}_n)\ g(\mathbf{x}_*)]^\top$ implies

[2]For simplicity, we assume a zero mean function since we can always re-center the training outputs around 0.

the following conditional distribution:

$$g_* \triangleq g(\mathbf{x}_*) \mid \mathbf{g} \sim \mathbf{N}\Big(\mathbf{k}_*^\top \mathbf{K}^{-1}\mathbf{g},\ k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top \mathbf{K}^{-1}\mathbf{k}_*\Big)\ , \tag{3.1}$$

where $\mathbf{k}_* = [k(\mathbf{x}_*, \mathbf{x}_1) \dots k(\mathbf{x}_*, \mathbf{x}_n)]^\top$ and $\mathbf{K}$ denotes the Gram matrix induced by $k(\mathbf{x}, \mathbf{x}')$ on $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ for which $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. For a noisy observation y perturbed by Gaussian noise such that $y \sim \mathbf{N}(g(\mathbf{x}), \sigma^2)$, Equation 3.1 above can be integrated with $\mathbf{N}(\mathbf{g}, \sigma^2\mathbf{I})$ to yield:

$$g_* \triangleq g(\mathbf{x}_*) \mid \mathbf{y} \sim \mathbf{N}\Big(\mathbf{k}_*^\top (\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y},\ k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}_*\Big)\ , \tag{3.2}$$

which explicitly forms the predictive distribution of a Gaussian process. The defining parameter $\mathbf{\Theta}$ of $k(\mathbf{x}, \mathbf{x}')$ (see Section 3.1.2.2) is crucial to the predictive performance and needs to be optimized via minimizing the negative log-likelihood of $\mathbf{y}$:

$$\ell(\mathbf{\Theta}) = \frac{1}{2}\log\Big|\mathbf{K}_{\mathbf{\Theta}} + \sigma^2\mathbf{I}\Big| + \frac{1}{2}\mathbf{y}^\top\Big(\mathbf{K}_{\mathbf{\Theta}} + \sigma^2\mathbf{I}\Big)^{-1}\mathbf{y}\ , \tag{3.3}$$

where we now use the subscript $\mathbf{\Theta}$ to indicate that $\mathbf{K}$ is a function of $\mathbf{\Theta}$. In practice, both training $\mathbf{\Theta}$ and prediction incur $\mathbf{O}(n^3)$ processing cost, which prevents direct use of Gaussian processes on large datasets that might contain more than tens of thousands of training inputs.

3.1.2.2 Sparse Spectrum Gaussian Processes

Sparse spectrum Gaussian processes (SSGPs) (Gal and Turner, 2015; Hoang et al., 2017; Lázaro-Gredilla et al., 2010) exploit Theorem 3.1.1 below to re-express the Gaussian kernel $k(\mathbf{x}, \mathbf{x}') \triangleq \sigma^2 \exp(-0.5\,(\mathbf{x} - \mathbf{x}')^\top\,\mathbf{\Theta}^{-1}\,(\mathbf{x} - \mathbf{x}'))$ (where $\mathbf{\Theta} \triangleq \text{diag}[\theta_1^2 \dots \theta_d^2]$) as an integration over a spectrum of cosine functions such that the integrating distribution (over the frequencies that parameterize these functions) is a multivariate Gaussian.

Theorem 3.1.1 (Bochner Theorem). *Let $k(\mathbf{x}, \mathbf{x}')$ denote a Gaussian kernel defined above and let $q(\mathbf{r}) \sim \mathbf{N}(\mathbf{0}, (4\pi^2\mathbf{\Theta})^{-1})$. It follows that:*

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}_{\mathbf{r}\sim q(\mathbf{r})}\Big[\sigma^2 \cos\Big(2\pi\mathbf{r}^\top(\mathbf{x} - \mathbf{x}')\Big)\Big]\ , \tag{3.4}$$

where $\mathbf{r}$ is a d-dimensional random variable that parameterizes $\cos(2\pi\mathbf{r}^\top(\mathbf{x} - \mathbf{x}'))$. In practice, $\mathbf{r}$ is often referred to as the spectral frequency.

This allows us to approximate the original Gram matrix $\mathbf{K}$ with a low-rank matrix $\mathbf{K}'$ constructed by a linear kernel $\mathbf{K}'(\mathbf{x}, \mathbf{x}') = \mathbf{\Phi}(\mathbf{x})^\top\mathbf{\Phi}(\mathbf{x})$ with feature map $\mathbf{\Phi}(\mathbf{x}) = [\phi_1(\mathbf{x}) \dots \phi_{2m}(\mathbf{x})]^\top$ comprising $2m$ basis trigonometric functions (Hoang et al., 2017). Each pair of odd- and even-index basis functions $\phi_{2i-1}(\mathbf{x}) = \cos(2\pi\mathbf{r}_i^\top\mathbf{x})$ and $\phi_{2i}(\mathbf{x}) = \sin(2\pi\mathbf{r}_i^\top\mathbf{x})$ is parameterized by the same sample of spectral parameter $\mathbf{r}_i \sim q(\mathbf{r})$. For efficient computation, m is often selected to be

significantly smaller than n (i.e., the number of training examples). However, to guarantee that $\|\mathbf{K} - \mathbf{K}'\|_2 \leq \lambda$ with probability at least $1 - \delta$, m needs to be as large as $\mathbf{O}(n^2/\lambda^2 \log(n/\delta))$ (Mohri et al., 2018)[3], which makes the total prediction complexity much worse than that of a full GP.

Alternatively, one can use kernel sketching methods (Avron et al., 2017; Chamakh et al., 2020; Musco and Musco, 2016; Rahimi and Recht, 2007; Sriperumbudur and Szabó, 2015) to generate feature maps that scale more favorably with the effective dimension of the kernel matrix, which empirically tends to be on the order of $\mathbf{O}(\log n)$. However, the pitfall of these methods is that without knowing the exact parameter configuration $\boldsymbol{\Theta}$ that underlies the data, they cannot sample from the true probability $q(\mathbf{r})$, which is necessary for their analyses. As such, existing random maps (Avron et al., 2017; Rahimi and Recht, 2007) that were generated based on this spectral construction often depend on a parameter initialization, and their approximation quality is only guaranteed for that particular parameter setting instead of uniformly over the entire parameter space. This motivates us to revisit the sample complexity of SSGP from a setting that specifically searches for a reorientation of the input distribution such that the reoriented data exhibits a disentangled cluster structure. Such disentanglement provides a more sample-efficient bound as we show in our analysis in Section 3.1.3.1 below.

3.1.3 Provable Approximation of SSGPs with Improved Sample Complexity

We first show how a sparse spectrum Gaussian process (SSGP) (Lázaro-Gredilla et al., 2010) can be approximated well with a provably low sample complexity. This is achieved by revisiting its sample complexity which, unlike prior work (Avron et al., 2017; Mohri et al., 2018; Rahimi and Recht, 2007), explicitly characterizes and accounts for a certain set of data disentanglement conditions. Importantly, our new analysis (Section 3.1.3.1) yields practical bounds on both an SSGP's prediction and model evidence (Section 3.1.3.2) that hold with high probability uniformly over the entire parameter space[4]. Furthermore, our analysis also inspires an encoding algorithm that finds a latent space to disentangle the encoded coordinates of data into well-separated clusters on which a sparse spectrum GP can approximate a GP provably well (Section 3.1.3.3). Our experiments show that such a latent space can be found for several real-world datasets (Section 3.1.4).

[3] See Theorem 6.28 in Chapter 6 of (Mohri et al., 2018).

[4] In contrast, existing literature often generates bounds on either an SSGP's prediction or its model evidence (for training) for a single parameter configuration, which makes such an analysis only heuristic.

3.1.3.1 Practically Improved Sample Complexity for Sparse Spectrum Gaussian Processes

This section derives a new data-oriented feature map to approximate a Gaussian process parameterized with a Gaussian kernel. Unlike existing work which assumes knowledge of the true kernel parameters (Avron et al., 2017; Musco and Musco, 2016; Rahimi and Recht, 2007), our derivation remains oblivious to such parameters and therefore holds universally over their entire candidate space. We assume that the GP prior of interest is of the form $g(\mathbf{x}) \sim \text{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ where $k(\mathbf{x}, \mathbf{x}')$ represents its Gaussian kernel in Section 3.1.2.2.

We give our analysis in three parts: (1) the spectral sampling scheme and a notion of approximation loss; (2) a set of practical data conditions which can be either observed from a raw data distribution or approximately imposed on the data via a certain embedding; (3) a theoretical analysis that delivers our key result that establishes an improved sample complexity when our data conditions are met.

Spectral Sampling Scheme and Spectral Loss

We show that $g(\mathbf{x})$ can be approximated by $g'(\mathbf{x}) = \sum_{i=1}^{p} g_i(\mathbf{x})$ with provable data-oriented guarantees where $g_i(\mathbf{x}) \sim \text{GP}(0, (1/\sqrt{p})k_i(\mathbf{x}, \mathbf{x}'))$. To achieve this, we first establish in Lemma 3.1.2 that the induced Gram matrix $\mathbf{K}$ of $k(\mathbf{x}, \mathbf{x}')$ on any dataset can be represented as an expectation over a space of induced Gram matrices $\{\mathbf{K}_i\}_{i=1}^{p}$ produced by a corresponding space of random kernels $\{k_i(\mathbf{x}, \mathbf{x}')\}_{i=1}^{p}$.

Lemma 3.1.2. *Let $k(\mathbf{x}, \mathbf{x}')$ and $\mathbf{K}$ denote a Gaussian kernel parameterized by $\boldsymbol{\Theta}$ (Section 3.1.2.2) and its induced Gram matrix on an arbitrary set of training inputs, respectively. There exists a space $\mathcal{K}$ of random kernels $\kappa(\mathbf{x}, \mathbf{x}')$ and a $\boldsymbol{\Theta}$-independent distribution ρ over $\mathcal{K}$ for which $\mathbf{K} = \mathbb{E}_{\kappa}[\mathbf{K}_{\kappa}]$ where $\mathbf{K}_{\kappa}$ denotes the induced Gram matrix of κ on the same set of training inputs.*

This follows directly from Theorem 3.1.1 above which states that $k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[\sigma^2 \cos(2\pi \mathbf{r}^\top (\mathbf{x} - \mathbf{x}'))]$ where $\mathbf{r} \sim \mathbf{N}(\mathbf{0}, (4\pi^2 \boldsymbol{\Theta})^{-1})$. We can choose $\kappa(\mathbf{x}, \mathbf{x}'; \epsilon) = \cos(\epsilon^\top \boldsymbol{\Theta}^{-0.5}(\mathbf{x} - \mathbf{x}'))$ where $\epsilon \sim \mathbf{N}(\mathbf{0}, \mathbf{I})$ which implies $\mathbf{k}(\mathbf{x}, \mathbf{x}') = \mathbb{E}_{\epsilon}[\kappa(\mathbf{x}, \mathbf{x}'; \epsilon)]$. Thus, $\mathbf{K} = \mathbf{E}_{\epsilon}[\mathbf{K}_{\epsilon}]$ where the $\boldsymbol{\Theta}$-independent parameter ϵ indexes κ and $\mathbf{K}_{\epsilon}$ is the induced Gram matrix of κ. Leveraging the result of Lemma 3.1.2, a naïve analysis (Mohri et al., 2018) using worst-case concentration bounds to derive a conservative estimate for a sufficient number of samples would require a prohibitively expensive sample complexity of $\mathbf{O}(n^2 \log n)$.

Such analyses, however, often ignore the input distribution, which can be used to sample more selectively, thereby significantly reducing the sample complexity. This is demonstrated below in Theorem 2 which shows that when the input distribution exhibits a certain degree of compactness and separation (as defined in Conditions 1-3), we only require $\mathbf{O}((\log^2 n/\lambda^2) \log\log(n/\delta))$ sampled kernels $\{\kappa_i\}_{i=1}^{p}$ indexed by $\{\epsilon_i\}_{i=1}^{p}$ to produce an average Gram matrix $\mathbf{K}' = \frac{1}{p}\sum_{i=1}^{p} \mathbf{K}_{\epsilon_i}$ that is sufficiently close to $\mathbf{K}$ in spectral norm (see Definition 3.1.3) with probability at least $1 - \delta$.

Definition 3.1.3 (Spectral Closeness). *Given $\lambda > 0$, the symmetric matrices $\mathbf{K}$ and $\mathbf{K}'$ are λ-close if $\|\mathbf{K} - \mathbf{K}'\|_2 \leq \lambda$ where $\|\mathbf{K} - \mathbf{K}'\|_2 = \lambda_{\max}(\mathbf{K} - \mathbf{K}')$ denotes the largest eigenvalue of $\mathbf{K} - \mathbf{K}'$.*

Thus, parameterizing the GP prior with $\mathbf{K}'$ instead of $\mathbf{K}$ allows us to derive an upper bound on the expected difference between their induced model evidence (for learning kernel parameters) and prediction losses (for testing) with respect to the same parameter setup (Theorem 3). Theorem 3 importantly exploits the fact that the bound in Theorem 2 holds universally over the entire space of parameters, which allows us to bound the prediction difference between the original and approximated GPs with respect to their own optimized parameters (that are not necessarily the same).

Practical Conditions on Data Distributions

We now outline key practical data conditions, which can be satisfied approximately via an encoding algorithm that transforms the input data into a latent space where such conditions are met. These conditions are necessary for deriving a practically improved sample complexity in Section 3.1.3.1.

Condition 1. For each parameter configuration $\mathbf{\Theta} = \text{diag}[\theta_1^2, \ldots, \theta_d^2]$, there exists a mixture distribution $\mathcal{M}(\mathbf{x}; \gamma = (\gamma_1, \ldots, \gamma_b), \pi = (\pi_1, \ldots, \pi_b), \mathbf{c} = (\mathbf{c}_1, \ldots, \mathbf{c}_b))$ with at most $b = \mathbf{O}(\log n)$ Gaussian components $\mathbf{N}(\mathbf{x}; \mathbf{c}_i, \gamma_i^2 \mathbf{\Theta}^{-1})$ over the data space with the mixing weights $\pi_i \propto 2^{\frac{i}{2}}$ and variances $\gamma_i = \mathbf{O}(\frac{1}{\sqrt{d}})$ that *generate* the observed data in d-dimensional space.

Condition 2. The i^{th} Gaussian component as defined in Condition **1** above was used to generate $2^{\frac{i}{2}}$ data points of the observed dataset. This can be substantiated easily with high probability given the above setup in Condition **1** that assigns selection probability $\pi_i \propto 2^{\frac{i}{2}}$ to the i^{th}-component.

Condition 3. For each parameter configuration $\mathbf{\Theta} = \text{diag}[\theta_1^2, \ldots, \theta_d^2]$, the mixture distribution of data in Condition **1** has sufficiently separated cluster centers. That is, for all $i \neq j$:

$$\|\mathbf{\Theta}^{-1/2}(\mathbf{c}_i - \mathbf{c}_j)\|_2^2 \; > \; \frac{3}{2} \log\left(\frac{2^a}{2^a - 1}\right) \qquad \text{where} \qquad a \; = \; \frac{1}{\log 2} \log\left(\frac{n^4}{n^4 - \lambda^4}\right). \tag{3.5}$$

These conditions impose that the observed data can be separated into a number of clusters with exponentially growing sizes and concentration (see the small variances defined in Condition **1** and the imposed sizes of Condition **2**). Intuitively, this means data points that belong to clusters with high concentration are responsible for kernel entries with high values whereas those in clusters with low concentration generate entries with low values. This is easy to see since high concentration reduces the distance between data points, thus increasing their kernel values and vice versa.

Furthermore, as imposed by Condition **2**, clusters with high concentration also have denser populations and induce kernel entries with high values. In addition, Condition **3** requires that clusters are well-separated, which implies that a large number of kernel entries are small and therefore can be approximated cheaply. Together, these conditions form the foundations of our

reduced complexity analysis for SSGP in Theorem 3.1.4. Interestingly, we show that such conditions also inspire the development of a probabilistic algorithm that finds an encoding of the input that (approximately) satisfies these conditions while preserving the statistical properties of the input (Section 3.1.3.3). This results in an improved sample complexity for SSGPs in practice (see Section 3.1.3.1).

Main Results

To understand the intuition of why an improved sample complexity can be obtained, we note that when data is partitioned in clusters with different concentrations and sizes, the kernel entries are also partitioned into multiple value bands with a narrow width (i.e., low variance). Exploiting this, we can calibrate a significantly lower sample complexity for each band using concentration inequalities that improve with lower variance (Chernoff, 1952; Hoeffding, 1963).

Then, to combine these in-band sample complexities efficiently, we further exploit the data conditions in Section 3.1.3.1 to show that statistically, value bands with smaller widths also tend to be populated more densely[5]. This allows us to aggregate these in-band sample costs into an overall sample complexity with low cost. In practice, this also inspires an embedding algorithm (Section 3.1.3.3) that transforms the data in such a way that the distribution of their induced kernel entries will be denser in narrower bands, which is advantageous in our analysis.

Formally, let $\mathcal{C}$ be the set of all kernel entries indexed by (u, v) in the Gram matrix $\mathbf{K}$ such that $\mathbf{x}_u$ and $\mathbf{x}_v$ belong to the same cluster and $\mathcal{C}'$ be its complement. Also, let $\mathcal{C}$ be partitioned into b value-bands $\kappa_i = \{(u, v) \in \mathcal{C} \mid 1 - \mathbf{O}(2^{1-i}) \leq \mathbf{K}_{uv}^4 \leq 1 - \mathbf{O}(2^{-i})\}$ for $i \in [1 \dots b]$ and let $\kappa_0 = \{(u, v) \in \mathcal{C} \mid \mathbf{K}_{uv}^4 \geq 1 - \mathbf{O}(2^{-b})\}$ be a band that is only populated by very large kernel entries. Theorem 3.1.4 below shows that we can construct a λ-spectral approximation of $\mathbf{K}$ with arbitrarily high probability and low sample complexity.

Theorem 3.1.4. *For any $1 \geq \delta \geq \mathbf{O}(\exp(b - \sqrt{d}))$, if the training data has n data points and satisfies Conditions **1-3** above with respect to λ, then with probability at least $1 - 2\delta$, the approximation $\mathbf{K}' = (1/p) \sum_{i=1}^{p} \mathbf{K}_{\epsilon_i}$ where $\epsilon_i \sim \mathbf{N}(\mathbf{0}, \mathbf{I})$ is λ- spectral close to $\mathbf{K}$.*

Proof Sketch of Theorem 3.1.4

First, with a proper choice of a clustering partition, the cross-cluster entries in $\mathbf{K}$ are guaranteed to be sufficiently small so as to be well-approximated by zero. We can then show with high probability that any kernel entry that corresponds to a pair of unique data points from the same cluster can be well-approximated with a sample complexity that scales favorably with the cluster's variance. In particular, we show that kernel values induced by data points generated by lower-variance clusters (see Condition **1**) will have smaller approximation variances than those generated by data from higher-variance clusters and therefore require fewer samples to produce the same level of approximation.

[5]The intuition here is that kernel entries in narrower bands are cheaper (in term of sample cost) to approximate.

Second, for certain configurations of mixture weights, Condition **2** asserts that the number of data points from each cluster is inversely proportional to the cluster variance, which implies that a small sample complexity is enough to approximate the majority of kernel entries. More specifically, Lemma 3.1.6 shows that when the input points are distributed into clusters with certain choices of variances $\{\gamma_i\}_{i=1}^{b}$ and at an inversely proportional ratio $\mathbf{O}(\gamma_i^{-1})$, then with high probability, over all clusters, the kernel entries (excluding those on the diagonal) associated with pairs in the i-th cluster belong to their corresponding band κ_i.

Lemma 3.1.9 shows that for $p = \mathbf{O}(\log^2 n/\lambda^2 \log(\log n/\delta))$, with probability $1 - \delta/b$, the total approximation error of all kernel entries in the $\mathcal{C}_i$ will be at most $\lambda^2/4b$, which implies with probability $1 - \delta$, the total approximation cost for items in $\mathcal{C}$ is at most $\lambda^2/4$. Next, Lemma 3.1.5 establishes that with the above data distribution, $\mathcal{C}$ accounts for $n^2/4$ entries while $\mathcal{C}'$ accounts for $3n^2/4$ entries, which needs to be approximated with error at most $3\lambda^2/4$.

Finally, Lemma 3.1.7 shows that when the clusters are sufficiently well-separated (see Condition **3**), any kernel value corresponding to an arbitrary data pair with points belonging to different clusters is guaranteed to be smaller than λ^2/n^2, which then guarantees a total error of at most $3\lambda^2/4$ when they are uniformly approximated with zero. Putting these together yields a total error of λ^2 with probability $1 - 2\delta$, which implies $\mathbf{K}$ and $\mathbf{K}'$ are λ-spectrally close since $\|\mathbf{K} - \mathbf{K}'\|_2 \leq \|\mathbf{K} - \mathbf{K}'\|_{\mathrm{F}} \leq \lambda$. The detailed proof is as follows.

Detailed Proof of Theorem 3.1.4

Let $\Delta(\mathbf{x}_u, \mathbf{x}_v) \triangleq |\mathbf{K}(\mathbf{x}_u, \mathbf{x}_v) - \mathbf{K}'(\mathbf{x}_u, \mathbf{x}_v)|$ where $\mathbf{K}'(\mathbf{x}_u, \mathbf{x}_v) = (1/p)\sum_{i=1}^{p} \mathbf{K}_{\epsilon_i}(\mathbf{x}_u, \mathbf{x}_v)$, and where $\epsilon_i \sim \mathbf{N}(\mathbf{0}, \mathbf{I})$ as defined in Lemma 3.1.2 above. We will first measure the approximation loss across different value-bands of $\mathbf{K}(\mathbf{x}_u, \mathbf{x}_v)$, thereby deriving tight sample bounds for each band. Combining these with the union bound allows us to establish a much cheaper overall sample complexity as compared to the naïve $\mathbf{O}(n^2 \log n)$ bound.

Lemma 3.1.5. *Suppose the data distribution follows Conditions* **1**-**3** *above. Let* $\mathbf{c}(\mathbf{x}_u)$ *denote the cluster index of each data point* $\mathbf{x}_u$*. Let* $\mathcal{C} \triangleq \{u, v \mid \mathbf{c}(\mathbf{x}_u) = \mathbf{c}(\mathbf{x}_v)\}$ *and* $\mathcal{C}' \triangleq \{u, v \mid \mathbf{c}(\mathbf{x}_u) \neq \mathbf{c}(\mathbf{x}_v)\}$ *denote the sets of in-cluster and out-cluster kernel entries, respectively, where* $|\mathcal{C}| \simeq \frac{n^2}{4}$ *and* $|\mathcal{C}'| \simeq \frac{3n^2}{4}$.

Proof. By Condition **2**, since n data points are scattered across b clusters and each cluster i has $2^{i/2}$ points, it follows that:

$$
\begin{aligned}
n &= \sum_{i=1}^{b} 2^{\frac{i}{2}} = \frac{\sqrt{2^{b+1}} - \sqrt{2}}{\sqrt{2} - 1} \\
\Rightarrow |\mathcal{C}| &= \sum_{i=1}^{b} 2^{i} = 2^{b+1} - 1 = \left(n\left(\sqrt{2} - 1\right) + \sqrt{2}\right)^2 - 1 \simeq \frac{n^2}{4} \\
\Rightarrow |\mathcal{C}'| &= n^2 - |\mathcal{C}| \simeq \frac{3n^2}{4}\,.
\end{aligned} \tag{3.6}
$$

This also implies that $b = \mathbf{O}(\log n)$, which is consistent with Condition **1** above. □

Lemma 3.1.6. *Let $\mathcal{C}_i = \{(u, v) \in \mathcal{C} \mid \mathbf{c}(\mathbf{x}_u) = \mathbf{c}(\mathbf{x}_v) = i\}$ for $i \in [1 \dots b]$. Then with probability at least $1 - \delta$, for $\delta \geq \mathbf{O}(\exp(\log n - \sqrt{d}))$, the following holds for all i and $(u, v) \in \mathcal{C}_i$ for which $u \neq v$:*

$$\left(1 - \frac{1}{2^{a+i-1}}\right)^{\frac{1}{4}} \leq \mathbf{K}(\mathbf{x}_u, \mathbf{x}_v) < \left(1 - \frac{1}{2^{a+i}}\right)^{\frac{1}{4}} \text{ where } a = \frac{1}{\log 2} \log\left(\frac{n^4}{n^4 - \lambda^4}\right) \tag{3.7}$$

Proof. If $\mathbf{x}_u$ and $\mathbf{x}_v$ are both generated from component i of the data distribution as defined in Condition **1**, it follows that $\mathbf{\Theta}^{-1/2}(\mathbf{x}_u - \mathbf{x}_v) \sim \mathbf{N}(\mathbf{c}_i, \gamma_i^2 \mathbf{I})$. Therefore, by standard chi-squared tail bounds, with probability at least $1 - 2e^{-t}$, we have:

$$\|\mathbf{\Theta}^{-1/2}(\mathbf{x}_u - \mathbf{x}_v)\|_2^2 = \gamma_i^2 d \pm \mathbf{O}\left(\gamma_i^2 \sqrt{dt}\right), \tag{3.8}$$

where d is the data dimension. Using this, we can then figure out a setting for γ_i^2 such that $\mathbf{K}(\mathbf{x}_u, \mathbf{x}_v)$ follows the above condition in Equation 3.7. In particular, set

$$\mathcal{L}(i) = \log\left(\frac{2^{a+i}}{2^{a+i} - 1}\right) \quad \text{and} \quad \mathcal{U}(i) = \log\left(\frac{2^{a+i-1}}{2^{a+i-1} - 1}\right). \tag{3.9}$$

We can then choose:

$$\gamma_i^2 = \frac{1}{4d}\Big(\mathcal{U}(i) + \mathcal{L}(i)\Big) \quad \text{and} \quad t = \sqrt{d}\left(\frac{\mathcal{U}(i) - \mathcal{L}(i)}{\mathcal{U}(i) + \mathcal{L}(i)}\right) \simeq \mathbf{O}\left(\sqrt{d}\right), \tag{3.10}$$

so that by plugging these choices in Equation 3.8 above, we have with probability at least $1 - 2e^{-t}$:

$$\begin{aligned} \|\mathbf{\Theta}^{-1/2}(\mathbf{x}_u - \mathbf{x}_v)\|_2^2 &\in \left[\frac{1}{2}\log\left(\frac{2^{a+i}}{2^{a+i} - 1}\right), \frac{1}{2}\log\left(\frac{2^{a+i-1}}{2^{a+i-1} - 1}\right)\right] \\ \Rightarrow \mathbf{K}(\mathbf{x}_u, \mathbf{x}_v) &\in \left[\left(1 - \frac{1}{2^{a+i-1}}\right)^{\frac{1}{4}}, \left(1 - \frac{1}{2^{a+i}}\right)^{\frac{1}{4}}\right]. \end{aligned} \tag{3.11}$$

Now, note that for any δ for which $\delta \geq \mathbf{O}\left(4^b e^{-\sqrt{d}}\right) \geq \mathbf{O}\left(4^i e^{-\sqrt{d}}\right)$ $\forall i \leq b$, we have $\delta/4^i \geq 2e^{-t}$ since $t \simeq \mathbf{O}(\sqrt{d})$. This also means $\delta \geq \mathbf{O}(\exp(\log n - \sqrt{d}))$ since $b = \mathbf{O}(\log n)$.

That is, Equation 3.11 and hence, Equation 3.7, hold with probability at least $1 - 2e^{-t} \geq 1 - \delta/4^i$ for each entry in $\mathcal{C}_i$. For each cluster i, even though there are up to 2^i kernel entries, by the triangle inequality it is easy to see that we only need to apply a union bound over at most $2^{i/2}$ (carefully selected) entries (excluding the entries on the diagonal) to meet Equation 3.7 with probability at least $1 - 2^i(\delta/4^i) = 1 - \delta/2^i$.

Subsequently, applying a union bound over all clusters gives us that with probability at least $1 - \delta \sum_{i=1}^{b} 1/2^i \geq 1 - \delta$, all kernel entries within the i-th cluster satisfy Equation 3.7 simultaneously for $1 \leq i \leq b$. □

Lemma 3.1.7. *For all* $(u,v) \in \mathcal{C}' \triangleq \{u,v \mid \mathbf{c}(\mathbf{x}_u) \neq \mathbf{c}(\mathbf{x}_v)\}$, *we have* $\mathbf{K}(\mathbf{x}_u,\mathbf{x}_v) < \left(1-\frac{1}{2^a}\right)^{\frac{1}{4}}$ *where* $a = \frac{1}{\log 2}\log\left(\frac{n^4}{n^4-\lambda^4}\right)$ *as defined in Lemma 3.1.6 above.*

Proof. For any (u,v) for which $\mathbf{c}(\mathbf{x}_u) = i$ and $\mathbf{c}(\mathbf{x}_v) = j$ and $i \neq j$, we have:

$$\begin{aligned}
\|\mathbf{\Theta}^{-1/2}(\mathbf{x}_u - \mathbf{x}_v)\|_2^2 &\geq \|\mathbf{\Theta}^{-1/2}(\mathbf{c}_i - \mathbf{c}_j)\|_2^2 - \|\mathbf{\Theta}^{-1/2}(\mathbf{x}_u - \mathbf{c}_i)\|_2^2 - \|\mathbf{\Theta}^{-1/2}(\mathbf{x}_v - \mathbf{c}_j)\|_2^2 \\
&\geq \|\mathbf{\Theta}^{-1/2}(\mathbf{c}_i - \mathbf{c}_j)\|_2^2 - \frac{1}{2}\log\left(\frac{2^{a+i}}{2^{a+i}-1}\right) - \frac{1}{2}\log\left(\frac{2^{a+j}}{2^{a+j}-1}\right) \\
&\geq \|\mathbf{\Theta}^{-1/2}(\mathbf{c}_i - \mathbf{c}_j)\|_2^2 - \log\left(\frac{2^a}{2^a-1}\right) \\
\Rightarrow \mathbf{K}(\mathbf{x}_u,\mathbf{x}_v) &= \exp\left(-\frac{1}{2}\|\mathbf{\Theta}^{-1/2}(\mathbf{x}_u - \mathbf{x}_v)\|_2^2\right) \\
&\leq \exp\left(-\frac{1}{2}\cdot\|\mathbf{\Theta}^{-1/2}(\mathbf{c}_i - \mathbf{c}_j)\|_2^2 + \frac{1}{2}\log\left(\frac{2^a}{2^a-1}\right)\right) < \left(1-\frac{1}{2^a}\right)^{\frac{1}{4}}
\end{aligned}$$

since for all (i,j), by Condition **3**:

$$\|\mathbf{\Theta}^{-1/2}(\mathbf{c}_i - \mathbf{c}_j)\|_2^2 > \frac{3}{2}\log\left(\frac{2^a}{2^a-1}\right). \tag{3.12}$$

This completes our proof for the stated result of Lemma 3.1.7. □

Corollary 3.1.8. *With probability at least* $1-\delta$, *there are exactly* n *entries that are greater than* $1-2^{-(a+b)}$ *where* $a = \frac{1}{\log 2}\log\left(\frac{n^4}{n^4-\lambda^4}\right)$. *These are the diagonal entries* $\mathbf{K}(\mathbf{x}_u,\mathbf{x}_u)$ *with* $1 \leq u \leq n$.

Proof. Lemma 3.1.6 asserts that with probability $1-\delta$, all kernel entries $\mathbf{K}(\mathbf{x}_u,\mathbf{x}_v)$, where $\mathbf{c}(\mathbf{x}_u) = \mathbf{c}(\mathbf{x}_v) = i$, belong to their respective band $\kappa_i = \{(u,v) \mid 1-1/2^{a+i-1} \leq \mathbf{K}^4(\mathbf{x}_u,\mathbf{x}_v) \leq 1-1/2^{a+i}\}$. When this happens, all in-cluster entries (except the diagonal entries) will have values between $1-1/2^a$ and $1-1/2^{a+b}$ (since there are b bands) and as such, off-cluster entries will either be smaller than $1-1/2^a$ or larger than $1-1/2^{a+b}$. But then Lemma 3.1.7 further guarantees that all off-cluster entries are smaller than $1-1/2^a$, following Condition **3**. Thus, it follows that the only entries that are larger than $1-1/2^{a+b}$ are the diagonal items and there are exactly n of them. □

Lemma 3.1.9. *Let* $\kappa_i = \left\{(u,v) \mid 1-1/2^{a+i-1} \leq \mathbf{K}^4(\mathbf{x}_u,\mathbf{x}_v) < 1-1/2^{a+i}\right\}$. *It follows that for each* $i \in [1\ldots b]$, *with probability at least* $1-\delta/b$:

$$\sum_{(u,v)\in\mathcal{G}_i} \Delta^2(\mathbf{x}_u,\mathbf{x}_v) \leq \frac{\lambda^2}{b}, \tag{3.13}$$

if the kernel approximation $\mathbf{K}'(\mathbf{x}_u, \mathbf{x}_v) \triangleq \frac{1}{p}\sum_{t=1}^{p} \mathbf{K}_{\epsilon_t}(\mathbf{x}_u, \mathbf{x}_v)$ *is formed using at least*

$$p = \frac{b|\kappa_i|}{\lambda^2 \cdot 2^{a+i}} \log\left(\frac{b|\kappa_i|}{\delta}\right) = \mathbf{O}\left(\frac{\log^2 n}{\lambda^2}\log\left(\frac{\log n}{\delta}\right)\right)$$

samples.

Proof. For all (u, v), we have $\mathbf{K}_{\epsilon_t}(\mathbf{x}_u, \mathbf{x}_v) = \cos(\epsilon_t^\top \mathbf{\Theta}^{-1/2}(\mathbf{x}_u - \mathbf{x}_v))$ where $\epsilon_t \sim \mathbf{N}(0, \mathbf{I})$ and,

$$\mathbf{K}_{\epsilon_t}(\mathbf{x}_u, \mathbf{x}_v) \;=\; \cos\left(\sum_{\ell=1}^{d} \epsilon_t^\ell \cdot \left(\frac{\mathbf{x}_u^\ell - \mathbf{x}_v^\ell}{\theta_\ell}\right)\right) \;\triangleq\; \cos\left(\mathbf{z}_{uv}^t\right) . \tag{3.14}$$

Since $\epsilon_t^\ell \sim \mathbf{N}(0, 1)$, $\mathbf{z}_{uv}^t$ is then a weighted sum of Gaussian random variables and $\mathbf{z}_{uv}^t \sim \mathbf{N}(0, \mathbf{\Sigma}_{uv}^t)$, where $\mathbf{\Sigma}_{uv}^t \triangleq (\mathbf{x}_u - \mathbf{x}_v)^\top \mathbf{\Theta}^{-1}(\mathbf{x}_u - \mathbf{x}_v)$, which in turn implies:

$$\begin{aligned}
\mathbb{E}[\cos(\mathbf{z}_{uv}^t)] \;&=\; \exp\left(-0.5\mathbf{\Sigma}_{uv}^t\right) \;=\; \mathbf{K}(\mathbf{x}_u, \mathbf{x}_v) , \\
\mathbb{V}[\cos(\mathbf{z}_{uv}^t)] \;&=\; \frac{1}{2}\left[1 - \mathbb{E}[\cos(\mathbf{z}_{uv}^t)]^2\right]^2 \;=\; \frac{1}{2}\Big(1 - \mathbf{K}^2(\mathbf{x}_u, \mathbf{x}_v)\Big)^2 \;\leq\; 2 \times \frac{1}{2^{a+i}} ,
\end{aligned} \tag{3.15}$$

where the last inequality follows from the choice of $(u, v) \in \kappa_i$ and the definition of the κ_i above. Next, applying the Chernoff-Hoeffding inequality and union bounding over the κ_i, we have:

$$\begin{aligned}
&\exp \Pr\left(\forall (u, v) \in \kappa_i : \Delta(\mathbf{x}_u, \mathbf{x}_v) \;\leq\; \frac{\epsilon}{p}\right) \;\geq\; 1 - 2|\kappa_i| \exp\left(-\frac{\epsilon^2}{4\sum_{t=1}^{p} \mathbb{V}\left[\cos(\mathbf{z}_{uv}^t)\right]}\right) \\
&\Rightarrow \Pr\left(\sum_{(u,v)\in\kappa_i} \Delta^2(\mathbf{x}_u, \mathbf{x}_v) \;\leq\; \frac{|\kappa_i|\epsilon^2}{p^2}\right) \;\geq\; 1 - 2|\kappa_i| \exp\left(-\frac{\epsilon^2 \cdot 2^{a+i}}{8p}\right) .
\end{aligned} \tag{3.16}$$

Thus, setting $\epsilon^2 = \frac{\lambda^2 p^2}{4b|\kappa_i|}$ and $p \geq \frac{32b|\kappa_i|}{\lambda^2 \cdot 2^{a+i}} \log\left(\frac{2b|\kappa_i|}{\delta}\right)$ yields:

$$\Pr\left(\sum_{(u,v)\in\mathcal{G}_i} \Delta^2(\mathbf{x}_u, \mathbf{x}_v) \leq \frac{\lambda^2}{4b}\right) \geq 1 - 2|\kappa_i| \exp\left(-\frac{\lambda^2 p \cdot 2^{a+i}}{32b|\kappa_i|}\right) \;\geq\; 1 - \frac{\delta}{b} . \tag{3.17}$$

where the last inequality follows from the above choice of p. Since $|\kappa_i| = 2^i$ by Condition **2**, we further have $p \geq \frac{32b}{\lambda^2 \cdot 2^a} \log\left(\frac{b \cdot 2^{b+1}}{\delta}\right) = \mathbf{O}\left(\frac{\log^2 n}{\lambda^2}\log\left(\frac{\log n}{\delta}\right)\right)$. □

Lemma 3.1.9 thus establishes a very strong sample complexity of $\mathbf{O}(\log^2 n \log\log n)$ for approximating all kernel entries within a narrow band of values, which is significantly cheaper than the sample complexity of $\mathbf{O}(n^2 \log n)$ we would get if we were to ignore the distribution of kernel values in different bands. This is made clear in Corollary 3.1.10 below, which combines Lemmas 3.1.6, 3.1.7 and 3.1.9 to establish an overall sample complexity resulting in only a small approximation loss accumulated over all bands.

Corollary 3.1.10. *If a kernel approximation* $\mathbf{K}'$ *of* $\mathbf{K}$ *is formed such that*

$$\mathbf{K}'(\mathbf{x}_u, \mathbf{x}_v) \triangleq \frac{1}{p} \sum_{t=1}^{p} \mathbf{K}_{\epsilon_t}(\mathbf{x}_u, \mathbf{x}_v)$$

for all in-cluster entries $(u, v) \in \mathcal{C}$ *using* $p = \mathbf{O}\left((\log^2 n/\lambda^2) \log(\log n/\delta)\right)$ *samples and* $\mathbf{K}'(\mathbf{x}_{u'}, \mathbf{x}_{v'}) \triangleq 0$ *for all off-cluster entries* $(u', v') \in \mathcal{C}'$*, then,*

$$\|\mathbf{K} - \mathbf{K}'\|_2^2 \;\leq\; \|\mathbf{K} - \mathbf{K}'\|_F^2 \;\leq\; \lambda^2 \,,$$

with probability at least $1 - 2\delta$ *with* $\delta \geq \mathbf{O}\left(\exp(\log n - \sqrt{d})\right)$*. This immediately guarantees that* $\mathbf{K}'$ *is spectrally close to* $\mathbf{K}$ *using the notion of* λ*-closeness (see Definition 3.1.3).*

Proof. By Lemma 3.1.6, with probability $1 - \delta$, $|\kappa_i| = |\mathcal{C}_i|$ simultaneously for all i. Thus, applying a union bound over this event and the results obtained in Lemma 3.1.9 for all clusters, we have the following bound on the total approximation loss over in-cluster entries in $\mathcal{C}$ with probability $1 - 2\delta$:

$$\sum_{(u,v)\in\mathcal{C}} \Delta^2(\mathbf{x}_u, \mathbf{x}_v) \;\leq\; \frac{\lambda^2}{4} \,. \tag{3.18}$$

Furthermore, by Lemma 3.1.7, we also have the following bound on the total approximation loss over off-cluster entries in $\mathcal{C}'$ (which were approximated uniformly by zero):

$$\sum_{(u,v)\in\mathcal{C}'} \Delta^2(\mathbf{x}_u, \mathbf{x}_v) \;\leq\; \frac{3n^2}{4}\Big(\mathbf{K}(\mathbf{x}_u, \mathbf{x}_v) - 0\Big)^2 \;\leq\; \frac{3n^2}{4}\sqrt{1 - \frac{1}{2^a}} \;=\; \frac{3\lambda^2}{4} \,, \tag{3.19}$$

when the last inequality is due to the facts (established in Lemma 3.1.7) that $\mathbf{K}^4(\mathbf{x}_u, \mathbf{x}_v) \leq 1 - 1/2^a$ and that $a = \dfrac{1}{\log 2} \log\left(\dfrac{n^4}{n^4 - \lambda^4}\right)$. Finally, combining these yields:

$$\|\mathbf{K} - \mathbf{K}'\|_2^2 \leq \|\mathbf{K} - \mathbf{K}'\|_{\mathrm{F}}^2 = \sum_{(u,v)\in\mathcal{C}} \Delta^2(\mathbf{x}_u, \mathbf{x}_v) + \sum_{(u,v)\in\mathcal{C}'} \Delta^2(\mathbf{x}_u, \mathbf{x}_v) \leq \frac{1}{4}\lambda^2 + \frac{3}{4}\lambda^2 = \lambda^2 \tag{3.20}$$

□

3.1.3.2 Approximation Loss for Prediction and Model Evidence

In terms of prediction and model evidence approximation, our result holds simultaneously for all parameter configurations and is thus oblivious to the choice of parameters (see Theorem 3.1.11). While existing kernel sketch methods (Avron et al., 2017; Musco and Musco, 2016) generically

achieve near-linear complexity for the approximate feature map[6], they often require knowledge of the parameters to construct the kernel approximations. In contrast, our result in Theorem 3.1.4 can be leveraged to bound the same prediction discrepancy when the original and approximated GPs use their own optimized parameter configurations, as shown in Theorem 3.1.19 below. To establish Theorem 3.1.19, however, we first establish an intermediate result that bounds the prediction and model evidence in the case when both the original and approximated GPs use the same parameter configurations.

Theorem 3.1.11. *Let $\delta < 1$ be a user-specified confidence as defined previously in Theorem 3.1.4 and let $\mathbf{K}'$ be an approximation to $\mathbf{K}$ for which $\|\mathbf{K} - \mathbf{K}'\|_2^2 \leq \lambda^2$ with probability $1 - \delta$, uniformly over the entire parameter space. Then, with probability $1 - \delta$, the following hold:*

$$\mathbb{E}[g(\mathbf{x}_*)] \;=\; \left(1 \pm \frac{\lambda}{\sigma^2}\right) \mathbb{E}[g'(\mathbf{x}_*)] \qquad \textit{and} \qquad \mathbb{V}[g(\mathbf{x}_*)] \;=\; \left(1 \pm \frac{\lambda}{\sigma^2}\right) \mathbb{V}[g'(\mathbf{x}_*)] \;\pm\; \frac{\lambda}{\sigma^2} \tag{3.21}$$

where σ^2 is the noise of the variance (Equation 3.2), and $g(\mathbf{x}_*)$, $g'(\mathbf{x}_*)$ respectively denote the predictive distributions of the full GP and the approximated GP pertaining to an arbitrary test input $\mathbf{x}_*$.

Proof. This follows directly from Lemma 3.1.13 and Lemma 3.1.14 below. □

Detailed Proof of Theorem 3.1.11

Lemma 3.1.12. *Let $\mathbf{K}$ and $\mathbf{K}'$ be positive semidefinite matrices in $\mathbb{R}^{n \times n}$ such that $-\lambda\mathbf{I} \preceq \mathbf{K} - \mathbf{K}' \preceq \lambda\mathbf{I}$, $\mathbf{Q} \triangleq \mathbf{K} + \sigma^2\mathbf{I}$ and $\mathbf{Q}' \triangleq \mathbf{K}' + \sigma^2\mathbf{I}$ for some $\lambda, \sigma > 0$, then:*

$$\|\mathbf{Q}'^{-1}\|_2 = \left(1 \pm \frac{\lambda}{\sigma^2}\right) \|\mathbf{Q}^{-1}\|_2 \,. \tag{3.22}$$

Proof. By definition of the spectral norm, we have $\forall \mathbf{x} \in \mathbb{R}^n$:

$$\mathbf{K} - \mathbf{K}' \;\preceq\; \lambda\mathbf{I} \,, \tag{3.23}$$

which implies

$$\begin{aligned} \mathbf{Q} &\preceq \mathbf{K}' + (\sigma^2 + \lambda)\mathbf{I} \\ &\preceq \left(\frac{\sigma^2 + \lambda}{\sigma^2}\right) \mathbf{K}' + (\sigma^2 + \lambda)\mathbf{I} \\ &= \left(1 + \frac{\lambda}{\sigma^2}\right) \mathbf{Q}' \,. \end{aligned} \tag{3.24}$$

[6](Avron et al., 2017; Musco and Musco, 2016) achieves a complexity of $\mathbf{O}(nm^2)$ where m scales with the effective dimension of the kernel matrix.

where $\preceq$ and $\succeq$ denote the Loewner inequality operators. Likewise, by symmetry, we also have:

$$\mathbf{Q}' \preceq \left(1 + \frac{\lambda}{\sigma^2}\right)\mathbf{Q}\,. \tag{3.25}$$

Let $\mathbf{A} \triangleq (1 + \lambda/\sigma^2)\mathbf{Q}'$ and $\mathbf{B} \triangleq \mathbf{Q}$. Since $\mathbf{A}$ and $\mathbf{B}$ are symmetric and positive semidefinite, there exist $\mathbf{U}, \mathbf{V}$ with orthogonal rows and columns and diagonal matrices $\mathbf{\Sigma}, \mathbf{\Sigma}'$ for which $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^\top$ and $\mathbf{B} = \mathbf{V}\mathbf{\Sigma}'\mathbf{V}^\top$. We further let $\mathbf{A}^{-1/2} \triangleq \mathbf{U}\mathbf{\Sigma}^{-1/2}$ and $\mathbf{B}^{-1/2} \triangleq \mathbf{V}\mathbf{\Sigma}'^{-1/2}$.

Then, we can rewrite Equation 3.24 as:

$$\begin{aligned}
\mathbf{A} - \mathbf{B} &\succeq 0 \\
\Rightarrow \mathbf{B}^{-1/2}(\mathbf{A} - \mathbf{B})\mathbf{B}^{-1/2} &\succeq 0 \\
\Rightarrow \mathbf{B}^{-1/2}\mathbf{A}\mathbf{B}^{-1/2} - \mathbf{I} &\succeq 0 \\
\Rightarrow \mathbf{A}^{-1/2}\mathbf{B}^{1/2}(\mathbf{B}^{-1/2}\mathbf{A}\mathbf{B}^{-1/2})\mathbf{B}^{-1/2}\mathbf{A}^{1/2} &\succeq \mathbf{A}^{-1/2}\mathbf{B}^{1/2}\mathbf{B}^{-1/2}\mathbf{A}^{1/2} \\
\Rightarrow \mathbf{A}^{1/2}\mathbf{B}^{-1}\mathbf{A}^{1/2} &\succeq \mathbf{I} \\
\Rightarrow \mathbf{A}^{-1/2}(\mathbf{A}^{1/2}\mathbf{B}^{-1}\mathbf{A}^{1/2})\mathbf{A}^{-1/2} &\succeq \mathbf{A}^{-1/2}\mathbf{A}^{-1/2} \\
\Rightarrow \mathbf{B}^{-1} &\succeq \mathbf{A}^{-1} \\
\Rightarrow \mathbf{Q}^{-1} &\succeq \frac{\sigma^2}{\sigma^2 + \lambda}\mathbf{Q}'^{-1} \\
\Rightarrow \left(1 + \frac{\lambda}{\sigma^2}\right)\mathbf{Q}^{-1} &\succeq \mathbf{Q}'^{-1}\,.
\end{aligned} \tag{3.26}$$

Again, by symmetry, we can rewrite Equation 3.25 as:

$$\begin{aligned}
\mathbf{Q}'^{-1} &\succeq \frac{\sigma^2}{\sigma^2 + \lambda}\mathbf{Q}^{-1} \succeq \left(1 - \frac{\lambda}{\sigma^2 + \lambda}\right)\mathbf{Q}^{-1} \\
&\succeq \left(1 - \frac{\lambda}{\sigma^2}\right)\mathbf{Q}^{-1}\,.
\end{aligned} \tag{3.27}$$

Therefore, we have $\|\mathbf{Q}'^{-1}\|_2 = (1 \pm \lambda/\sigma^2)\|\mathbf{Q}^{-1}\|_2$. □

Let $g(\mathbf{x}_*)$ and $g'(\mathbf{x}_*)$ respectively denote the predictive distributions of full GP and the approximated GP pertaining to an arbitrary test input $\mathbf{x}_*$. We then state the following lemmas:

Lemma 3.1.13. *Let $\mathbf{K}'$ denote an approximation that is λ-close to the original kernel $\mathbf{K}$. The induced predictive mean of $\mathbf{K}'$ is bounded by a factor of $1 \pm \lambda/\sigma^2$ times the original predictive mean.*

$$\mathbb{E}[g(\mathbf{x}_*)] = \left(1 \pm \frac{\lambda}{\sigma^2}\right)\mathbb{E}[g'(\mathbf{x}_*)]\,. \tag{3.28}$$

Proof. Let $\mathbf{k}_* \triangleq [k(\mathbf{x}_*, \mathbf{x}_i)]_{i=1}^n$ where $\mathbf{x}_i$ denotes the i-th training data point. We have:

$$\begin{aligned}
\mathbb{E}[g(\mathbf{x}_*)] &= \frac{1}{2}\left((\mathbf{k}_* + \mathbf{y})^\top \mathbf{Q}^{-1}(\mathbf{k}_* + \mathbf{y}) - \mathbf{k}_*^\top \mathbf{Q}^{-1}\mathbf{k}_* - \mathbf{y}^\top \mathbf{Q}^{-1}\mathbf{y}\right) \\
&= \frac{1}{2}\left(1 \pm \frac{\lambda}{\sigma^2}\right)\left((\mathbf{k}_* + \mathbf{y})^\top \mathbf{Q}'^{-1}(\mathbf{k}_* + \mathbf{y}) - \mathbf{k}_*^\top \mathbf{Q}'^{-1}\mathbf{k}_* - \mathbf{y}^\top \mathbf{Q}'^{-1}\mathbf{y}\right) \\
&= \left(1 \pm \frac{\lambda}{\sigma^2}\right)\mathbb{E}[g'(\mathbf{x}_*)]\,, \qquad (3.29)
\end{aligned}$$

where the first and third equations follow from adding and subtracting the same terms to the expression of $g(\mathbf{x}_*)$ – see Equation 3.2 – while the second equation follows from applying Lemma 3.1.12 above. □

Lemma 3.1.14. *Let $\mathbf{K}'$ denote an approximation that is λ-close to the original kernel $\mathbf{K}$. The induced predictive variance of $\mathbf{K}'$ is bounded by a factor of $1 \pm \lambda/\sigma^2$ of the original predictive variance up to a constant bias of λ/σ^2,*

$$\mathbb{V}[g(\mathbf{x}_*)] = \left(1 \pm \frac{\lambda}{\sigma^2}\right)\mathbb{V}[g'(\mathbf{x}_*)] \pm \frac{\lambda}{\sigma^2}\,. \qquad (3.30)$$

Proof. Following the definition of the Gaussian kernel, we assume that the signal of the SE (Squared Exponential) kernel is unitary [7]. As such,

$$\begin{aligned}
\mathbb{V}[g(\mathbf{x}_*)] &= 1 - \mathbf{k}_*^\top \mathbf{Q}^{-1}\mathbf{k}_* \\
&= 1 - \left(1 \pm \frac{\lambda}{\sigma^2}\right)\mathbf{k}_*^\top \mathbf{Q}'^{-1}\mathbf{k}_* \\
&= \left(1 \pm \frac{\lambda}{\sigma^2}\right)\left(1 - \mathbf{k}_*^\top \mathbf{Q}'^{-1}\mathbf{k}_*\right) \pm \frac{\lambda}{\sigma^2} \\
&= \left(1 \pm \frac{\lambda}{\sigma^2}\right)\mathbb{V}[g'(\mathbf{x}_*)] \pm \frac{\lambda}{\sigma^2}\,, \qquad (3.31)
\end{aligned}$$

where (again) the above equation follows straightforwardly from applying Lemma 3.1.12 and standard algebraic manipulation. Lemma 3.1.13 and Lemma 3.1.14 thus provide an explicit bound on the difference between the original and approximated predictive distributions. We will now establish another bound on the difference between the original and approximated negative log-likelihoods (i.e., the training objectives) in Lemma 3.1.15 and Lemma 3.1.16 below. □

Lemma 3.1.15. *Let $\mathbf{K}'$ denote an approximation that is λ-close to the original kernel $\mathbf{K}$. Let $\mathbf{Q} = \mathbf{K} + \sigma^2\mathbf{I}$ and $\mathbf{Q}' = \mathbf{K}' + \sigma^2\mathbf{I}$. We have:*

$$\log|\mathbf{Q}'| = \left(1 \pm \tau_{\lambda,\sigma}(\mathbf{K})\right)\log|\mathbf{Q}|\,. \qquad (3.32)$$

[7] This simplifies the analysis and does not restrict the expressiveness of the kernel since we can either normalize the output or absorb it into the length-scales (i.e., the θ_i).

where the spectral constant $\tau_{\lambda,\sigma}(\mathbf{K})$ of $\mathbf{K}$ is defined below:

$$\tau_{\lambda,\sigma}(\mathbf{K}) \triangleq \frac{max\Big(\Big|\log\left(1+\frac{\lambda}{\sigma^2}\right)\Big|, \Big|\log\left(1-\frac{\lambda}{\sigma^2}\right)\Big|\Big)}{min\Big(\Big|\log(\lambda_{\min}(\mathbf{K})+\sigma^2)\Big|, \Big|\log(\lambda_{\max}(\mathbf{K})+\sigma^2)\Big|\Big)} . \tag{3.33}$$

Proof. Let $\lambda_1 \leq \lambda_2 \cdots \leq \lambda_n$ and $\lambda'_1 \leq \lambda'_2 \cdots \leq \lambda'_n$ be the eigenvalues of $\mathbf{K}$ and $\mathbf{K}'$ respectively. Applying the Courant-Fischer theorem on the result obtained in Lemma 3.1.12, we have:

$$\lambda'_i + \sigma^2 = \left(1 \pm \frac{\lambda}{\sigma^2}\right)(\lambda_i + \sigma^2) . \tag{3.34}$$

This implies:

$$\begin{aligned}
\log|\mathbf{Q}'| &\leq \sum_{i=1}^{n}\Big|\log(\lambda'_i+\sigma^2)\Big| = \sum_{i=1}^{n}\left|\log(\lambda_i+\sigma^2)+\log\left(1\pm\frac{\lambda}{\sigma^2}\right)\right| \\
&\leq \sum_{i=1}^{n}\Big|\log(\lambda_i+\sigma^2)\Big| + \sum_{i=1}^{n}\max\left(\left|\log\left(1+\frac{\lambda}{\sigma^2}\right)\right|, \left|\log\left(1-\frac{\lambda}{\sigma^2}\right)\right|\right) \\
&\leq \Big(1+\tau_{\lambda,\sigma}(\mathbf{K})\Big)\sum_{i=1}^{n}\Big|\log(\lambda_i+\sigma^2)\Big| = \Big(1+\tau_{\lambda,\sigma}(\mathbf{K})\Big)\log|\mathbf{Q}| .
\end{aligned} \tag{3.35}$$

Similarly, by symmetry, we have:

$$\begin{aligned}
\log|\mathbf{Q}'| &\geq \sum_{i=1}^{n}\Big|\log(\lambda_i+\sigma^2)\Big| - \sum_{i=1}^{n}\max\left(\left|\log\left(1+\frac{\lambda}{\sigma^2}\right)\right|, \left|\log\left(1-\frac{\lambda}{\sigma^2}\right)\right|\right) \\
&\geq \Big(1-\tau_{\lambda,\sigma}(\mathbf{K})\Big)\sum_{i=1}^{n}\Big|\log(\lambda_i+\sigma^2)\Big| = \Big(1-\tau_{\lambda,\sigma}(\mathbf{K})\Big)\log|\mathbf{Q}| .
\end{aligned} \tag{3.36}$$

Together, Equation 3.35 and Equation 3.36 imply $\log|\mathbf{Q}'| = \Big(1 \pm \tau_{\lambda,\sigma}(\mathbf{K})\Big)\log|\mathbf{Q}|$. □

Lemma 3.1.16. *Let $\mathbf{K}'$ denote an approximation that is λ-close to the original kernel $\mathbf{K}$. With $\tau_{\lambda,\sigma}(\mathbf{K})$ previously defined in Lemma 3.1.15, we have:*

$$\ell'(\mathbf{\Theta}) = \left(1 \pm max\left(\tau_{\lambda,\sigma}(\mathbf{K}), \frac{\lambda}{\sigma^2}\right)\right)\ell(\mathbf{\Theta}) . \tag{3.37}$$

where $\ell(\mathbf{\Theta})$ and $\ell'(\mathbf{\Theta})$ respectively denote the negative log likelihood of the full GP and the approximated GP evaluated at the hyper-parameters $\mathbf{\Theta} = diag[\theta_1^2, \theta_2^2 \ldots \theta_d^2]$ as defined previously.

Proof. We have:

$$\begin{aligned}
\ell'(\boldsymbol{\Theta}) &= \frac{1}{2}\log|\mathbf{Q}'| + \frac{1}{2}\mathbf{y}^\top \mathbf{Q}'^{-1}\mathbf{y} \\
&= \frac{1}{2}\left(1 \pm \tau_{\lambda,\sigma}(\mathbf{K})\right)\log|\mathbf{Q}| + \frac{1}{2}\left(1 \pm \frac{\lambda}{\sigma^2}\right)\mathbf{y}^\top \mathbf{Q}^{-1}\mathbf{y} \\
&= \left(1 \pm \max\left(\tau_{\lambda,\sigma}(\mathbf{K}), \frac{\lambda}{\sigma^2}\right)\right)\frac{1}{2}\Big(\log|\mathbf{Q}| + \mathbf{y}^\top \mathbf{Q}^{-1}\mathbf{y}\Big) \\
&= \left(1 \pm \max\left(\tau_{\lambda,\sigma}(\mathbf{K}), \frac{\lambda}{\sigma^2}\right)\right)\ell(\boldsymbol{\Theta})\,. \qquad \square
\end{aligned} \tag{3.38}$$

Using the result of Lemma 3.1.16 above, we can further analyze how the quality of the optimized parameter $\boldsymbol{\Theta}'_* = \arg\max_{\boldsymbol{\Theta}} \ell'(\boldsymbol{\Theta})$ of the approximated training objective compares to the true optimizer of the original objective function $\boldsymbol{\Theta}_* = \arg\max_{\boldsymbol{\Theta}} \ell(\boldsymbol{\Theta})$ in Lemma 3.1.17 below.

Lemma 3.1.17. *Let $\boldsymbol{\Theta}_*$ and $\boldsymbol{\Theta}'_*$ denote the optimal hyper-parameters obtained by respectively minimizing the negative log likelihood of the full GP and the approximated GP. We have:*

$$\ell'(\boldsymbol{\Theta}'_*) = \left(1 \pm \mathit{max}\left(\tau_{\lambda,\sigma}(\mathbf{K}), \frac{\lambda}{\sigma^2}\right)\right)\ell(\boldsymbol{\Theta}_*)\,. \tag{3.39}$$

Proof. By Lemma 3.1.16, we have:

$$\begin{aligned}
\ell'(\boldsymbol{\Theta}'_*) &\leq \ell'(\boldsymbol{\Theta}_*) \\
&\leq \left(1 + \max\left(\tau_{\lambda,\sigma}(\mathbf{K}), \frac{\lambda}{\sigma^2}\right)\right)\ell(\boldsymbol{\Theta}_*)
\end{aligned} \tag{3.40}$$

and

$$\begin{aligned}
\ell'(\boldsymbol{\Theta}'_*) &\geq \left(1 - \max\left(\tau_{\lambda,\sigma}(\mathbf{K}), \frac{\lambda}{\sigma^2}\right)\right)\ell(\boldsymbol{\Theta}'_*) \\
&\geq \left(1 - \max\left(\tau_{\lambda,\sigma}(\mathbf{K}), \frac{\lambda}{\sigma^2}\right)\right)\ell(\boldsymbol{\Theta}_*)\,.
\end{aligned} \tag{3.41}$$

Together, these results imply $\ell'(\boldsymbol{\Theta}'_*) = \left(1 \pm \max\left(\tau_{\lambda,\sigma}(\mathbf{K}), \frac{\lambda}{\sigma^2}\right)\right)\ell(\boldsymbol{\Theta}_*)$. $\square$

Lemma 3.1.18. *Let $\delta \in (0,1)$ and let $\mathbf{K}'$ denote an approximation of $\mathbf{K}$ for which $\|\mathbf{K} - \mathbf{K}'\|_2^2 \leq \lambda^2$ with probability at least $1-\delta$ uniformly over the entire parameter space. Let $\boldsymbol{\Theta}_*$ and $\boldsymbol{\Theta}'_*$ denote the optimal hyper-parameters obtained by respectively minimizing the negative log-likelihood of the full GP and the approximated GP. Then, with probability $1-\delta$, the following holds:*

$$\mathbb{E}[g'(\mathbf{x}_*; \boldsymbol{\Theta}'_*)] = (1 \pm \rho(\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*)) \cdot \mathbb{E}[g(\mathbf{x}_*; \boldsymbol{\Theta}_*)] + \wp(\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*) \tag{3.42}$$

where $\rho(\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*)$ and $\wp(\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*)$ are constant with respect to $\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*$

Proof. We have:

$$\begin{aligned}
\mathbb{E}[g(\mathbf{x}_*;\boldsymbol{\Theta}_*)] &= \mathbf{k}_*^\top \mathbf{Q}^{-1}\mathbf{y}\Big|_{\boldsymbol{\Theta}_*} \\
&= \frac{1}{2}\Big[(\mathbf{k}_*+\mathbf{y})^\top\mathbf{Q}^{-1}(\mathbf{k}_*+\mathbf{y}) - \mathbf{k}_*\mathbf{Q}^{-1}\mathbf{k}_* + \log|\mathbf{Q}|\Big]\Big|_{\boldsymbol{\Theta}_*} - \frac{1}{2}\ell(\boldsymbol{\Theta}_*) \\
&\geq -\frac{1}{2}\left[\ell(\boldsymbol{\Theta}_*) + 1 - \sum_{i=1}^{n}\log\left(\lambda_i+\sigma^2\right)\Big|_{\boldsymbol{\Theta}_*}\right] \qquad (3.43)
\end{aligned}$$

On the other hand, we have:

$$\begin{aligned}
\mathbb{E}[g(\mathbf{x}_*;\boldsymbol{\Theta}_*)] &= \mathbf{k}_*^\top \mathbf{Q}^{-1}\mathbf{y} \\
&\leq \frac{1}{2}\Big[\mathbf{k}_*^\top\mathbf{Q}^{-1}\mathbf{k}_* + \mathbf{y}^\top\mathbf{Q}^{-1}\mathbf{y}\Big]\Big|_{\boldsymbol{\Theta}_*} \\
&\leq \frac{1}{2}\left[\ell(\boldsymbol{\Theta}_*) + 1 - \sum_{i=1}^{n}\log\left(\lambda_i+\sigma^2\right)\Big|_{\boldsymbol{\Theta}_*}\right] \qquad (3.44)
\end{aligned}$$

Thus, we have:

$$\mathbb{E}[g(\mathbf{x}_*;\boldsymbol{\Theta}_*)] = \pm\frac{1}{2}\left[\ell(\boldsymbol{\Theta}_*) + 1 - \sum_{i=1}^{n}\log\left(\lambda_i+\sigma^2\right)\Big|_{\boldsymbol{\Theta}_*}\right] \qquad (3.45)$$

and by symmetry:

$$\begin{aligned}
\mathbb{E}[g'(\mathbf{x}_*;\boldsymbol{\Theta}'_*)] &= \pm\frac{1}{2}\left[\ell'(\boldsymbol{\Theta}'_*) + 1 - \sum_{i=1}^{n}\log\left(\lambda'_i+\sigma^2\right)\Big|_{\boldsymbol{\Theta}'_*}\right] \\
&= (1\pm\rho(\lambda,\sigma,\boldsymbol{\Theta}_*,\boldsymbol{\Theta}'_*))\cdot\mathbb{E}[g(\mathbf{x}_*;\boldsymbol{\Theta}_*)] \;+\; \wp(\lambda,\sigma,\boldsymbol{\Theta}_*,\boldsymbol{\Theta}'_*) \qquad (3.46)
\end{aligned}$$

where $\wp(\boldsymbol{\Theta}_*,\boldsymbol{\Theta}'_*)$ is a constant as defined below:

$$\begin{aligned}
\rho(\lambda,\sigma,\boldsymbol{\Theta}_*,\boldsymbol{\Theta}'_*) &\triangleq \max\left(\tau_{\lambda,\sigma}(\mathbf{K}), \tau_{\lambda,\sigma}(\mathbf{K}'), \frac{\lambda}{\sigma^2}\right) \\
\wp(\lambda,\sigma,\boldsymbol{\Theta}_*,\boldsymbol{\Theta}'_*) &\triangleq \left(\sum_{i=1}^{n}\log\frac{\lambda_i+\sigma^2\Big|_{\boldsymbol{\Theta}_*}}{\lambda'_i+\sigma^2\Big|_{\boldsymbol{\Theta}'_*}}\right) \pm \rho(\lambda,\sigma,\boldsymbol{\Theta}_*,\boldsymbol{\Theta}'_*)\cdot\left(1-\sum_{i=1}^{n}\log(\lambda_i+\sigma^2)\Big|_{\boldsymbol{\Theta}_*}\right)
\end{aligned}$$

□

Finally, Theorem 3.1.19 analyzes how close the approximated predictive mean is to the full GP predictive mean when both are evaluated at the optimizer of their respective training objective.

Theorem 3.1.19. *Let $\delta < 1$ be user-specified confidence as defined in Theorem 3.1.4. Let $\mathbf{K}'$ denote an approximation to $\mathbf{K}$ for which $\|\mathbf{K} - \mathbf{K}'\|_2^2 \leq \lambda^2$ with probability at least $1-\delta$ uniformly over the entire parameter space. Let $\boldsymbol{\Theta}_*$ and $\boldsymbol{\Theta}'_*$ denote the optimal hyperparameters obtained by respectively minimizing the negative log-likelihood of a full GP and the approximated GP. With probability $1-\delta$, the following holds:*

$$\mathbb{E}[g'(\mathbf{x}_*; \boldsymbol{\Theta}'_*)] \quad = \quad (1 \pm \rho(\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*)) \cdot \mathbb{E}[g(\mathbf{x}_*; \boldsymbol{\Theta}_*)] \quad + \quad \wp(\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*) \tag{3.47}$$

where $\rho(\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*)$ and $\wp(\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*)$ are constants with respect to $\lambda, \sigma, \boldsymbol{\Theta}_*, \boldsymbol{\Theta}'_*$.

Proof. This follows immediately from Lemma 3.1.17 above, which was built on the result of Theorem 3.1.11 above. This completes our loss analysis for SSGPs. □

3.1.3.3 Optimizing Feature Map Complexity

We next present a practical probabilistic embedding algorithm that transforms the input data to meet the requirements of Conditions **1-3**. Our method is built on the rich literature of variational auto-encoders, a.k.a VAE (Kingma and Welling, 2013), which is a broad class of deep generative models that combine the rigor of Bayesian methods and rich parameterization of (deep) neural networks to discover (non-linear) low-dimensional embeddings of data while preserving their statistical properties. We first provide a short review on VAEs below, followed by an augmentation that aims to achieve the impositions in Conditions **1-3** above.

Variational Auto-Encoders (VAEs)

Let $\mathbf{x}$ be a random variable with density function $p(\mathbf{x})$. We want to learn a latent variable model $p_\theta(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p_\theta(\mathbf{z}|\mathbf{x})$ that captures this generative process. The latent variable model comprises a fixed latent prior $p(\mathbf{z})$ and a parametric likelihood $p_\theta(\mathbf{z}|\mathbf{x})$. To learn θ, we maximize the variational evidence lower-bound (ELBO) $\mathbf{L}(\mathbf{x}; \theta, \phi)$ of $\log p_\theta(\mathbf{x})$:

$$\mathbf{L}(\mathbf{x}; \theta, \phi) \quad \triangleq \quad \mathbb{E}_{\mathbf{z} \sim q_\phi}\Big[\log p_\theta(\mathbf{x}|\mathbf{z})\Big] - \mathbb{KL}\Big(q_\phi(\mathbf{z}||\mathbf{x})||p(\mathbf{z})\Big) \tag{3.48}$$

with respect to an arbitrary posterior surrogate $q_\phi(\mathbf{z}|\mathbf{x}) \simeq p_\theta(\mathbf{z}|\mathbf{x})$ over the latent variable $\mathbf{z}$. The ELBO is always a lower-bound on $\log p_\theta(\mathbf{x})$ regardless of our choice of $q_\phi(\mathbf{z}|\mathbf{x})$. This is due to the non-negativity of the KL divergence as seen in the first part of the above equation.

This can be viewed as a stochastic auto-encoder with $p_\theta(\mathbf{x}|\mathbf{z})$ and $q_\phi(\mathbf{z}|\mathbf{x})$ acting as the encoder and decoder, respectively. Here, θ and ϕ characterize the neural network parameterization of these models. Their learning is enabled via a re-parameterization of $q_\phi(\mathbf{z}|\mathbf{x})$ that enables stochastic gradient ascent.

Re-configuring Data via an Augmenting Variational Auto-Encoder

To augment the above VAE framework (Kingma and Welling, 2013; Mathieu et al., 2019) to account for the impositions in Conditions **1** and **2**, we ideally want to configure the parameterization of the above generative process to guarantee that the marginal posterior $q(\mathbf{z}) = \int_{\mathbf{x}} q(\mathbf{z}|\mathbf{x})p(\mathbf{x})\mathrm{d}\mathbf{x}$ will manifest itself in the form of a mixture of Gaussians with the desired concentration and population densities as stated in Condition **1**.

However, it is often difficult to make such an imposition directly given that we typically have no prior knowledge of $p(\mathbf{x})$. We instead impose the desired structure on the latent prior $p(\mathbf{z})$ and then penalize the divergence between $q_\phi(\mathbf{z})$ and $p(\mathbf{z})$ while optimizing for the above ELBO in Equation 3.48. That is, we parameterize $p(\mathbf{z}) = \pi_1\ \mathbf{N}(\mathbf{z}; \mathbf{c}_1, \gamma_1^2\boldsymbol{\Theta}^{-1}) + \ldots + \pi_b\ \mathbf{N}(\mathbf{z}; \mathbf{c}_b, \gamma_b^2\boldsymbol{\Theta}^{-1})$ where $\pi_i \propto 2^{i/2}$ (see Condition **2**), which encodes the desired clustering structure. This is then reflected on the marginal posterior $q(\mathbf{z})$ via augmenting the above ELBO as,

$$\mathbf{L}_\alpha(\mathbf{x}; \theta, \phi) \triangleq \mathbb{E}_{\mathbf{z}\sim q_\phi}\Big[\log p_\theta(\mathbf{x}|\mathbf{z})\Big] - \mathbb{KL}\Big(q_\phi(\mathbf{z}||\mathbf{x})||p(\mathbf{z})\Big) - \alpha\mathbb{KL}\Big(q(\mathbf{z})||p(\mathbf{z})\Big), \tag{3.49}$$

where the penalty term $\alpha\mathbb{KL}(q(\mathbf{z})||p(\mathbf{z}))$ serves as an incentive to encourage $q(\mathbf{z})$ to assume the same clustering structure as $p(\mathbf{z})$. The parameter α can be manually set to adjust the strength of the incentive. To encourage separation among learned clusters (see Condition **3**), we also add an extra penalty term to the above augmented ELBO,

$$\mathbf{L}_{\alpha,\beta}(\mathbf{x}; \theta, \phi) \triangleq \mathbf{L}_\alpha(\mathbf{x}; \theta, \phi) + \beta \sum_{i\neq j} \mathbb{KL}\Big(\mathbf{N}\left(\mathbf{z}; \mathbf{c}_i, \gamma_i^2\boldsymbol{\Theta}^{-1}\mathbf{I}\right)\ ||\ \mathbf{N}\left(\mathbf{z}; \mathbf{c}_j, \gamma_j^2\boldsymbol{\Theta}^{-1}\mathbf{I}\right)\Big). \tag{3.50}$$

Once these clusters are learned, we can use the resulting encoding network $q_\phi(\mathbf{z}|\mathbf{x})$ to transform each training input $\mathbf{x}$ into its latent projection and subsequently train an SSGP on the latent space of $\mathbf{z}$ (instead of training it on the original data space). Our previous analysis can then be applied to $\mathbf{z}$ to give the desired sample complexity. The empirical efficiency of the proposed method is demonstrated in Section 3.1.4 below. Note that the cost of training the embedding is linear in the number of data points and therefore does not noticeably affect our overall running time.

3.1.4 Experiments

Datasets. This section presents our empirical studies on two real datasets: (a) the ABALONE dataset (Waugh) with 3000 data points which were used to train a model that predicts the age of abalone (number of rings on its shell) from physical measurements such as length, diameter, height, whole weight, shucked weight, viscera weight, and shell weight; and (b) the GAS SENSOR dataset with 4 million data points (Burgués et al., 2018; Burgués and Marco, 2018) which was used to train a model that predicts the CO concentration (ppm) from measurements of humidity, temperature, flow rate, heater voltage and the resistant measures of 14 gas sensors.

In both settings, we compare our revised SSGP method with the traditional SSGP on both datasets to demonstrate its sample efficiency. In particular, our SSGP method is applied on the

embedded space of data which was generated and configured using the auto-encoding method in Section 3.1.3.3 to approximately meet the aforementioned Conditions **1-3**.

The traditional SSGP method on the other hand was applied directly to the data space. The prediction root-mean-square-error (RMSE) achieved by each method is reported at different sample complexities in Figure 3.1 below. All reported performances were averaged over 5 independent runs on a computing server with a Tesla K40 GPU with 12GB RAM.

Detailed Model Parameterization

Our embedding algorithm is based on a VAE implementation where the latent prior, posterior, and the likelihood of the data generation process are represented via separate mixtures of k Gaussian distributions over a 4-dimensional space. For the latent prior, we set (and fixed) the means of each Gaussian component (i.e., the prior cluster means) at k equidistant points on a 4-dimensional sphere centered at zero with an optimizable radius. For the latent posterior and likelihood, the mean and covariance entries of each component in the mixture are parameterized as outputs of their respective neural networks, which we refer to as Gaussian nets.

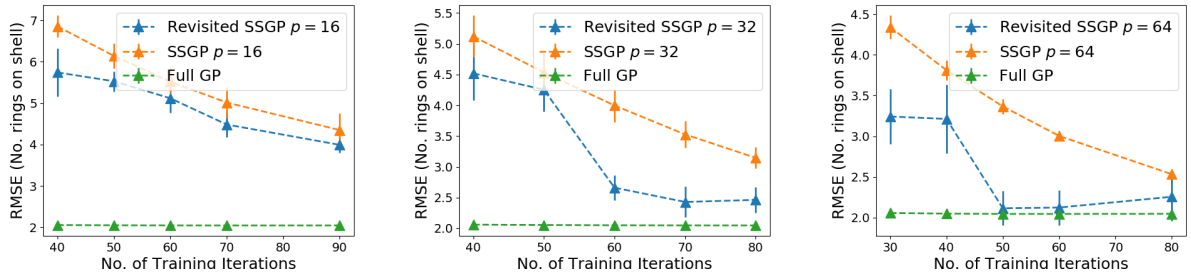
In turn, the Gaussian nets are parameterized separately. Each starts with a linear layer comprising of 10 neurons whose outputs are fed simultaneously to two separate hidden (linear) layers with 10 hidden neurons each. Their outputs are then used to form the mean and covariance entries of the corresponding Gaussian component. All neurons are activated by a ReLU unit, and in addition, the (batch) outputs of the first linear layer are also standardized via a learnable 1D batch-norm layer to ensure the stability of batch optimization. The mixing weights that combine such Gaussian nets in the mixtures are also parameterized as the outputs of a linear layer with $k = 8$ neurons where $k = 8$ is also the number of components in our mixture.

The above parameterized latent prior, posterior, and likelihood are then connected in the variational lower-bound (ELBO) as expressed in the first two terms of Equation 3.49. This ELBO objective is then combined with two regularization terms weighted with (manually tuned) parameters $\alpha = 8.0$ and $\beta = 1.2$ as detailed in Equation 3.50. The entire function is optimized via gradient descent using the standard Adam optimizer with the default setting implemented in PyTorch (Paszke et al., 2017).

Once learned, the outputs of the latent posterior were used as the encoded data which were fed as input to our revisited SSGP. For practical implementation, we also found that additionally passing the encoded data to the latent likelihood generates a reconfigured version of the original data which helps to marginally improve the performance. All of our reported results below are generated with respect to this version of reconfiguration. All of our implementations of GP, SSGP and revisited SSGP that make use of the output of this reconfiguration process, are also in PyTorch.

Results and Discussions

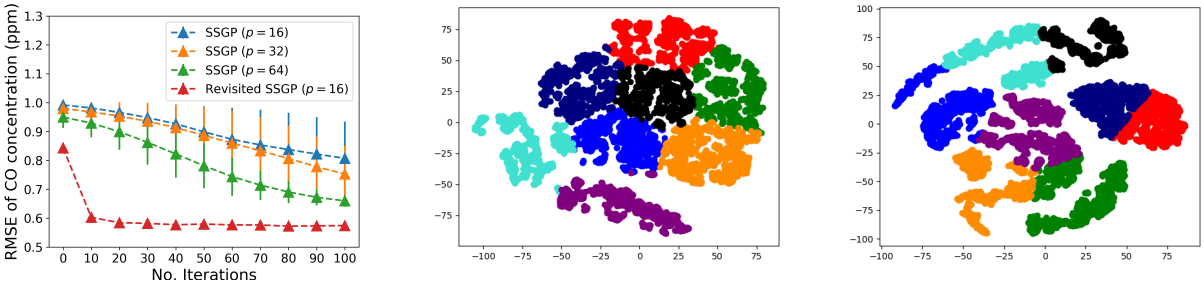
It can be observed from the results that at all levels of sample complexity, the revised SSGP achieves substantially better performance than its vanilla SSGP counterpart (Figures 3.1 and 3.2).



(a) (b) (c)

Figure 3.1: Performance comparison between our revised SSGP and the traditional SSGP on the ABALONE dataset (Waugh) at varying sample complexities (see Theorem 3.1.5) $p = 16, 32$ and 64.

This is expected since our revised SSGP is guaranteed to require many fewer samples than the vanilla SSGP when the data is reconfigured to exhibit a certain clustering structure (see Conditions **1-3** and Theorem 3.1.4). As such, when both are set to operate at the same level of sample complexity, one would expect the revised SSGP to achieve better performance since SSGP generally performs better when its sample complexity is set closer to the required threshold. On the larger GAS SENSOR dataset (which contains approximately 4M data points), we also observe the same phenomenon from the performance comparison graph as shown in Figure 3.2a below: A vanilla SSGP needs to increase its number of samples to marginally improve its predictive performance while our revisited SSGP is able to outperform the former with the least number of samples ($p = 16$).



(a) (b) (c)

Figure 3.2: Graphs of (a) performance comparison between our revisited SSGP's (with sample complexity $p = 16$) and the vanilla SSGP's (with sample complexity $p = 16, 32, 64$) on the GAS SENSOR dataset (Burgues); and visualizations of (b) original and (c) reconfigured data distributions of GAS SENSOR data on a 2-dimensional latent space generated by our auto-encoding algorithm in Section 3.1.3.3. Additional results on this dataset are listed in Figures 3.3 and 3.4.

Furthermore, a closer look into the data distribution (visualized in a 2D space in Figure 3.2b) and the data reconfigured data distribution (visualized in a 2D space in Figure 3.2c) also corroborates our hypothesis earlier that a well-separated data partition with high in-cluster concentration (in the form of a mixture of clusters – see Condition **1**) can be found (by our embedding algorithm in Section 3.1.3.3) to reconfigure our data distribution to (approximately) meet the necessary technical conditions that enable our sample-complexity enhancement analysis (see Section 3.1.3.1).

The following ablation studies will list additional results on data visualization and performance comparison on GAS SENSOR data for supporting the discussed results.

Ablation: The Effect of Data Re-configuration

This section describes an ablation study to demonstrate the effectiveness of our data reconfiguration component (i.e., to approximately meet the practical Conditions **1**-**3** of our refined analysis). Specifically, we demonstrate this by contrasting the scatter plots of data embeddings (see Figure 3.3) before and after reconfiguration using our algorithm in Section 3.1.3.3 below. The visualizations are shown for 3 different samples of data, each of which has 10K data points.

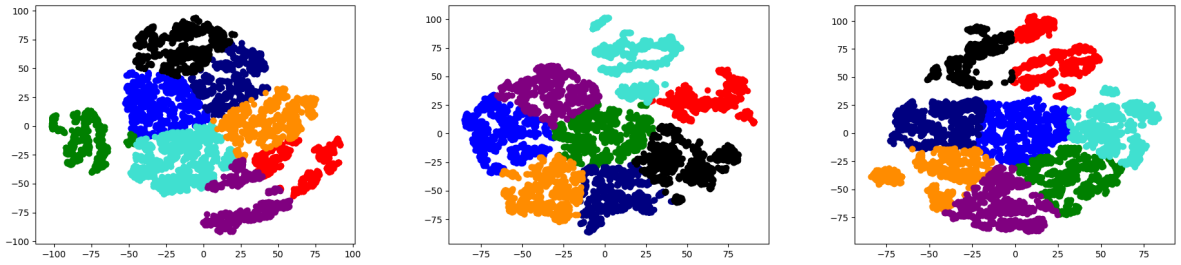
For each data sample, its embedding was clustered and re-clustered before and after its reconfiguration. Both clustering processes were generated independently using K-Means to provide an objective visual measurement of the reconfiguration effects of our algorithm.

Observing the above visual excerpts, it appears that after reconfiguration, the clusters across different data samples all became significantly more disentangled with a visibly increased distance between their cluster centers. This provides conclusive evidence of the data disentangling effect of our embedding algorithm. More importantly, this demonstration further reveals a practical aspect of data that has not been investigated before in the existing literature on GP.

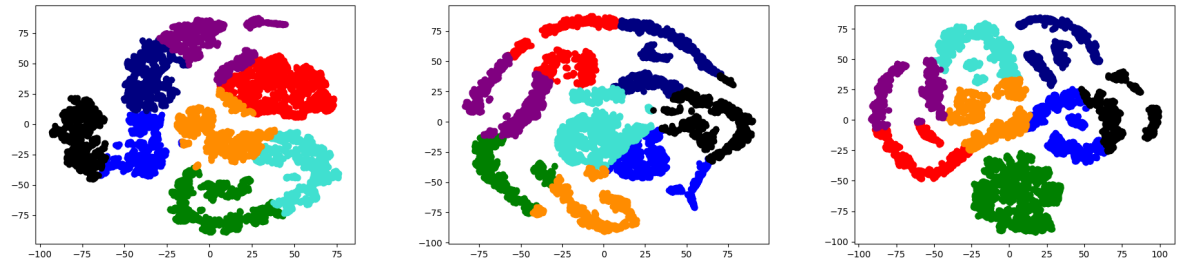
Data (especially experimental data) is often the manifestation of how latent concepts that underlie them were observed and depending on specific parameters of the observation process, these concepts might manifest differently in either more or less useful forms for learning. This raises the question of whether one can reorient the observation process to increase the utility of such data.

In this vein of thought, to address the above question, our data reconfiguration algorithm can be considered to be one potential solution that uses a parameterized construction of a latent space to provide a handle on how to reorient the latent concepts that underlie our data. For an intuitive example, imagine how we would look at the outside world via a narrowed pigeonhole. With different viewing angles, we would perceive the same scene outside differently and apparently, some angles provide a much better perception of that scene (thus, allowing us to interpret the scene more accurately).

In technical terms, such a reorientation is implemented in our algorithm via the regularization of the mixture composition of the latent prior while constraining the entire embedding process to have it reflected on the latent posterior – see Equation 3.50 – which was used to encode data into a latent space that exhibits the desired separation effect. Such separation/disentanglement is then shown (empirically) to be richer in information and can be leveraged to improve the sample



(S1): Original Embedding (S2): Original Embedding (S3): Original Embedding



(S1): ReconFigure Embedding (S2): ReconFigure Embedding (S3): ReconFigure Embedding

Figure 3.3: Visualizations of original (top) and reconfigured (bottom) data embeddings for 3 different (randomly selected) data samples annotated with S1, S2 and S3, respectively. Each visual excerpt is annotated with different colors corresponding to the different clusters that the data belong to. All visualizations are generated using t-SNE (van der Maaten and Hinton, 2008).

complexity of SSGP (see the ablation study below).

Ablation: Performance Comparison with SSGP on Large Data

To demonstrate the effectiveness of the data disentanglement in reducing the sample complexity of SSGP, we compare the performance of SSGP and our revisited SSGP (which was instead applied on the reconfigured space of data) at different levels of sample complexity. All results were generated for two different data samples extracted from GAS-SENSOR (Burgues). One of these (containing 500K data points) is in fact on the same scale of the most extensive datasets used in the GP literature. All performance plots were visualized in Figure 3.4. For each experiment, the data sample is divided into a train/test partition with an 8-2 ratio. All results were averaged over 5 independent runs.

We see that our revised SSGP consistently achieves better performance than its SSGP counterpart at all complexity levels. In particular, in all cases of the 10K setting, the performance of our revised SSGP is also shown to approach closely that of the full GP, which serves as a gold-standard lower-bound on the achievable prediction error. This concludes our empirical demonstration which (we believe) has shown that with a proper reconfiguration of data, the predictive

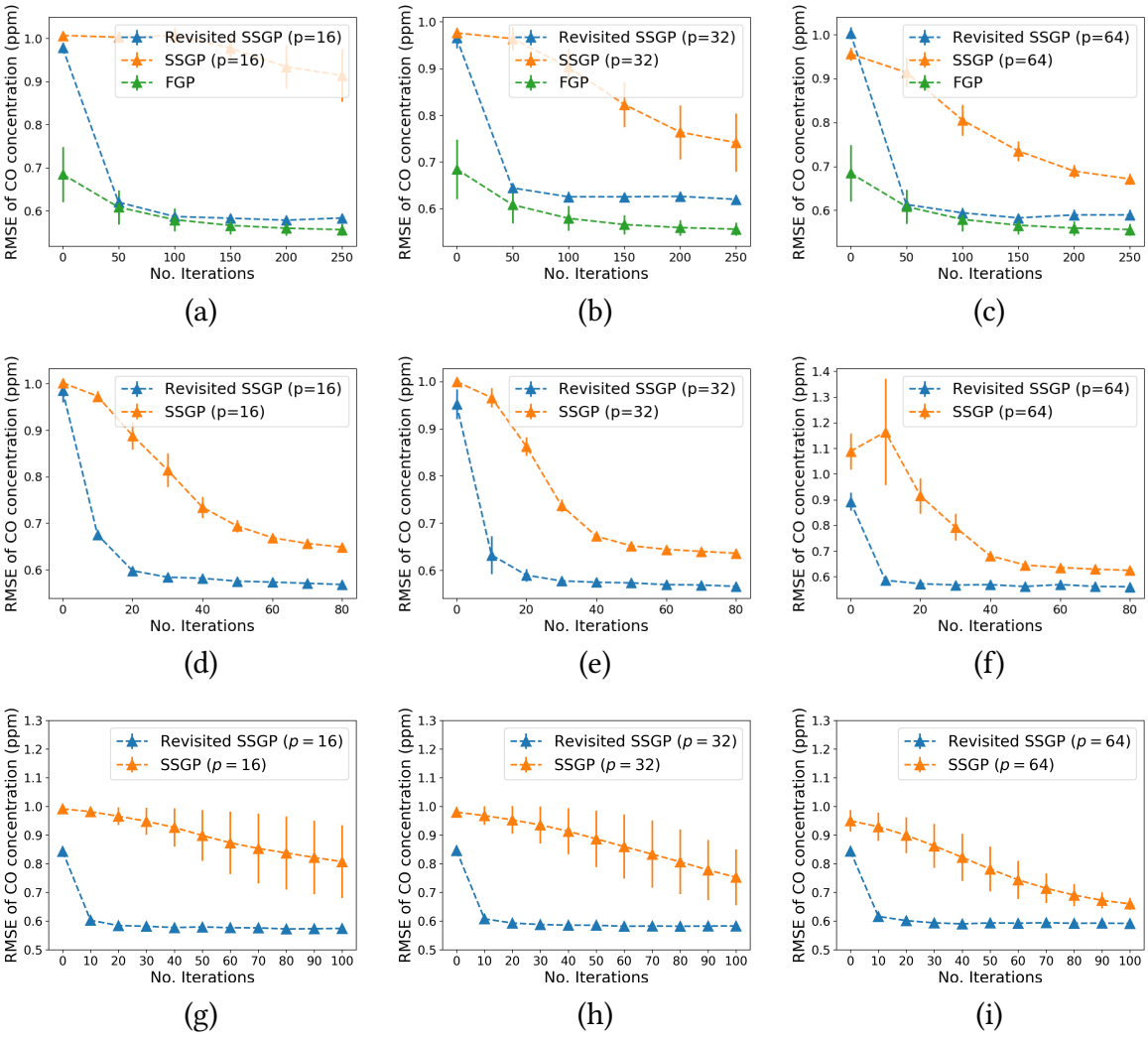


Figure 3.4: Graphs of performance comparisons between the full GP, our revised SSGP and the traditional SSGP on a 10K sample (a-c); 500K sample (d-f) and the entire GAS SENSOR dataset (Burgues) totaling approximately 4M data points (g-i). In both settings, the performance differences were plotted at $p = 16$, 32 and 64. Note that for the 500K sized sample and the entire dataset (which contains 4M data points), the full GP model is not applicable due to its inability (memory- and computation-wise) to store and invert the corresponding large covariance matrix.

performance of a GP can be well-preserved at a much cheaper sample complexity as compared to the previous conservative estimate yielded by SSGP. In fact, the performance trend of SSGP as depicted in the above graphs shows that with more samples, it also slowly converges towards the performance level of GP and our revised SSGP but at a much greater sample complexity – see the shrinking performance gap between revisited SSGP and SSGP from Figure 3.4d to Figure 3.4e; and similarly, from Figure 3.4g to Figure 3.4h.

3.1.5 Conclusion

We present a new method and analysis for approximating Gaussian processes. We obtain provable guarantees for both training and inference, which are the first to hold simultaneously over the entire space of kernel parameters. Our results complement existing work in kernel approximation that often assumes knowledge of its defining parameters. Our results also reveal important (practical) insights that allow us to develop an algorithmic handle on the tradeoff between approximation quality and sample complexity, which is achieved via finding an embedding that disentangles the latent coordinates of data. Our empirical results show for many datasets, such a disentangled embedding space can be found, which leads to a significantly reduced sample complexity of SSGP.

3.2 Approximate Matrix Trace Estimation

Similar to Gausian Processes in the previous section, Matrix trace estimation is ubiquitous in machine learning applications, especially when the data too large to realize the full matrix in the memory. Matrix trace estimation has traditionally relied on Hutchinson's method, which requires $O(\log(1/\delta)/\epsilon^2)$ matrix-vector product queries to achieve a $(1\pm\epsilon)$-multiplicative approximation to $\mathrm{tr}(A)$ with failure probability δ on positive-semidefinite input matrices A. Recently, the Hutch++ algorithm was proposed, which reduces the number of matrix-vector queries from $O(1/\epsilon^2)$ to the optimal $O(1/\epsilon)$, and the algorithm succeeds with constant probability. However, in the high probability setting, the non-adaptive Hutch++ algorithm suffers an extra $O(\sqrt{\log(1/\delta)})$ multiplicative factor in its query complexity. Non-adaptive methods are important, as they correspond to sketching algorithms, which are mergeable, highly parallelizable, and provide low-memory streaming algorithms as well as low-communication distributed protocols. In this work, we close the gap between non-adaptive and adaptive algorithms, showing that even non-adaptive algorithms can achieve $O\left(\sqrt{\log(1/\delta)}/\epsilon + \log(1/\delta)\right)$ matrix-vector products. In addition, we prove matching lower bounds demonstrating that, up to a $\log\log(1/\delta)$ factor, no further improvement in the dependence on δ or ϵ is possible by any non-adaptive algorithm. Finally, our experiments demonstrate the superior performance of our sketch over the adaptive Hutch++ algorithm, which is less parallelizable, as well as over the non-adaptive Hutchinson's method.

3.2.1 Problem and Motivation

The problem of implicit matrix trace estimation arises naturally in a wide range of applications (Ubaru and Saad, 2018). The popular applications of implicit trace estimation include counting triangles and computing the Estrada Index in graphs (Avron, 2010; Estrada and Hatano, 2008), approximating the generalized rank of a matrix (Zhang et al., 2015), and studying non-convex loss landscapes from the Hessian matrix of large neural networks (NNs) (Ghorbani et al., 2019; Yao et al., 2020), where it is almost always impossible to store the whole Hessian matrix or NNs that have many million to billion parameters.

To define the problem, we consider the *matrix-vector product model* as formalized in (Fika and Koukouvinos, 2017; Rashtchian et al., 2020; Sun et al., 2021), where there is a real symmetric input matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ that cannot be explicitly presented but one has oracle access to $\boldsymbol{A}$ via matrix-vector queries, i.e., one can obtain $\boldsymbol{A}\mathbf{q}$ for any desired query vector $\mathbf{q} \in \mathbb{R}^n$. For example, due to a tremendous amount of trainable parameters of large NNs, it is often prohibitive to compute or store the entire Hessian matrix $\boldsymbol{H}$ with respect to some loss function from the parameters (Ghorbani et al., 2019), which is often used to study the non-convex loss landscape. However, with Pearlmutter's trick (Pearlmutter, 1994) one can compute $\boldsymbol{H}\mathbf{q}$ for any chosen vector $\mathbf{q}$. The goal is to efficiently estimate the trace of $\boldsymbol{A}$, denoted by $\mathrm{tr}(\boldsymbol{A})$, up to ϵ error, i.e., to compute a quantity within $(1\pm\epsilon)\mathrm{tr}(\boldsymbol{A})$. For efficiency, such algorithms are randomized and succeed with probability

at least $1-\delta$. The minimum number of queries q required to solve the problem is referred to as the *query complexity*.

Computing matrix-vector products $\boldsymbol{A}\mathbf{q}$ through oracle access, however, can be costly. For example, computing Hessian-vector products $\boldsymbol{H}\mathbf{q}$ on large NNs takes approximately twice the time of backpropagation. When estimating the eigendensity of $\boldsymbol{H}$, one computes $\mathrm{tr}(f(\boldsymbol{H}))$ for some density function f, and needs repeated access to the matrix-vector product oracle. As a result, even with Pearlmutter's trick and distributed computation on modern GPUs, it takes 20 hours to compute the eigendensity of a single Hessian $\boldsymbol{H}$ with respect to the cross-entropy loss on the CIFAR-10 dataset (Krizhevsky et al., 2009), from a set of fixed weights for ResNet-18 (He et al., 2016) which has approximately 11 million parameters (Ghorbani et al., 2019). Thus, it is important to understand the fundamental limits of implicit trace estimation as the query complexity in terms of the desired approximation error ϵ and the failure probability δ.

Hutchinson's method (Hutchinson, 1989), a simple yet elegant randomized algorithm, is the ubiquitous workforce for implicit trace estimation. Letting $\boldsymbol{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_q] \in \mathbb{R}^{n\times q}$ be q vectors with i.i.d. Gaussian or Rademacher (i.e., ± 1 with equal probability) random variables, Hutchinson's method returns an estimate of $\mathrm{tr}(\boldsymbol{A})$ as $\frac{1}{q}\sum_{i=1}^{q}\mathbf{q}_i^\top \boldsymbol{A}\mathbf{q}_i = \frac{1}{q}\mathrm{tr}(\boldsymbol{Q}^\top\boldsymbol{A}\boldsymbol{Q})$. Although Hutchinson's method dates back to 1990, it is surprisingly not well-understood on positive semi-definite (PSD) matrices. It was originally shown that for PSD matrices $\boldsymbol{A}$ with the $\mathbf{q}_i$ being Gaussian random variables, in order to obtain a multiplicative $(1\pm\epsilon)$ approximation to $\mathrm{tr}(\boldsymbol{A})$ with probability at least $1-\delta$, $O(\log(1/\delta)/\epsilon^2)$ matrix-vector queries suffice (Roosta-Khorasani and Ascher, 2015).

A recent work (Meyer et al., 2020) proposes a variance-reduced version of Hutchinson's method that shows only $O(1/\epsilon)$ matrix-vector queries are needed to achieve a $(1\pm\epsilon)$-approximation to any PSD matrix with constant success probability, in contrast to the $O(1/\epsilon^2)$ matrix-vector queries needed for Hutchinson's original method. The key observation is that the variance of the estimated trace in Hutchinson's method is largest when there is a large gap between the top few eigenvalues and the remaining ones. Thus, by splitting the number of matrix-vector queries between approximating the top $O(1/\epsilon)$ eigenvalues, i.e., by computing a rank-$O(1/\epsilon)$ approximation to $\boldsymbol{A}$, and performing trace estimation on the remaining part of the spectrum, one needs only $O(1/\epsilon)$ queries in total to achieve a $(1\pm\epsilon)$ approximation to $\mathrm{tr}(\boldsymbol{A})$. Furthermore, Meyer et al. (2020) shows $\Omega(1/\epsilon)$ queries are in fact necessary for *any* trace estimation algorithm, up to a logarithmic factor, for algorithms succeeding with constant success probability. While Meyer et al. (2020) mainly focuses on the improvement on ϵ in the query complexity with constant failure probability, we focus on the dependence on the failure probability δ.

Achieving a low failure probability δ is important in applications where failures are highly undesirable, and the low failure probability regime is well-studied in related areas such as compressed sensing (Gilbert et al., 2013), data stream algorithms (Jayram and Woodruff, 2011; Kamath et al., 2021), distribution testing (Diakonikolas et al., 2020), and so on. While one can always reduce the failure probability from a constant to δ by performing $O(\log(1/\delta))$ independent repetitions and taking the median, this multiplicative overhead of $O(\log(1/\delta))$ can cause a huge

slowdown in practice, e.g., in the examples above involving large Hessians.

Algorithm 1 Hutch++: Stochastic trace estimation with **adaptive** matrix-vector queries

Input: Matrix-vector multiplication oracle for PSD matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$. Number m of queries.
Output: Approximation to $\mathrm{tr}(\boldsymbol{A})$.

1: Sample $\boldsymbol{S} \in \mathbb{R}^{n\times \frac{m}{3}}$ and $\boldsymbol{G} \in \mathbb{R}^{n\times \frac{m}{3}}$ with i.i.d. $\mathcal{N}(0,1)$ entries.
2: Compute an orthonormal basis $\boldsymbol{Q} \in \mathbb{R}^{n\times \frac{m}{3}}$ for the span of $\boldsymbol{AS}$ via $\boldsymbol{QR}$ decomposition.
3: **Return** $t = \mathrm{tr}(\boldsymbol{Q}^\top \boldsymbol{AQ}) + \frac{3}{m}\mathrm{tr}(\boldsymbol{G}^\top(\boldsymbol{I}-\boldsymbol{QQ}^\top)\boldsymbol{A}(\boldsymbol{I}-\boldsymbol{QQ}^\top)\boldsymbol{G})$.

Algorithm 2 NA-Hutch++: Stochastic trace estimation with **non-adaptive** matrix-vector queries

Input: Matrix-vector multiplication oracle for PSD matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$. Number m of queries.
Output: Approximation to $\mathrm{tr}(\boldsymbol{A})$.

1: Fix constants c_1, c_2, c_3 such that $c_1 < c_2$ and $c_1 + c_2 + c_3 = 1$.
2: Sample $\boldsymbol{S} \in \mathbb{R}^{n\times c_1 m}$, $\boldsymbol{R} \in \mathbb{R}^{n\times c_2 m}$, and $\boldsymbol{G} \in \mathbb{R}^{n\times c_3 m}$, with i.i.d. $\mathcal{N}(0,1)$ entries.
3: $\boldsymbol{Z} = \boldsymbol{AR}$, $\boldsymbol{W} = \boldsymbol{AS}$
4: **Return** $t = \mathrm{tr}((\boldsymbol{S}^\top\boldsymbol{Z})^\dagger(\boldsymbol{W}^\top\boldsymbol{Z})) + \frac{1}{c_3 m}(\mathrm{tr}(\boldsymbol{G}^\top\boldsymbol{AG}) - \mathrm{tr}(\boldsymbol{G}^\top\boldsymbol{Z}(\boldsymbol{S}^\top\boldsymbol{Z})^\dagger\boldsymbol{W}^\top\boldsymbol{G}))$.

Two algorithms were proposed in (Meyer et al., 2020): Hutch++ (**Algorithm** 1), which requires *adaptively* chosen matrix-vector queries and NA-Hutch++ (**Algorithm** 2) [8] that only requires *non-adaptively* chosen queries. We call the matrix-vector queries adaptively chosen if subsequent queries are dependent on previous queries $\mathbf{q}$ and observations $\boldsymbol{A}\mathbf{q}$, whereas the algorithm is non-adaptive if all queries can be chosen at once without any prior information about $\boldsymbol{A}$. Note that Hutchinson's method uses only non-adaptive queries. Meyer et al. (2020) shows that Hutch++ can use $O(\sqrt{\log(1/\delta)}/\epsilon + \log(1/\delta))$ adaptive matrix-vector queries to achieve $(1\pm\epsilon)$ approximation with probability at least $1-\delta$, while NA-Hutch++ can use $O(\log(1/\delta)/\epsilon)$ non-adaptive queries. Thus, in many parameter regimes the non-adaptive algorithm suffers an extra $\sqrt{\log(1/\delta)}$ multiplicative factor over the adaptive algorithm.

It is important to understand the query complexity of non-adaptive algorithms for trace estimation because the advantages of non-adaptivity are plentiful: algorithms that require only non-adaptive queries can be easily parallelized across multiple machines while algorithms with adaptive queries are inherently sequential. Furthermore, non-adaptive algorithms correspond to sketching algorithms which are the basis for many streaming algorithms with low memory (Muthukrishnan, 2005) or distributed protocols with low-communication overhead (for an example application to low rank approximation, see (Boutsidis et al., 2016)). We note that there are numerous works on estimating matrix norms in a data stream (Braverman et al., 2018, 2020; Li et al., 2014; Li and Woodruff, 2016), most of which use trace estimation as a subroutine.

[8] $\dagger$ denotes the Moore-Penrose pseudoinverse.

3.2.2 Related Work

Matrix Trade Estimation A summary of prior work on the query complexity of trace estimation of PSD matrices is given in **Table** 3.1. For the upper bounds, prior to the work of (Avron and Toledo, 2011), the analysis of implicit trace estimation mainly focused on the variance of estimation with different types of query vectors. Avron and Toledo (2011) gave the first upper bound on the query complexity. The work of Roosta-Khorasani and Ascher (2015) improved the bounds in (Avron and Toledo, 2011). On the lower bound side, although Roosta-Khorasani and Ascher (2015) gives a necessary condition on the query complexity for Gaussian query vectors, this condition does not directly translate to a bound on the minimum number of query vectors. The work of Meyer et al. (2020) gives the first lower bound on the query complexity in terms of ϵ but only works for constant failure probability.

Upper Bounds				
Prior Work	Query Complexity	Query Vector Type	Failure Probability	Algorithm Type
Avron and Toledo (2011)	$O(\log(1/\delta)/\epsilon^2)$	Gaussian	δ	non-adaptive
Avron and Toledo (2011)	$O(\log(\text{rank}(\boldsymbol{A})/\delta)/\epsilon^2)$	Rademacher	δ	non-adaptive
Roosta-Khorasani and Ascher (2015)	$O(\log(1/\delta)/\epsilon^2)$	Gaussian, Rademacher	δ	non-adaptive
Meyer et al. (2020)	$O(\sqrt{\log(1/\delta)}/\epsilon + \log(1/\delta))$	Gaussian, Rademacher	δ	adaptive
Meyer et al. (2020)	$O(\log(1/\delta)/\epsilon)$	Gaussian, Rademacher	δ	non-adaptive
This Work	$O(\sqrt{\log(1/\delta)}/\epsilon + \log(1/\delta))$	Gaussian	δ	non-adaptive
Lower Bounds				
Meyer et al. (2020)	$\Omega(1/(\epsilon\log(1/\epsilon)))$	–	constant	adaptive
Meyer et al. (2020)	$\Omega(1/\epsilon)$	–	constant	non-adaptive
This Work	$\Omega(\sqrt{\log(1/\delta)}/\epsilon + \frac{\log(1/\delta)}{\log\log(1/\delta)})$	–	δ	non-adaptive

Table 3.1: Upper and lower bounds on the query complexity for trace estimation of PSD matrices.

Subspace Embedding A closely related topic is subspace embedding that has broad applications in graph algorithms, optimization, and machine learning in general. One of the most seminal results is from Johnson and Lindenstrauss (1984) where it is proven that a subspace mapping that preserve pair-wise distances can be done in randomized polynomial time. Recently, Sobczyk and Luisier (2022) further improves to even tighter bounds on a lower dimension of the embedded space for any given matrix regardless of spectrum, also by relying on **Algorithm** 1).

3.2.3 Our Contributions

Improving the Non-adaptive Query Complexity. We give an improved analysis of the query complexity of the non-adaptive trace estimation algorithm `NA-Hutch++` (**Algorithm** 2), based on a new low-rank approximation algorithm and analysis in the high probability regime, instead of applying an off-the-shelf low-rank approximation algorithm as in (Meyer et al., 2020). Instead of $O(\log(1/\delta)/\epsilon)$ queries as shown in (Meyer et al., 2020), we show that $O(\sqrt{\log(1/\delta)}/\epsilon + \log(1/\delta))$ non-adaptive queries suffice to achieve a multiplicative $(1 \pm \epsilon)$ approximation of the

trace with probability at least $1-\delta$, which matches the query complexity of the adaptive trace estimation algorithm Hutch++. Since our algorithm is non-adaptive, it can be used in subroutines in streaming and distributed settings for estimating the trace, with lower memory than was previously possible for the same failure probability.

Theorem 3.2.1 (Restatement of Theorem 3.2.5)**.** *Let $\boldsymbol{A}$ be any PSD matrix. If NA-Hutch++ is implemented with*

$$m = O\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon} + \log(1/\delta)\right)$$

matrix-vector multiplication queries, then with probability $1-\delta$, the output t of NA-Hutch++ satisfies $(1-\epsilon)tr(\boldsymbol{A}) \leq t \leq (1+\epsilon)tr(\boldsymbol{A})$.

The improved dependence on δ is perhaps surprising in the non-adaptive setting, as simply repeating a constant-probability algorithm would give an $O(\log(1/\delta)/\epsilon)$ dependence. Our non-adaptive algorithm is as good as the best-known adaptive algorithm, and much better than previous non-adaptive algorithms (Hutchinson, 1989; Meyer et al., 2020). The key difference between our analysis and the analysis in (Meyer et al., 2020) is in the number of non-adaptive matrix-vector queries we need to obtain an $O(1)$-approximate rank-k approximation to $\boldsymbol{A}$ in Frobenius norm.

Specifically, to reduce the total number of matrix-vector queries, our queries are split between (1) computing $\tilde{\boldsymbol{A}}$, a rank-k approximation to the matrix $\boldsymbol{A}$, and (2) performing trace estimation on $\boldsymbol{A}-\tilde{\boldsymbol{A}}$. Let $\boldsymbol{A}_k = \min_{\text{rank-}k\ \boldsymbol{A}} \|\boldsymbol{A}-\boldsymbol{A}_k\|_F$ be the best rank-k approximation to $\boldsymbol{A}$ in Frobenius norm. For our algorithm to work, we require $\|\boldsymbol{A}-\tilde{\boldsymbol{A}}\| \leq O(1)\|\boldsymbol{A}-\boldsymbol{A}_k\|_F$ with probability $1-\delta$. Previous results from Clarkson and Woodruff (2009) show the number of non-adaptive queries required to compute $\tilde{\boldsymbol{A}}$ is $O(k\log(1/\delta))$, where each query is an i.i.d. Gaussian or Rademacher vector. We prove $O(k+\log(1/\delta))$ non-adaptive Gaussian query vectors suffice to compute $\tilde{\boldsymbol{A}}$. Low rank approximation requires both a so-called subspace embedding and an approximate matrix product guarantee (see, e.g., (Woodruff, 2014), for a survey on sketching for low-rank approximation), and we show both hold with the desired probability, with some case analysis, for Gaussian queries. A technical overview can be found in Section 3.2.5.

The improvement in the number of non-adaptive queries to achieve $O(1)$-approximate rank-k approximation has many other implications, which can be of independent interest. For example, since low-rank approximation algorithms are extensively used in streaming algorithms suitable for low-memory settings, this new result directly improves the space complexity of the state-of-the-art streaming algorithm for Principle Component Analysis (PCA) (Boutsidis et al., 2016) from $O(d\cdot(k\log(1/\delta)))$ to $O(d\cdot(k+\log(1/\delta)))$ for constant approximation error ϵ, where d is the dimension of the input.

Lower Bound. Previously, no lower bounds were known on the query complexity in terms of δ in a high probability setting. In this work, we give a novel matching lower bound for non-adaptive (i.e., sketching) algorithms for trace estimation, with novel techniques based on a new

family of hard input distributions, showing that our improved $O(\sqrt{\log(1/\delta)}/\epsilon + \log(1/\delta))$ upper bound is optimal, up to a $\log\log(1/\delta)$ factor, for any $\epsilon \in (0,1)$. The methods previously used to prove an $\Omega(1/\epsilon)$ lower bound with constant success probability (up to logarithmic factors) in (Meyer et al., 2020) do not apply in the high probability setting. Indeed, Meyer et al. (2020) gives two lower bound methods based on a reduction from two types of problems: (1) a communication complexity problem, and (2) a distribution testing problem between clean and negatively spiked random covariance matrices. Technique (1) does not apply since there is not a multi-round lower bound for the Gap-Hamming communication problem used in (Meyer et al., 2020) that depends on δ. One might think that since we are proving a non-adaptive lower bound, we could use a non-adaptive lower bound for Gap-Hamming (which exists, see (Jayram and Woodruff, 2011)), but this is wrong because even the non-adaptive lower bound in (Meyer et al., 2020) uses a 2-round lower bound for Gap-Hamming, and there is no such lower bound known in terms of δ. Technique (2) also does not apply, as it involves a $1/\epsilon \times 1/\epsilon$ matrix, which can be recovered exactly with $1/\epsilon$ queries; further, increasing the matrix dimensions would break the lower bound as their two cases would no longer need to be distinguished. Thus, such a hard input distribution fails to show the additive $\Omega(\log(1/\delta))$ term in the lower bound.

Our starting point for a hard instance is a family of Wigner matrices (see Definition 3.2.3) shifted by an identity matrix so that they are PSD. However, due to the strong concentration properties of these matrices, they can only be used to provide a lower bound of $\Omega(\sqrt{\log(1/\delta)}/\epsilon)$ when $\epsilon < 1/\sqrt{\log(1/\delta)}$. Indeed, setting δ to be a constant, in this case, recovers the $\Omega(1/\epsilon)$ lower bound shown in (Meyer et al., 2020) but via a completely different technique. For larger ϵ, we consider a new distribution testing problem between clean Wigner matrices and the same distribution with a large rank-1 noisy PSD matrix and then argue with probability roughly δ, all non-adaptive queries have unusually tiny correlation with this rank-1 matrix, thus making it indistinguishable between the two distributions. This gives the desired additive $\Omega(\log(1/\delta))$ lower bound, up to a $\log\log(1/\delta)$ factor.

Theorem 3.2.2 (Restatement of Theorem 3.2.11)**.** *Suppose $\mathcal{A}$ is a non-adaptive query-based algorithm that returns a $(1\pm\epsilon)$-multiplicative estimate to $\mathrm{tr}(\boldsymbol{A})$ for any PSD matrix $\boldsymbol{A}$ with probability at least $1-\delta$. Then, the number of matrix-vector queries must be at least*

$$m = \Omega\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon} + \frac{\log(1/\delta)}{\log(\log(1/\delta))}\right).$$

3.2.4 Problem Setting

Notation. A matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is symmetric positive semi-definite (PSD) if it is real, symmetric, and has non-negative eigenvalues. Hence, $x^\top \mathbf{A} x \geq 0$ for all $x \in \mathbb{R}^n$. Let $\mathrm{tr}(\boldsymbol{A}) = \sum_{i=1}^n \boldsymbol{A}_{ii}$ denote the trace of $\boldsymbol{A}$. Let $\|\boldsymbol{A}\|_F = (\sum_{i=1}^n \sum_{j=1}^n \boldsymbol{A}_{ij}^2)^{1/2}$ denote the Frobenius norm and $\|\boldsymbol{A}\|_{op} =$

$\sup_{\|\mathbf{v}\|_2=1} \|\boldsymbol{A}\mathbf{v}\|_2$ denote the operator norm of $\boldsymbol{A}$. Let $\mathcal{N}(\mu, \sigma^2)$ denote the Gaussian distribution with mean μ and variance σ^2. Our analysis extensively relies on the following facts:

Definition 3.2.3 (Gaussian and Wigner Random Matrices)**.** *We let $\boldsymbol{G} \sim \mathcal{N}(n)$ denote an $n \times n$ random Gaussian matrix with i.i.d. $\mathcal{N}(0,1)$ entries. We let $\boldsymbol{W} \sim \mathcal{W}(n) = \boldsymbol{G} + \boldsymbol{G}^\top$ denote an $n \times n$ Wigner matrix, where $\boldsymbol{G} \sim \mathcal{N}(n)$.*

Fact 3.2.1 (χ^2 Tail Bound (**Lemma 1** of (Laurent and Massart, 2000)))**.** *Let $Z \sim \chi^2(n)$. Then for any $x > 0$,*

$$\Pr[Z \geq n + 2\sqrt{nx} + 2x] \leq e^{-x}$$
$$\Pr[Z \leq n - 2\sqrt{nx}] \leq e^{-x}.$$

Fact 3.2.2 (Rotational Invariance of a standard Gaussian)**.** *Let $\boldsymbol{R} \in \mathbb{R}^{n \times n}$ be an orthogonal matrix. Let $\mathbf{g} \in \mathbf{R}^n$ be a random vector with i.i.d. $\mathcal{N}(0,1)$ entries. Then $\boldsymbol{R}\mathbf{g}$ has the same distribution as $\mathbf{g}$.*

Fact 3.2.3 (Upper Gaussian Tail Bound)**.** *Let $Z \sim \mathcal{N}(0, \sigma^2)$ be a univariate Gaussian random variable. Then for any $t > 0$,*

$$\Pr[Z \geq t] \leq \exp(-\frac{t^2}{2\sigma^2}).$$

Fact 3.2.4 (Lower Gaussian Tail Bound)**.** *Letting $Z \sim \mathcal{N}(0,1)$ be a univariate Gaussian random variable, for any $t > 0$,*

$$\Pr[Z \geq t] \geq \frac{1}{\sqrt{2\pi}} \cdot \frac{1}{t} \exp(t^2/2).$$

Lemma 3.2.4 (Concentration of Singular Values of a Gaussian Random Matrix (**Eq. 2.3** of (Rudelson and Vershynin, 2010)))**.** *Let $\boldsymbol{G} \sim \mathcal{N}(n)$, and $s_{max}(\boldsymbol{G})$ denote the maximum singular value of $\boldsymbol{G}$. Then $\forall t \geq 0$,*

$$\Pr[s_{max}(\boldsymbol{G}) \leq 2\sqrt{n} + t] \geq 1 - 2\exp(-t^2/2)$$

Fact 3.2.5 (KL Divergence Between Multivariate Gaussian Distributions (**Eq. 8** of (Soch and Allefeld, 2016), or Section 9 of (Duchi))**.** *Let $\mathcal{P} \sim \mathcal{N}(\mu_1, \boldsymbol{\Sigma}_1)$ and $\mathcal{Q} \sim \mathcal{N}(\mu_2, \boldsymbol{\Sigma}_2)$ be two k-dimensional multivariate normal distributions. The Kullback-Leibler divergence between $\mathcal{P}$ and $\mathcal{Q}$ is*

$$\mathcal{D}_{KL}(\mathcal{P} \parallel \mathcal{Q}) = \frac{1}{2}\Big\{(\mu_2 - \mu_1)^T \boldsymbol{\Sigma}_2^{-1}(\mu_2 - \mu_1) + tr(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1) - \ln \frac{\det(\boldsymbol{\Sigma}_1)}{\det(\boldsymbol{\Sigma}_2)} - k\Big\}.$$

Fact 3.2.6 (Conditioning Increases KL Divergence ((Wu, 2020))). *Let $\mathcal{P}_{Y|X}$, $\mathcal{Q}_{Y|X}$ be two conditional probability distributions over spaces $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$, let $\mathcal{P}_Y = \mathcal{P}_{Y|X}\mathcal{P}_X$ and $\mathcal{Q}_Y = \mathcal{Q}_{Y|X}\mathcal{P}_X$. Then,*

$$\mathcal{D}_{KL}(\mathcal{P}_Y \parallel \mathcal{Q}_Y) \leq \mathcal{D}_{KL}(\mathcal{P}_{Y|X} \parallel \mathcal{Q}_{Y|X} \mid \mathcal{P}_X) := \int \mathcal{D}_{KL}(\mathcal{P}_{Y|X=x} \parallel \mathcal{Q}_{Y|X=x}) d\mathcal{P}_X.$$

Fact 3.2.7 (KL Divergence Data Processing Inequality (Page 18 of (Duchi, 2021))). *For any function f and random variables X and Y on the same probability space, it holds that*

$$\mathcal{D}_{KL}(f(X) \parallel f(Y)) \leq \mathcal{D}_{KL}(X \parallel Y).$$

3.2.5 An Improved Analysis of NA-Hutch++

Suppose we are trying to compute a sketch so as to estimate the trace of a matrix $\boldsymbol{A}$ up to a $(1 \pm \epsilon)$-factor with success probability at least $1 - \delta$. Note that we focus on the case where we make matrix-vector queries *non-adaptively*. For any algorithm that accomplishes this with a small constant failure probability, one can simply repeat this procedure $O(\log(1/\delta))$ times to amplify the success probability to $1 - \delta$. Since these queries are non-adaptive and must be presented before any observations are made, it seems intuitive that the number of non-adaptive queries of NA-Hutch++ (**Algorithm** 2) should be $O(\log(1/\delta)/\epsilon)$ as shown in (Meyer et al., 2020). In this section, we give a proof sketch as to why this can be reduced to $O(\sqrt{\log(1/\delta)}/\epsilon + \log(1/\delta))$ as stated in our main **Theorem** 3.2.5.

Theorem 3.2.5. *Let $\boldsymbol{A}$ be a PSD matrix. If NA-Hutch++ is implemented with*

$$m = O\left(\sqrt{\log(1/\delta)}/\epsilon + \log(1/\delta)\right)$$

matrix-vector multiplication queries, then with probability $1 - \delta$, the output of NA-Hutch++, denoted by t, satisfies $(1-\epsilon)tr(\boldsymbol{A}) \leq t \leq (1+\epsilon)tr(\boldsymbol{A})$.

3.2.5.1 Proof Sketch of Theorem 3.2.5

Recall that NA-Hutch++ splits its matrix-vector queries between computing an $O(1)$-approximate rank-k approximation $\widetilde{\boldsymbol{A}}$ and performing Hutchinson's estimate on the residual matrix $\boldsymbol{A} - \widetilde{\boldsymbol{A}}$. The key to an improved query complexity of NA-Hutch++ is on the analysis of the size of random Gaussian sketching matrices $\boldsymbol{S}$, $\boldsymbol{R}$ in Algorithm 2 that one needs to get an $O(1)$-approximate rank-k approximation $\widetilde{\boldsymbol{A}}$ in the Frobenius norm. To get the desired rank-k approximation, we need $\boldsymbol{S}$ and $\boldsymbol{R}$ to satisfy two properties: 1) subspace embedding as in **Lemma** 3.2.6 and 2) approximate matrix product for orthogonal subspaces as in **Lemma** 3.2.7. Specifically, we show

in **Lemma** 3.2.7 that choosing $\boldsymbol{S}$ and $\boldsymbol{R}$ to be of size $O(k + \log(1/\delta))$ suffices to get the second property with probability $1 - \delta$.

After that, we show in **Lemma** 3.2.8 that if a sketching matrix $\boldsymbol{S}$ satisfies the two properties mentioned above, with size $O(k + \log(1/\delta))$, one gets an $O(1)$-approximate low rank approximation with probability $1 - \delta$ when solving a sketched version of the regression problem $\min_{\boldsymbol{X}} \|\boldsymbol{S}^T(\boldsymbol{AX} - \boldsymbol{B})\|_F$ for fixed matrices $\boldsymbol{A}, \boldsymbol{B}$ with $\text{rank}(\boldsymbol{A}) = k$. **Lemma** 3.2.8 serves as an intermediate step to construct an $O(1)$-approximate rank-k approximation $\widetilde{\boldsymbol{A}}$ with $\boldsymbol{S}, \boldsymbol{R}$ having a size of only $O(k + \log(1/\delta))$ in **Theorem** 3.2.9.

Finally, we combine **Theorem** 3.2.10 from (Meyer et al., 2020), which shows the trade-off between the rank k and the number l spent on estimating the small eigenvalues, and **Theorem** 3.2.9, which shows the number of non-adaptive queries one needs to get a desired rank-k factor, to conclude in **Theorem** 3.2.5 that NA-Hutch++ needs only $O\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon} + \log(1/\delta)\right)$ non-adaptive queries, by setting $k = \frac{\sqrt{\log(1/\delta)}}{\epsilon}$.

3.2.5.2 Detailed Proof of Theorem 3.2.5

Lemma 3.2.6 (Subspace Embedding (Theorem 6 of (Woodruff, 2014))). *Given $\delta \in (0, \frac{1}{2})$ and $\epsilon \in (0, 1)$, let $\boldsymbol{S} \in \mathbb{R}^{r \times n}$ be a random matrix with i.i.d. Gaussian random variables $\mathcal{N}(0, \frac{1}{r})$. Then for any fixed d-dimensional subspace $\boldsymbol{A} \in \mathbb{R}^{n \times d}$, and for $r = O((d + \log(\frac{1}{\delta}))/\epsilon^2)$, the following holds with probability $1 - \delta$ simultaneously for all $x \in \mathbb{R}^d$,*

$$\|\boldsymbol{SA}x\|_2 = (1 \pm \epsilon)\|\boldsymbol{A}x\|_2.$$

Lemma 3.2.7 (Approximate Matrix Product for Orthogonal Subspaces). *Given $\delta \in (0, \frac{1}{2})$, let $\boldsymbol{U} \in \mathbb{R}^{n \times k}, \boldsymbol{W} \in \mathbb{R}^{n \times p}$ be two matrices with orthonormal columns such that $\boldsymbol{U}^T\boldsymbol{W} = 0$, $p \geq \max(k, \log(1/\delta))$, rank$(\boldsymbol{U}) = k$ and rank$(\boldsymbol{W}) = p$. Let $\boldsymbol{S} \in \mathbb{R}^{r \times n}$ be a random matrix with i.i.d. Gaussian random variables $\mathcal{N}(0, \frac{1}{r})$. For $r = O(k + \log(\frac{1}{\delta}))$, the following holds with probability $1 - \delta$,*

$$\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{SW}\|_F \leq O(1)\|\boldsymbol{W}\|_F.$$

Note that we will apply the above two lemmas with constant ϵ. The proof intuition is as follows: consider a sketch matrix $\boldsymbol{S}$ of size r with i.i.d. $\mathcal{N}(0, \frac{1}{r})$ random variables as in **Lemma** 3.2.7. The range of $\boldsymbol{U} \in \mathbb{R}^{n \times k}$ corresponds to an orthonormal basis of a rank-k low-rank approximation to $\boldsymbol{A}$, and the range of $\boldsymbol{W} \in \mathbb{R}^{n \times p}$ is the orthogonal complement. Note that both $\boldsymbol{SU}$ and $\boldsymbol{SW}$ are random matrices consisting of i.i.d. $\mathcal{N}(0, \frac{1}{r})$ random variables, and thus the task is to bound the size, in Frobenius norm, of the product of two random Gaussian matrices with high probability. Intuitively, the size of the matrix product is proportional to the rank k and inversely

proportional to our sketch size r. The overall failure probability δ, however, is inversely proportional to k, since as k grows, the matrix product involves summing over more squared Gaussian random variables, i.e., χ^2 random variables and thus becomes even more concentrated. We show that for $k \geq \log(1/\delta)$, a sketch size of $O(k)$ suffices since the failure probability for each χ^2 random variable is small enough to pay a union bound over k terms. On the other hand, when $k < \log(1/\delta)$, we show that $r = O(\log(1/\delta))$ suffices for the union bound. Combining the two cases gives $r = O(k + \log(1/\delta))$. The detailed proof is as below.

Proof. Let $\boldsymbol{G} = \sqrt{r}\boldsymbol{U}^T\boldsymbol{S}^T \in \mathbb{R}^{k\times r}$ and $\boldsymbol{H} = \sqrt{r}\boldsymbol{S}\boldsymbol{W} \in \mathbb{R}^{r\times p}$. Since both $\boldsymbol{U}$ and $\boldsymbol{W}$ have orthonormal columns, both $\boldsymbol{G}$ and $\boldsymbol{H}$ are random matrices with i.i.d. Gaussian random variables $\mathcal{N}(0,1)$. Furthermore, let $\mathbf{g}_i, \forall i \in [k]$ denote the i-th row of $\boldsymbol{G}$ and $\mathbf{h}_j, \forall j \in [p]$ denote the j-th column of $\boldsymbol{H}$.

$$\begin{aligned}
\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{W}\|_F^2 &= \|\frac{1}{\sqrt{r}}\boldsymbol{G}\frac{1}{\sqrt{r}}\boldsymbol{H}\|_F^2 \\
&= \frac{1}{r^2}\sum_{i=1}^{k}\sum_{j=1}^{p}\langle \mathbf{g}_i, \mathbf{h}_j\rangle^2 \\
&= \frac{1}{r^2}\sum_{i=1}^{k}\sum_{j=1}^{p}\|g_i\|_2^2\left\langle \frac{\mathbf{g}_i}{\|\mathbf{g}_i\|_2}, \mathbf{h}_j\right\rangle^2 \\
&= \frac{1}{r^2}\sum_{i=1}^{k}\|g_i\|_2^2\left(\sum_{j=1}^{p}\langle \frac{\mathbf{g}_i}{\|\mathbf{g}_i\|_2}, \mathbf{h}_j\rangle^2\right).
\end{aligned}$$

Since $\|\frac{\mathbf{g}_i}{\|\mathbf{g}\|_2}\|_2 = 1$, $\left\langle \frac{\mathbf{g}_i}{\|\mathbf{g}_i\|_2}, \mathbf{h}_j\right\rangle \sim \mathcal{N}(0,1)$. Thus,

$$\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{W}\|_F^2 = \frac{1}{r^2}\sum_{i=1}^{k}\mathbf{c}_i\cdot\mathbf{d}_i,$$

where $\mathbf{c}_i \sim \chi^2(r)$, $\mathbf{d}_i \sim \chi^2(p)$, $\forall i \in [k]$. Note that since $\boldsymbol{W}$ has orthonormal columns, $\|\boldsymbol{W}\|_F^2 = p$.

The number r of rows our random sketch matrix $\boldsymbol{S}$ needs in order to obtain an upper bound on the product of random Gaussian matrices $\boldsymbol{S}\boldsymbol{U}$ and $\boldsymbol{S}\boldsymbol{W}$, up to a constant factor of $\|\boldsymbol{W}\|_F$, depends on the concentration of $\boldsymbol{S}\boldsymbol{U}$ and $\boldsymbol{S}\boldsymbol{W}$. Specifically, to apply the χ^2 tail bound on some random variable $\mathbf{v} \sim \chi^2(d)$ from Fact 3.2.1 and to get that $\mathbf{v}$ concentrates around $O(1)d$ with probability $1-\delta$, the degree d needs to be at least $\log(1/\delta)$. Since we require $p = \text{rank}(\boldsymbol{W}) \geq \log(1/\delta)$, $\boldsymbol{S}\boldsymbol{W}$ is concentrated with high probability. The concentration of $\boldsymbol{S}\boldsymbol{U}$ depends on $\text{rank}(\boldsymbol{U}) = k$. To upper bound $\|(\boldsymbol{S}\boldsymbol{U})^T(\boldsymbol{S}\boldsymbol{W})\|_F$, we consider two cases for k:

Case I: Consider the case when $k \geq \log(\frac{1}{\delta})$:

Since $p \geq k \geq \log(\frac{1}{\delta})$, by **Fact** 3.2.1, $\forall i \in [k]$,

$$\Pr[\mathbf{d}_i \leq O(1)p] \geq 1 - e^{-O(k)}.$$

Since $r = O(k + \log(1/\delta))$, by **Fact** 3.2.1, $\forall i \in [k]$,

$$\Pr[\mathbf{c}_i \leq O(1)k] \geq 1 - e^{-O(k)}.$$

By a union bound over $2k$ χ^2 random variables,

$$\Pr\left[\sum_{i=1}^{k} \mathbf{c}_i \cdot \mathbf{d}_i \leq O(1)k^2 p\right] \geq 1 - 2k \cdot e^{-O(k)}.$$

Thus with probability $1 - O(\delta)$,

$$\begin{aligned}\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{W}\|_F^2 &= \frac{1}{r^2}\sum_{i=1}^{k} \mathbf{c}_i \cdot \mathbf{d}_i \\ &\leq \frac{1}{r^2}O(1)k^2 p \\ &= \frac{1}{r^2}O(1)k^2\|\boldsymbol{W}\|_F^2.\end{aligned}$$

And so $r = O(k + \log(1/\delta))$ gives $\|\boldsymbol{U}\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{W}\|_F \leq O(1)\|\boldsymbol{W}\|_F$ with probability $1 - \delta$.

Case II: Consider the case when $k < \log(\frac{1}{\delta})$.

Since $p \geq \log(\frac{1}{\delta})$, by **Fact** 3.2.1, $\forall i \in [k]$,

$$\Pr\left[\mathbf{d}_i \leq O(1)p\right] \geq 1 - e^{-O(\log(1/\delta))}.$$

Since $r = O(k + \log(1/\delta))$, by **Fact** 3.2.1, $\forall i \in [k]$,

$$\Pr\left[\mathbf{c}_i \leq O(1)\log(1/\delta)\right] \geq 1 - e^{-O(\log(1/\delta))}.$$

By a union bound over $2k$ χ^2 random variables, for $k < \log(1/\delta)$

$$\Pr\left[\sum_{i=1}^{k} \mathbf{c}_i \cdot \mathbf{d}_i \leq O(1)k\log(1/\delta)p\right] \geq 1 - 2k \cdot e^{-O(\log(1/\delta))}.$$

Thus with probability $1 - O(\delta)$,

$$\begin{aligned}\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{W}\|_F^2 &= \frac{1}{r^2}\sum_{i=1}^{k} \mathbf{c}_i \cdot \mathbf{d}_i \\ &\leq \frac{1}{r^2}O(1)k\log(1/\delta)p \\ &= \frac{1}{r^2}O(1)k\log(1/\delta)\|\boldsymbol{W}\|_F^2.\end{aligned}$$

Since $k < \log(1/\delta)$, $r = O(k + \log(1/\delta))$ in this case gives the following with probability $1 - \delta$:

$$\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{W}\|_F \leq O(1)\|\boldsymbol{W}\|_F.$$

Combining **Case I** and **Case II** allows us to conclude that for $r = O(k+\log(1/\delta))$, $\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{W}\|_F \leq O(1)\|\boldsymbol{W}\|_F$ with probability $1 - \delta$.

□

Next, the following **Lemma** 3.2.8 is needed to construct an $O(1)$-approximate rank-k approximation $\widetilde{\boldsymbol{A}}$ with $\boldsymbol{S}, \boldsymbol{R}$ having a size of only $O(k + \log(1/\delta))$ as used in **Theorem** 3.2.9.

Lemma 3.2.8 (Upper Bound on Regression Error)**.** *Given $\delta \in (0, \frac{1}{2})$, let $\boldsymbol{A}, \boldsymbol{B}$ be matrices that both have n rows and rank$(\boldsymbol{A}) = k$. Let $\boldsymbol{S} \in \mathbb{R}^{n\times r}$ be a random matrix with i.i.d. $\mathcal{N}(0, \frac{1}{r})$ Gaussian random variables. Let $\widetilde{\boldsymbol{X}} = \arg\min_{\boldsymbol{X}}\|\boldsymbol{S}^T(\boldsymbol{A}\boldsymbol{X} - \boldsymbol{B})\|_F$ and $\boldsymbol{X}^* = \arg\min_{\boldsymbol{X}} \|\boldsymbol{A}\boldsymbol{X} - \boldsymbol{B}\|_F$. For $r = O(k + \log(1/\delta))$, the following holds with probability $1 - \delta$,*

$$\|\boldsymbol{A}\widetilde{\boldsymbol{X}} - \boldsymbol{B}\|_F \leq O(1)\|\boldsymbol{A}\boldsymbol{X}^* - \boldsymbol{B}\|_F.$$

Proof. Consider an orthonormal basis $\boldsymbol{U}$ for the column span of $\boldsymbol{A}$. Let $\widetilde{\boldsymbol{Y}} = \arg\min_{\boldsymbol{Y}}\|\boldsymbol{S}\boldsymbol{U}\boldsymbol{Y} - \boldsymbol{S}\boldsymbol{B}\|_2$ and $\boldsymbol{Y}^* = \arg\min_{\boldsymbol{Y}}\|\boldsymbol{U}\boldsymbol{Y} - \boldsymbol{B}\|_2$. By the normal equations, the solutions to the two least squares problems are $\widetilde{\boldsymbol{Y}} = (\boldsymbol{S}\boldsymbol{U})^\dagger\boldsymbol{S}\boldsymbol{B}$[9] and $\boldsymbol{Y}^* = \boldsymbol{U}^T\boldsymbol{B}$.
We first show that $\|\boldsymbol{U}\widetilde{\boldsymbol{Y}} - \boldsymbol{B}\|_F \leq O(1)\|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F$.

$$\begin{aligned}
\|\boldsymbol{U}\widetilde{\boldsymbol{Y}} - \boldsymbol{B}\|_F^2 &= \|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2 + \|\boldsymbol{U}\widetilde{\boldsymbol{Y}} - \boldsymbol{U}\boldsymbol{Y}^*\|_F^2 \\
&= \|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2 + \|\widetilde{\boldsymbol{Y}} - \boldsymbol{Y}^*\|_F^2 && (\boldsymbol{U} \text{ has orthonormal columns}) \\
&= \|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2 + \|(\boldsymbol{S}\boldsymbol{U})^\dagger\boldsymbol{S}\boldsymbol{B} - \boldsymbol{U}^T\boldsymbol{B}\|_F^2 \\
&= \|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2 + \|(\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{U})^{-1}\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{B} - \boldsymbol{U}^T\boldsymbol{B}\|_F^2.
\end{aligned}$$

Since $\boldsymbol{S}$ is a matrix with i.i.d. $\mathcal{N}(0, \frac{1}{r})$ Gaussian random variables, by **Fact** 3.2.6, for any vector $v \in \mathbb{R}^n$, with probability $1-\delta$ and for some fixed constant $\epsilon_1 \in (0, 1)$, $\|\boldsymbol{S}\boldsymbol{U}v\|_2 = (1\pm\epsilon_1)\|\boldsymbol{U}v\|_2$. This implies the singular values of $\boldsymbol{S}\boldsymbol{U}$ are in the range $[1 - \epsilon_1, 1 + \epsilon_1]$. Thus,

$$\begin{aligned}
\|\boldsymbol{U}\widetilde{\boldsymbol{Y}} - \boldsymbol{B}\|_F^2 &\leq \|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2 + O(1)\|(\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{U})((\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{U})^{-1}\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{B} - \boldsymbol{U}^T\boldsymbol{B})\|_F^2 \\
&= \|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2 + O(1)\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{B} - \boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{B}\|_F^2 \\
&= \|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2 + O(1)\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}(\boldsymbol{B} - \boldsymbol{U}\boldsymbol{Y}^*)\|_F^2.
\end{aligned}$$

[9] † denotes the Moore-Penrose pseudoinverse

Consider $p = \text{rank}(\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B})$. If $p = O(k)$, then $\text{rank}(\boldsymbol{B}) = O(k)$. For $r = O(k)$, we can use $\boldsymbol{S}$ to reconstruct $\boldsymbol{A}$ and $\boldsymbol{B}$. In this case, $\widetilde{\boldsymbol{X}} = \boldsymbol{X}^*$ and so $\|\boldsymbol{U}\widetilde{\boldsymbol{Y}} - \boldsymbol{B}\|_F \leq O(1)\|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F$. If $p = O(\log(1/\delta))$, then $\text{rank}(\boldsymbol{B}) = O(k + \log(1/\delta))$. For $r = O(k + \log(1/\delta))$, we can again use $\boldsymbol{S}$ to reconstruct $\boldsymbol{A}$ and $\boldsymbol{B}$ and get $\|\boldsymbol{U}\widetilde{\boldsymbol{Y}} - \boldsymbol{B}\|_F \leq O(1)\|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F$.

Now consider $p \geq \max(k, \log(1/\delta))$. First note that $\boldsymbol{B} - \boldsymbol{U}\boldsymbol{Y}^* = \boldsymbol{B} - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{B} = (\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{B})$, where $\boldsymbol{U}$ has orthonormal columns and thus, $\boldsymbol{U}\boldsymbol{U}^T$ is the projection matrix onto the column span $\text{col}(\boldsymbol{U})$ of $\boldsymbol{U}$. We have $(\boldsymbol{B} - \boldsymbol{U}\boldsymbol{Y}^*) \perp \text{col}(\boldsymbol{U})$. Second, we can w.l.o.g. assume that $\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}$ has orthonormal columns; indeed, otherwise let $\boldsymbol{U}'\boldsymbol{R}' = \boldsymbol{B} - \boldsymbol{U}\boldsymbol{Y}^*$ be the QR decomposition where $\boldsymbol{U}'$ is an orthonormal basis for $\text{col}(\boldsymbol{B} - \boldsymbol{U}\boldsymbol{Y}^*)$. Then $\|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}(\boldsymbol{B} - \boldsymbol{U}\boldsymbol{Y}^*)\|_F^2 = \|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{U}'\boldsymbol{R}'\|_F^2 = \|\boldsymbol{U}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{U}'\|_F^2$.

Applying **Lemma** 3.2.7, with probability $1 - O(\delta)$,

$$\begin{aligned}\|\boldsymbol{U}\widetilde{\boldsymbol{Y}} - \boldsymbol{B}\|_F^2 &\leq \|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2 + O(1)\|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2 \\ &= O(1)\|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F^2.\end{aligned}$$

This concludes that $\|\boldsymbol{U}\widetilde{\boldsymbol{Y}} - \boldsymbol{B}\|_F \leq O(1)\|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F$. Finally, consider the QR decomposition of $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{R}$ where $\boldsymbol{U}$ is an orthonormal basis for the column span of $\boldsymbol{A}$ and $\boldsymbol{R}$ is an arbitrary matrix. Let $\widetilde{\boldsymbol{X}} = \arg\min_{\boldsymbol{X}} \|\boldsymbol{S}\boldsymbol{A}\boldsymbol{X} - \boldsymbol{S}\boldsymbol{B}\|_2$ and $\boldsymbol{X}^* = \|\boldsymbol{A}\boldsymbol{X} - \boldsymbol{B}\|_2$. Note that

$$\begin{aligned}\min_{\boldsymbol{X}} \|\boldsymbol{S}\boldsymbol{A}\boldsymbol{X} - \boldsymbol{S}\boldsymbol{B}\|_F = \min_{\boldsymbol{Y}} \|\boldsymbol{S}\boldsymbol{U}\boldsymbol{R}\boldsymbol{Y} - \boldsymbol{S}\boldsymbol{B}\|_F = \min_{\boldsymbol{Y}} \|\boldsymbol{S}\boldsymbol{U}\boldsymbol{Y} - \boldsymbol{S}\boldsymbol{B}\|_F \\ \min_{\boldsymbol{X}} \|\boldsymbol{A}\boldsymbol{X} - \boldsymbol{B}\|_F = \min_{\boldsymbol{Y}} \|\boldsymbol{U}\boldsymbol{R}\boldsymbol{Y} - \boldsymbol{B}\|_F = \min_{\boldsymbol{Y}} \|\boldsymbol{U}\boldsymbol{Y} - \boldsymbol{B}\|_F.\end{aligned}$$

Thus,

$$\|\boldsymbol{A}\widetilde{\boldsymbol{X}} - \boldsymbol{B}\|_F = \|\boldsymbol{U}\widetilde{\boldsymbol{Y}} - \boldsymbol{B}\|_F \leq O(1)\|\boldsymbol{U}\boldsymbol{Y}^* - \boldsymbol{B}\|_F = O(1)\|\boldsymbol{A}\boldsymbol{X}^* - \boldsymbol{B}\|_F.$$

□

The following Theorem and its proof follows **Theorem 4.7** of (Clarkson and Woodruff, 2009), except that: 1) to get a rank k approximation to the matrix $\boldsymbol{A}$, the number of columns in the sketching matrices $\boldsymbol{S}$ and $\boldsymbol{R}$ was required to be $m = O(k\log(\frac{1}{\delta}))$ in **Theorem 4.7** of (Clarkson and Woodruff, 2009); 2) $\boldsymbol{S}$ and $\boldsymbol{R}$ in **Theorem 4.7** of (Clarkson and Woodruff, 2009) are random sign matrices. By applying **Lemma** 3.2.8, we show that this number m can be reduced to $O(k + \log(\frac{1}{\delta}))$, and consider a specific application to PSD matrices.

Theorem 3.2.9. *Let $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ be an arbitrary PSD matrix. Let $\boldsymbol{A}_k = \arg\min_{\text{rank-}k\, A_k} \|A - A_k\|_F$ be the optimal rank-k approximation to $\boldsymbol{A}$ in Frobenius norm. If $\boldsymbol{S} \in \mathbb{R}^{n\times m}$ and $\boldsymbol{R} \in \mathbb{R}^{n\times cm}$ are random matrices with i.i.d. $\mathcal{N}(0,1)$ entries for some fixed constant $c > 0$ with $m = O(k+\log(1/\delta))$, then with probability $1 - \delta$, the matrix $\widetilde{\boldsymbol{A}} = (\boldsymbol{A}\boldsymbol{R})(\boldsymbol{S}^T\boldsymbol{A}\boldsymbol{R})^\dagger(\boldsymbol{A}\boldsymbol{S})^T$ satisfies*

$$\|\boldsymbol{A} - \widetilde{\boldsymbol{A}}\|_F \leq O(1)\|\boldsymbol{A} - \boldsymbol{A}_k\|_F.$$

Proof. First, we consider $\boldsymbol{S}$ to be a random matrix with i.i.d. $\mathcal{N}(0, \frac{1}{m})$ entries and $\boldsymbol{R}$ to be a random matrix with i.i.d. $\mathcal{N}(0, \frac{1}{cm})$ entries.

Consider $\widetilde{\boldsymbol{X}} = \arg\min_{\boldsymbol{X}} \|\boldsymbol{S}^T\boldsymbol{A}\boldsymbol{R}\boldsymbol{X} - \boldsymbol{S}^T\boldsymbol{A}\|_F = (\boldsymbol{S}^T\boldsymbol{A}\boldsymbol{R})^\dagger\boldsymbol{S}^T\boldsymbol{A}$ and $\boldsymbol{X}^* = \arg\min_{\boldsymbol{X}} \|\boldsymbol{A}\boldsymbol{R}\boldsymbol{X} - \boldsymbol{A}\|_F$. By **Lemma** 3.2.8, with probability $1-\delta$,

$$\|\boldsymbol{A}\boldsymbol{R}\widetilde{\boldsymbol{X}} - \boldsymbol{A}\|_F \leq O(1)\|\boldsymbol{A}\boldsymbol{R}\boldsymbol{X}^* - \boldsymbol{A}\|_F.$$

Now let $\boldsymbol{A}_k = \arg\min_{\text{rank k } A_k} \|\boldsymbol{A} - \boldsymbol{A}_k\|_F$ be the optimal rank-k approximation to $\boldsymbol{A}$.

Consider $\boldsymbol{X}_{opt} = \arg\min_{\boldsymbol{X}} \|\boldsymbol{X}\boldsymbol{A}_k - \boldsymbol{A}\|_F$ and $\boldsymbol{X}' = \arg\min_{\boldsymbol{X}} \|\boldsymbol{X}\boldsymbol{A}_k\boldsymbol{R} - \boldsymbol{A}\boldsymbol{R}\|_F = (\boldsymbol{A}\boldsymbol{R})(\boldsymbol{A}_k\boldsymbol{R})^\dagger$. By **Lemma** 3.2.8 again, with probability $1-\delta$,

$$\begin{aligned}\|\boldsymbol{X}'\boldsymbol{A}_k - \boldsymbol{A}\|_F &= \|(\boldsymbol{A}\boldsymbol{R})(\boldsymbol{A}_k\boldsymbol{R})^\dagger\boldsymbol{A}_k - \boldsymbol{A}\|_F \\ &\leq O(1)\|\boldsymbol{X}_{opt}\boldsymbol{A}_k - \boldsymbol{A}\|_F = O(1)\|\boldsymbol{A} - \boldsymbol{A}_k\|_F.\end{aligned}$$

This implies a good rank-k approximation exists in the column span of $\boldsymbol{A}\boldsymbol{R}$. We now have with probability $1-\delta$,

$$\|\boldsymbol{A}\boldsymbol{R}\boldsymbol{X}^* - \boldsymbol{A}\|_F \leq \|(\boldsymbol{A}\boldsymbol{R})(\boldsymbol{A}_k\boldsymbol{R})^\dagger\boldsymbol{A}_k - \boldsymbol{A}\|_F \leq O(1)\|\boldsymbol{A} - \boldsymbol{A}_k\|_F.$$

Thus by a union bound, with probability $1-2\delta$,

$$\begin{aligned}\|\boldsymbol{A}\boldsymbol{R}(\boldsymbol{S}^T\boldsymbol{A}\boldsymbol{R})^\dagger\boldsymbol{S}^T\boldsymbol{A} - \boldsymbol{A}\|_F &= \|\boldsymbol{A}\boldsymbol{R}\widetilde{\boldsymbol{X}} - \boldsymbol{A}\|_F \\ &\leq O(1)\|\boldsymbol{A}\boldsymbol{R}\boldsymbol{X}^* - \boldsymbol{A}\|_F \\ &\leq O(1)\|\boldsymbol{A} - \boldsymbol{A}_k\|_F.\end{aligned}$$

Since we consider PSD $\boldsymbol{A}$, $\boldsymbol{S}^T\boldsymbol{A} = (\boldsymbol{A}\boldsymbol{S})^T$. Let $\widetilde{\boldsymbol{A}} = (\boldsymbol{A}\boldsymbol{R})(\boldsymbol{S}^T\boldsymbol{A}\boldsymbol{R})^\dagger(\boldsymbol{A}\boldsymbol{S})^T$, it follows that with probability $1-2\delta$,

$$\|\boldsymbol{A} - \widetilde{\boldsymbol{A}}\|_F \leq O(1)\|\boldsymbol{A} - \boldsymbol{A}_k\|_F.$$

Let $\boldsymbol{S}' = \sqrt{m}\boldsymbol{S}$ and $\boldsymbol{R}' = \sqrt{cm}\boldsymbol{R}$ so that both $\boldsymbol{S}'$ and $\boldsymbol{R}'$ have i.i.d. $\mathcal{N}(0,1)$ entries. Notice that $(\boldsymbol{A}\boldsymbol{R}')(\boldsymbol{S}'^T\boldsymbol{A}\boldsymbol{R}')^\dagger(\boldsymbol{A}\boldsymbol{S}')^T = (\boldsymbol{A}\boldsymbol{R})(\boldsymbol{S}^T\boldsymbol{A}\boldsymbol{R})^\dagger(\boldsymbol{A}\boldsymbol{S})^T$. Thus $\boldsymbol{S}$, $\boldsymbol{R}$ can be chosen to both be random matrices with i.i.d. $\mathcal{N}(0,1)$ entries. The theorem follows after adjusting δ by a constant factor. □

Theorem 3.2.10 (Theorem 4 of (Meyer et al., 2020)). *Let $\boldsymbol{A} \in \mathbb{R}^{d\times d}$ be PSD, $\delta \in (0, \frac{1}{2})$, $l \in \mathbb{N}^+, k \in \mathbb{N}^+$. Let $\widetilde{\boldsymbol{A}}$ and $\boldsymbol{\Delta}$ be any matrices with $tr(\boldsymbol{A}) = tr(\widetilde{\boldsymbol{A}}) + tr(\boldsymbol{\Delta})$ and $\|\boldsymbol{\Delta}\|_F \leq O(1)\|\boldsymbol{A} - \boldsymbol{A}_k\|_F$ where $\boldsymbol{A}_k = \arg\min_{rank\ k\ \boldsymbol{A}_k}\|\boldsymbol{A} - \boldsymbol{A}_k\|_F$. Let $H_l(\boldsymbol{M})$ denote Hutchinson's trace estimator with l queries on matrix $\boldsymbol{M}$. For fixed constants c, C, if $l \geq c\log(\frac{1}{\delta})$, then with probability $1-\delta$, $Z = tr(\widetilde{\boldsymbol{A}}) + H_l(\boldsymbol{\Delta})$,*

$$\|Z - tr(\boldsymbol{A})\| \leq C\sqrt{\frac{\log(1/\delta)}{kl}} \cdot tr(\boldsymbol{A}).$$

Proof. Set $k = l = O\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon}\right)$.

Consider $\widetilde{\boldsymbol{A}} = (\boldsymbol{AR})(\boldsymbol{S}^T\boldsymbol{AR})^\dagger(\boldsymbol{AS})^T$, where $\boldsymbol{S} \in \mathbb{R}^{n\times s}, \boldsymbol{R} \in \mathbb{R}^{n\times r}$ are both random matrices with i.i.d. $\mathcal{N}(0,1)$ entries, and $\boldsymbol{\Delta} = \boldsymbol{A} - \widetilde{\boldsymbol{A}}$. By **Theorem** 3.2.9, for $s = r = O\left(k + \log(1/\delta)\right) = O\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon} + \log(1/\delta)\right)$, with probability $1-\delta$,

$$\|\boldsymbol{\Delta}\|_F \leq O(1) \cdot \|\boldsymbol{A} - \boldsymbol{A}_k\|_F.$$

Thus for the output of NA-Hutch++, t, by **Theorem** 3.2.10 and a union bound, with probability $1-2\delta$,

$$|t - \mathrm{tr}(\boldsymbol{A})| \leq \epsilon \cdot \mathrm{tr}(\boldsymbol{A}).$$

The total number of non-adaptive queries NA-Hutch++ needs is

$$m = s + r + l = O\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon} + \log(1/\delta)\right).$$

□

3.2.6 Lower Bounds

In this section, we show that a query complexity of $O\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon} + \log(1/\delta)\right)$ is tight for any non-adaptive trace estimation algorithm, up to a $O(\log\log(1/\delta))$ factor, stated in **Theorem** 3.2.11. The analysis considers two separate cases: for small ϵ, we show the term $O\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon}\right)$ is tight in Section 3.2.6.1, and for any ϵ, we show the term $O(\log(1/\delta))$ is tight up to a $O(\log\log(1/\delta))$ factor in Section 3.2.6.2. When combined, these two lower bounds handle arbitrary ϵ, since the latter lower bound dominates precisely when the former lower bound does not apply.

Our hard distribution consists of shifted Wigner matrices and exploits the symmetry and concentration properties of the Gaussian ensemble.

Theorem 3.2.11 (Lower Bound for Non-Adaptive Queries)**.** *Let $\epsilon \in (0,1)$. Any algorithm that accesses a real PSD matrix $\boldsymbol{A}$ through matrix-vector multiplication queries $\boldsymbol{Aq}_1, \boldsymbol{Aq}_2, \ldots, \boldsymbol{Aq}_m$, where $\mathbf{q}_1, \ldots, \mathbf{q}_m$ are real-valued, non-adaptively chosen vectors, requires*

$$m = \Omega\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon} + \frac{\log(1/\delta)}{\log\log(1/\delta)}\right)$$

queries to output an estimate t such that with probability at least $1-\delta$, $(1-\epsilon)tr(\boldsymbol{A}) \leq t \leq (1+\epsilon)tr(\boldsymbol{A})$.

Proof of Theorem 3.2.11. For small $\epsilon = O(1/\sqrt{\log(1/\delta)})$, note that the first term $\frac{\sqrt{\log(1/\delta)}}{\epsilon}$ dominates. **Theorem** 3.2.13 (see Section 3.2.6.1) shows any algorithm needs $\Omega\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon}\right)$ non-adaptive queries in this case.

For $\epsilon > 1/\sqrt{\log(1/\delta)}$, note that the second term $\log(1/\delta)$ dominates. **Theorem** 3.2.14 (see Section 3.2.6.2) shows any algorithm needs $\Omega(\frac{\log(1/\delta)}{\log\log(1/\delta)})$ non-adaptive queries for any $\epsilon \in (0, 1)$.

The two cases combined imply an $\Omega\left(\frac{\sqrt{\log(1/\delta)}}{\epsilon} + \frac{\log(1/\delta)}{\log\log(1/\delta)}\right)$ lower bound. □

Our lower bounds crucially make use of rotational invariance of the Gaussian distribution (see Fact 3.2.2) to argue that the first q queries are, w.l.o.g., the standard basis vectors $e_1, ..., e_q$. Note that our queries can be assumed to be orthonormal. Both lower bounds use the family of $n \times n$ Wigner matrices (see Definition 3.2.3) with shifted mean, i.e., $\boldsymbol{W} + C \cdot \boldsymbol{I}$ for some $C > 0$ depending on $\|\boldsymbol{W}\|_{op}$, as part of the hard input distribution. The mean shift ensures that our ultimate instance is PSD with high probability.

3.2.6.1 Case 1: Lower Bound for Small ϵ

Suppose that we draw a matrix $\boldsymbol{G} \in \mathbb{R}^{n\times n}$ from the Gaussian distribution and try to learn the entries of the matrix via matrix-vector queries. After a few queries, it turns out that the conditional distribution of the remaining matrix is also Gaussian-distributed, no matter how the queries are chosen. This nice property allows concise reasoning for lower bounding the remaining uncertainty of the matrix, even after seeing a few query results.

Lemma 3.2.12. *(Conditional Distribution [Lemma 3.4 of (Simchowitz et al., 2018)]) Let $\boldsymbol{G} \sim \mathcal{N}(n)$ be as in Definition 3.2.3 and suppose our matrix is $\boldsymbol{W} = (\boldsymbol{G}+\boldsymbol{G}^\top)/2$. Suppose we have any sequence of vector queries, $\boldsymbol{v}_1, ..., \boldsymbol{v}_T$, along with responses $\boldsymbol{w}_i = \boldsymbol{W}\boldsymbol{v}_i$. Then, conditioned on our observations, there exists a rotation matrix $\boldsymbol{V}$, independent of $\boldsymbol{w}_i$, such that*

$$\boldsymbol{V}\boldsymbol{W}\boldsymbol{V}^\top = \begin{bmatrix} Y_1 & Y_2^\top \\ Y_2 & \widetilde{\boldsymbol{W}} \end{bmatrix},$$

where Y_1, Y_2 are deterministic and $\widetilde{\boldsymbol{W}} = (\widetilde{\boldsymbol{G}} + \widetilde{\boldsymbol{G}}^\top)/2$, where $\widetilde{\boldsymbol{G}} \sim \mathcal{N}(n - T)$.

Theorem 3.2.13 (Lower Bound for Small ϵ)**.** *For any PSD matrix $\boldsymbol{A}$ and all $\epsilon = O\left(1/\sqrt{\log(1/\delta)}\right)$, any algorithm that succeeds with probability at least $1 - \delta$ in outputting an estimate t such that $(1 - \epsilon)tr(\boldsymbol{A}) \le t \le (1 + \epsilon)tr(\boldsymbol{A})$, requires*

$$m = \Omega\left(\sqrt{\log(1/\delta)}/\epsilon\right)$$

matrix-vector queries.

Proof. By standard minimax arguments, it suffices to construct a hard distribution for any deterministic algorithm.

Consider $\boldsymbol{G} \sim \mathcal{N}(n)$ for $n = \Omega(\log(1/\delta))$. From concentration of the singular values of large Gaussian matrices (Lemma 3.2.4), with probability at least $1 - \delta/10$ we have $\|\boldsymbol{G}\|_{op} \leq C\sqrt{n}$ for some absolute constant C.

Therefore, consider the family of matrices $\boldsymbol{W} = \boldsymbol{I} + \frac{1}{2C\sqrt{n}}(\boldsymbol{G} + \boldsymbol{G}^\top)$. From our bound on $\|\boldsymbol{G}\|_{op}$, with probability at least $1 - \delta/10$, $\boldsymbol{W}$ is positive semi-definite and symmetric. Furthermore, since $\text{tr}(\boldsymbol{G}) \sim N(0, n)$, we see that $\text{tr}(\boldsymbol{W}) \leq 2n$ with probability at least $1 - \delta/10$.

We set the multiplicative error to $\epsilon = \frac{\sqrt{\log(1/\delta)}}{n}$ and it suffices to show that if we see only $n/2$ queries, we can compute $\text{tr}(\boldsymbol{W})$ up to additive error at best $c\sqrt{\log(1/\delta)}$ with probability at least $1 - \delta$, for some $c = \Omega(1)$. By **Lemma** 3.2.12, we see that conditioned on the queries, our matrix $\boldsymbol{W}$ can be decomposed into a determined part and a Gaussian submatrix $\widetilde{\boldsymbol{W}} = \frac{1}{2C\sqrt{n}}(\widetilde{\boldsymbol{G}} + \widetilde{\boldsymbol{G}}^\top)$, where $\widetilde{\boldsymbol{G}} \sim \mathcal{N}(n/2)$.

Therefore, our conditional distribution of the trace of $\boldsymbol{W}$ is, up to a deterministic shift, the same as the distribution of $\widetilde{\boldsymbol{W}}$, which is simply a Gaussian with variance $1/C^2$. Since we must determine a Gaussian of constant variance up to an additive error of $c\sqrt{\log(1/\delta)}$ with probability at least $1 - \delta$, we conclude that $c = \Omega(1)$. □

3.2.6.2 Case 2: Lower Bound for Every ϵ

We give a general $\Omega(\frac{\log(1/\delta)}{\log\log(1/\delta)})$ lower bound, that holds for every $\epsilon \in (0, 1)$, on the query complexity for non-adaptive trace estimation algorithms stated in **Theorem** 3.2.14. The proof of **Theorem** 3.2.14 is via a reduction to a distribution testing problem in **Problem** 3.2.15, whose hardness (in terms of query complexity) is shown in **Lemma** 3.2.15.

Theorem 3.2.14 (Lower Bound on Non-adaptive Queries for PSD Matrices)**.** *Let $\epsilon \in (0, 1)$. Any algorithm that accesses a real, PSD matrix $\boldsymbol{A}$ through matrix-vector queries $\boldsymbol{A}\mathbf{q}_1, \boldsymbol{A}\mathbf{q}_2, \ldots, \boldsymbol{A}\mathbf{q}_m$, where $\mathbf{q}_1, \ldots, \mathbf{q}_m$ are real-valued non-adaptively chosen vectors, requires*

$$m = \Omega\left(\frac{\log(1/\delta)}{\log\log(1/\delta)}\right)$$

to output an estimate t such that with probability at least $1 - \delta$, $(1 - \epsilon)tr(\boldsymbol{A}) \leq t \leq (1 + \epsilon)tr(\boldsymbol{A})$.

Proof. The proof is via reduction to a distribution testing problem stated in **Problem** 3.2.15. Given a real, PSD input matrix $\boldsymbol{A}$, let $\mathcal{A}$ be an algorithm that uses m non-adaptive matrix-vector queries and outputs a trace estimation t of $\boldsymbol{A}$ such that for some $\epsilon \in (0, 1)$, with probability at least $1 - \delta$, $(1 - \epsilon)\text{tr}(\boldsymbol{A}) \leq t \leq (1 + \epsilon)\text{tr}(\boldsymbol{A})$.

Consider $n = \log(1/\delta)$. Let $Z_i, \forall i \in [n]$ be the i-th diagonal entry of $\boldsymbol{W} \sim \mathcal{W}(n) = \boldsymbol{G} + \boldsymbol{G}^T$ as in Definition 3.2.3. Note that $\boldsymbol{G}$ has i.i.d. $\mathcal{N}(0, 1)$ entries, and that the diagonal of $\boldsymbol{G}$ and $\boldsymbol{G}^T$ are the same. This implies $Z_i \sim \mathcal{N}(0, 4)$.

Since the Z_i are i.i.d.,

$$\mathrm{tr}(\boldsymbol{W}) = \sum_{i=1}^{n} Z_i \sim \mathcal{N}(0, 4n) = \mathcal{N}\left(0, 4\log(1/\delta)\right).$$

By **Fact** 3.2.3,

$$\Pr[\mathrm{tr}(\boldsymbol{W}) \geq 2\sqrt{2}\log(1/\delta)] \leq \delta$$
$$\Pr[\mathrm{tr}(\boldsymbol{W}) \leq -2\sqrt{2}\log(1/\delta)] \leq \delta.$$

For a unit vector $\frac{\mathbf{g}}{\|\mathbf{g}\|_2} \in \mathbb{R}^n$,

$$\mathrm{tr}\left(\frac{\mathbf{g}}{\|\mathbf{g}\|_2}\frac{\mathbf{g}^T}{\|\mathbf{g}\|_2}\right) = \|\frac{\mathbf{g}}{\|\mathbf{g}\|_2}\|_2^2 = 1.$$

Let $\boldsymbol{B}$ be the random matrix generated from distribution $\mathcal{P}$ or $\mathcal{Q}$ in **Problem** 3.2.15. First, we claim that with probability at least $1 - 4\delta$, $\boldsymbol{B}$ is a PSD matrix. Note that $C\log^{3/2}(\frac{1}{\delta}) \cdot \frac{1}{\|\mathbf{g}\|_2^2}\mathbf{g}\mathbf{g}^T$ is PSD. Thus it suffices to show $\boldsymbol{W} + 6\sqrt{\log(\frac{1}{\delta})}\boldsymbol{I}$ is PSD with high probability.

By **Lemma** 3.2.4, with probability $1 - 2\delta$,

$$\|\boldsymbol{G}\|_{op} \leq 3\sqrt{\log(1/\delta)}.$$

By the triangle inequality and a union bound, with probability $1 - 4\delta$,

$$\|\boldsymbol{W}\|_{op} = \|\boldsymbol{G} + \boldsymbol{G}^T\|_{op} \leq 6\sqrt{\log(1/\delta)}.$$

This implies $\boldsymbol{W} + 6\sqrt{\log(\frac{1}{\delta})}\boldsymbol{I}$ is PSD with probability $1 - 4\delta$.

If $\boldsymbol{B} \sim \mathcal{P}$, with probability at least $1 - \delta$,

$$\begin{aligned}\mathrm{tr}(\boldsymbol{B}) &= C\log^{3/2}(1/\delta) + \mathrm{tr}(\boldsymbol{W}) + 6\log^{3/2}(1/\delta) \\ &\geq (C+6)\log^{3/2}(1/\delta) - 2\sqrt{2}\log(1/\delta).\end{aligned}$$

If $\boldsymbol{B} \sim \mathcal{Q}$, with probability at least $1 - \delta$,

$$\mathrm{tr}(\boldsymbol{B}) = \mathrm{tr}(\boldsymbol{W}) + 6\log^{3/2}(\log(1/\delta)) \leq 2\sqrt{2}\log(1/\delta) + 6\log^{3/2}(1/\delta).$$

Consider the trace estimation algorithm $\mathcal{A}$ and let the output $t = \mathcal{A}(\boldsymbol{B})$. Consider the constant $C > \frac{10(1+\epsilon)}{1-\epsilon} - 6$. If $\boldsymbol{B} \sim \mathcal{P}$, with probability at least $1 - 2\delta$,

$$\begin{aligned}t &\geq (1-\epsilon)\mathrm{tr}(\boldsymbol{B}) \\ &\geq (1-\epsilon)\left((C+6)\log^{3/2}(1/\delta) - 2\sqrt{2}\log(1/\delta)\right) \\ &> 6(1+\epsilon)\log^{3/2}(1/\delta).\end{aligned}$$

If $\boldsymbol{B} \sim \mathcal{Q}$, with probability at least $1 - 2\delta$,

$$\begin{aligned} t &\leq (1+\epsilon)\text{tr}(\boldsymbol{B}) \\ &\leq (1+\epsilon)\left(6\log^{3/2}(1/\delta) + 2\sqrt{2}\log(1/\delta)\right) \\ &< 6(1+\epsilon)\log^{3/2}(1/\delta). \end{aligned}$$

In the worst case, if any of the instances generated from $\mathcal{P}$ or $\mathcal{Q}$ is non-PSD, our algorithm $\mathcal{A}$ fails. Thus $\mathcal{A}$ determines which distribution $\boldsymbol{B}$ comes from with probability at least $1 - 6\delta$. By **Lemma** 3.2.15, this requires the number of matrix-vector queries $\mathcal{A}$ uses to be $m = \Omega(\frac{\log(1/\delta)}{\log\log(1/\delta)})$. □

Problem 3.2.15 (Hard PSD Matrix Distribution Test)**.** *Given $\delta \in (0, \frac{1}{2})$, set $n = \log(1/\delta)$. Choose $\mathbf{g} \in \mathbb{R}^n$ to be an independent random vector with i.i.d. $\mathcal{N}(0,1)$ entries. Consider two distributions:*

- *Distribution $\mathcal{P}$ on matrices $\left\{C\log^{3/2}(\frac{1}{\delta}) \cdot \frac{1}{\|\mathbf{g}\|_2^2}\mathbf{g}\mathbf{g}^T + \boldsymbol{W} + 6\sqrt{\log(\frac{1}{\delta})}\boldsymbol{I}\right\}$, for some fixed constant $C > 1$.*
- *Distribution $\mathcal{Q}$ on matrices $\left\{\boldsymbol{W} + 6\sqrt{\log(\frac{1}{\delta})}\boldsymbol{I}\right\}$.*

where $\boldsymbol{W} \sim \mathcal{W}(n) = \boldsymbol{G} + \boldsymbol{G}^T$ as in Definition 3.2.3. Let $\boldsymbol{A}$ be a random matrix drawn from either $\mathcal{P}$ or $\mathcal{Q}$ with equal probability. Consider any algorithm which, for a fixed query matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times q}$, observes $\boldsymbol{AQ}$, and guesses if $\boldsymbol{A} \sim \mathcal{P}$ or $\boldsymbol{A} \sim \mathcal{Q}$ with success probability at least $1 - \delta$.

Lemma 3.2.15 (Hardness of Problem 3.2.15)**.** *Given $\delta \in (0, \frac{1}{2})$. Consider a non-adaptively chosen query matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times q}$ on input $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, as in **Problem** 3.2.15, where $n = \log(1/\delta)$. If $q = o(\frac{\log(1/\delta)}{\log\log(1/\delta)})$, no algorithm can solve **Problem** 3.2.15 with success probability $1 - \delta$.*

Proof. We claim that without loss of generality, we only need to consider $\boldsymbol{Q}$ to be the first q standard basis vectors, i.e., $\boldsymbol{Q} = \boldsymbol{E}_q = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_q]$. First note that we only need to consider query matrix $\boldsymbol{Q}$ with orthonormal columns, since for general $\boldsymbol{Q}$, letting $\boldsymbol{Q} = \boldsymbol{UR}$ be the QR decomposition of $\boldsymbol{Q}$, we can reconstruct $\boldsymbol{AQ}$ from $(\boldsymbol{AU})\boldsymbol{R}$. Next, let $\bar{\boldsymbol{Q}} \in \mathbf{R}^{n\times(n-q)}$ be the orthonormal basis for null$(\boldsymbol{Q})$. Define an orthornomal matrix $\boldsymbol{R} = [\boldsymbol{Q}, \bar{\boldsymbol{Q}}] \in \mathbf{R}^{n\times n}$. By **Fact** 3.2.2, $\boldsymbol{WE}_q$ has the same distribution as $\boldsymbol{WRE}_q = \boldsymbol{WQ}$. Similarly, $\left(C\log(\frac{1}{\delta}) \cdot \frac{1}{\|\mathbf{g}\|_2^2}\mathbf{g}\mathbf{g}^T + \boldsymbol{W}\right)\boldsymbol{E}_q$ has the same distribution as

$$\left(C\log(\frac{1}{\delta}) \cdot \frac{1}{\|\mathbf{g}\|_2^2}\mathbf{g}\mathbf{g}^T + \boldsymbol{W}\right)\boldsymbol{Q}.$$

Therefore, we only need to consider the case when the queries are the first q standard basis vectors.

Consider the two possible observed distributions from **Problem** 3.2.15: 1) distribution $\mathcal{P}'$, which has

$$\left(C\log(\frac{1}{\delta}) \cdot \frac{1}{\|\mathbf{g}\|_2^2}\mathbf{g}\mathbf{g}^T + \boldsymbol{W} + 2\sqrt{\log(1/\delta)}\boldsymbol{I}\right)\boldsymbol{Q}$$

for fixed constant $C > 1$, and 2) distribution $\mathcal{Q}'$ which has

$$\left(\boldsymbol{W} + 2\sqrt{\log(1/\delta)}\boldsymbol{I}\right)\boldsymbol{Q}.$$

We argue that if the number q of queries is too small, then the total variation distance between $\mathcal{P}'$ and $\mathcal{Q}'$, conditioned on an event $\mathcal{E}$ with probability at least δ, is upper bounded by a small constant. This will imply that no algorithm can succeed with probability at least $1-\delta$. We upper bound the total variation distance between $\mathcal{P}'$ and $\mathcal{Q}'$ via the Kullback–Leibler (KL) divergence between $\mathcal{P}'$ and $\mathcal{Q}'$ and then apply Pinsker's inequality.

Consider the following event on over the randomness of $\mathbf{g}$:

$$\mathcal{E} = \left\{\mathbf{g} : \frac{1}{\|\mathbf{g}\|^2}\|\mathbf{g}^T\boldsymbol{Q}\|^2 \leq \frac{1}{50C^2n^3}\right\}.$$

Note that $\mathbf{g}^T\boldsymbol{Q} = [\langle \mathbf{g}, \mathbf{e}_1\rangle, \langle \mathbf{g}, \mathbf{e}_2\rangle, \ldots, \langle \mathbf{g}, \mathbf{e}_q\rangle] = [\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_q]$, i.e., the first q coordinates of $\mathbf{g}$. First, we show that $\Pr[\mathcal{E}] = \Omega(\delta)$.

Since $\mathbf{g}_i \sim \mathcal{N}(0,1)$, by **Fact** 3.2.4, for the i-th entry of $\mathbf{g}^T\boldsymbol{Q}$, $\forall i \in [q]$,

$$\Pr\left[|\mathbf{g}_i| \leq \frac{1}{10C \cdot n\sqrt{q}}\right] = \Omega\left(\frac{1}{n\sqrt{q}}\right),$$

which implies for a single entry,

$$\Pr\left[\mathbf{g}_i^2 \leq \frac{1}{100C^2 \cdot n^2 q}\right] = \Omega\left(\frac{1}{n\sqrt{q}}\right).$$

Since all q queries are independent, for all entries $i \in [q]$,

$$\Pr\left[\|\mathbf{g}^T\boldsymbol{Q}\|_2^2 \leq \frac{1}{100C^2 \cdot n^2}\right] = \Omega\left(\left(\frac{1}{n\sqrt{q}}\right)^q\right) = \Omega\left(\exp\left(-\frac{q}{2}\ln(n^2 q)\right)\right).$$

Consider the following conditional probability,

$$\begin{aligned}
&\Pr\left[\|\mathbf{g}^T\boldsymbol{Q}\|_2^2 \leq \frac{1}{100C^2 \cdot n^2} \wedge \|\mathbf{g}\|_2^2 \geq \frac{n}{2}\right] \\
=&\Pr\left[\|\mathbf{g}\|_2^2 \geq \frac{n}{2} \;\middle|\; \|\mathbf{g}^T\boldsymbol{Q}\|_2^2 \leq \frac{1}{100C^2 \cdot n^2}\right] \cdot \Pr\left[\|\mathbf{g}^T\boldsymbol{Q}\|_2^2 \leq \frac{1}{100C^2 \cdot n^2}\right].
\end{aligned}$$

Assume $q < \frac{n}{2}$ and let $\mathbf{g}_{(q+1):n}$ denote the $q+1$-th to the n-th entry of $\mathbf{g}$. Note that all entries of $\mathbf{g}$ are independent and $\|\mathbf{g}_{(q+1):n}\|_2^2 \sim \chi^2(d)$ with degree $d > \frac{n}{2}$. By **Fact** 3.2.1, since $\|\mathbf{g}\|_2^2 \geq \|\mathbf{g}_{(q+1):n}\|_2^2$,

$$\Pr\left[\|\mathbf{g}\|_2^2 \geq \frac{n}{2} \;\middle|\; \|\mathbf{g}^T\boldsymbol{Q}\|_2^2 \leq \frac{1}{100C^2 \cdot n^2}\right] = \Omega(1).$$

Thus,

$$\Pr\left[\frac{1}{\|\mathbf{g}\|_2^2}\|\mathbf{g}^T\boldsymbol{Q}\|_2^2 \le \frac{1}{50C^2n^3}\right] \ge \Pr\left[\|\mathbf{g}^T\boldsymbol{Q}\|_2^2 \le \frac{1}{100C^2\cdot n^2} \wedge \|\mathbf{g}\|_2^2 \ge \frac{n}{2}\right]$$
$$\ge \Omega(1)\cdot\Omega\left(\exp\left(-\frac{q}{2}\ln(n^2q)\right)\right).$$

Assume we only have a small number $q = o\left(\frac{\log(1/\delta)}{\log\log(1/\delta)}\right)$ of queries. Then,

$$\Pr[\mathcal{E}] = \Pr\left[\frac{1}{\|\mathbf{g}\|_2^2}\|\mathbf{g}^T\boldsymbol{Q}\|_2^2 \le \frac{1}{50C^2\cdot n^3}\right] \ge 10\delta. \tag{3.51}$$

Note that $n = \log(1/\delta)$, and so

$$\Pr[\mathcal{E}] = \Pr\left[C^2\log^3(\frac{1}{\delta})\frac{\|\mathbf{g}^T\boldsymbol{Q}\|_2^2}{\|\mathbf{g}\|_2^2} \le \frac{1}{50}\right] \ge 10\delta.$$

Next, note that it suffices to show that the probability of success conditioned on $\mathcal{E}$ is less than $1/3$. This implies our result since $\mathcal{E}$ occurs with probability at least 10δ, implying that our probability of failure is indeed $\Omega(\delta)$. Therefore, we focus on showing that the probability of success conditioned on $\mathbf{g} \in \mathcal{E}$ is small via standard information theoretic arguments with KL divergence bounds.

Conditioning on event $\mathcal{E}$, we now upper bound the KL divergence between $\mathcal{P}'$ and $\mathcal{Q}'$ conditioned on a fixed $\mathbf{g} \in \mathcal{E}$. Since both distributions come from symmetric matrices, we remove the redundant random variables from observed random matrices from $\mathcal{P}'$, $\mathcal{Q}'$ and consider only the lower triangular portion, so that both have dimensions $l = n + (n-1) + \cdots + (n-(q-1))$. Note that these redundant random variables in the upper triangular portion can be removed without increasing the KL divergence, since they are perfectly correlated with its counterpart variable in the lower triangular region, which we show as follows:

Consider two lists $L_{\mathcal{P}'}$, $L_{\mathcal{Q}'}$ of l random variables, corresponding to a vectorization of the observed lower triangular part of the random matrices from $\mathcal{P}'$ and $\mathcal{Q}'$. Consider also a function f, which duplicates parts of the random variables in $L_{\mathcal{P}'}$ and $L_{\mathcal{Q}'}$, such that $f(L_{\mathcal{P}'})$ and $f(L_{\mathcal{Q}'})$ reconstruct the original observed matrix of size $n \times q$ from $\mathcal{P}'$ and $\mathcal{Q}'$, respectively. Then, by the data processing inequality of KL divergence from **Fact** 3.2.7,

$$\mathcal{D}_{KL}(\mathcal{P}' \parallel \mathcal{Q}') = \mathcal{D}_{KL}(f(L_{\mathcal{P}'}) \parallel f(L_{\mathcal{Q}'})) \le \mathcal{D}_{KL}(L_{\mathcal{P}'} \parallel L_{\mathcal{Q}'}).$$

From now on, we assume that $\mathcal{P}'$, $\mathcal{Q}'$ are lower triangular. The KL divergence between $\mathcal{P}'|\mathbf{g}$ and $\mathcal{Q}'|\mathbf{g}$ considering the lower triangular part can be calculated since they are both multivariate Gaussians with the same covariance matrix (of rank l). The KL divergence thus only depends on the difference between the mean $\Delta\mu$ of the two multivariate Gaussians (see Fact 3.2.5), which

is the lower triangular part contained in $C\log^{3/2}(\frac{1}{\delta})\frac{\mathbf{g}\mathbf{g}^T}{\|\mathbf{g}\|_2^2}\boldsymbol{Q}$. Furthermore, since all redundant variables are removed, the distribution on the remaining variables is dimension-independent, with variance 2 from the randomness of $\boldsymbol{W}$.

Let $\widetilde{\boldsymbol{M}} = [\mathbf{m}_1, \ldots, \mathbf{m}_q]$ be the observed lower triangular parts of $\Delta\mu$, where $\mathbf{m}_i \in \mathbb{R}^{n-i+1}, \forall i \in [q]$. Let $\boldsymbol{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_q]$ where $\mathbf{q}_i \in \mathbb{R}^n, \forall i \in [q]$ be the queries. By **Fact** 3.2.5, for any $\mathbf{g} \in \mathcal{E}$ (an event of probability at least 10δ),

$$\begin{aligned}
\mathcal{D}_{KL}(\mathcal{P}'|\mathbf{g} \parallel \mathcal{Q}'|\mathbf{g}) &\leq \mathcal{D}_{KL}(L_{\mathcal{P}'}|\mathbf{g} \parallel L_{\mathcal{Q}'}|\mathbf{g}) && (3.52)\\
&\leq \sum_{i=1}^{q} \left\| C\log^{3/2}\left(\frac{1}{\delta}\right)\mathbf{m}_i \right\|_2^2 && (3.53)\\
&\leq C^2\log^3\left(\frac{1}{\delta}\right)\sum_{i=1}^{q}\left\| \frac{\mathbf{g}\mathbf{g}^T}{\|\mathbf{g}\|_2^2}\mathbf{q}_i \right\|_2^2 && (3.54)\\
&= C^2\log^3\left(\frac{1}{\delta}\right)\sum_{i=1}^{q}\left\langle \frac{\mathbf{g}}{\|\mathbf{g}\|_2}, \mathbf{q}_i \right\rangle^2 && (3.55)\\
&= C^2\log^3\left(\frac{1}{\delta}\right)\frac{\|\mathbf{g}^T\boldsymbol{Q}\|_2^2}{\|\mathbf{g}\|_2^2} && (3.56)\\
&\leq \frac{1}{50}. && (3.57)
\end{aligned}$$

By **Fact** 3.2.6, since conditioning (on $\mathbf{g}$) increases KL divergence between $\mathcal{P}'$ and $\mathcal{Q}'$, let $f(\mathbf{g})$ be the conditional probability density of $\mathbf{g}$ on $\mathcal{E}$. Then,

$$\mathcal{D}_{KL}(\mathcal{P}' \parallel \mathcal{Q}') \leq \int_{\mathbf{g}} \mathcal{D}_{KL}(\mathcal{P}'|\mathbf{g} \parallel \mathcal{Q}'|\mathbf{g}) f(\mathbf{g}) d\mathbf{g} \leq \mathcal{D}_{KL}(\mathcal{P}'|\mathbf{g} \parallel \mathcal{Q}'|\mathbf{g}) = \frac{1}{50}.$$

By Pinsker's inequality, given $\mathcal{E}$ happens,

$$\mathcal{D}_{TV}(\mathcal{P}' \parallel \mathcal{Q}') \leq \sqrt{\frac{1}{2}\mathcal{D}_{KL}(\mathcal{P}' \parallel \mathcal{Q}')} = \sqrt{\frac{1}{100}} < \frac{1}{3}.$$

If the total variation distance between any two distributions $\mathcal{P}'$ and $\mathcal{Q}'$ is at most δ, then any algorithm that distinguishes between $\mathcal{P}'$ and $\mathcal{Q}'$ can succeed with probability at most [10] $\frac{1}{2} + \frac{\delta}{2}$.

Since $\mathcal{D}_{TV}(\mathcal{P}' \parallel \mathcal{Q}') \leq \frac{1}{3}$ in our case, this implies that any algorithm for distinguishing $\mathcal{P}'$ and $\mathcal{Q}'$ can succeed with probability at most $\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{3} = \frac{2}{3}$, and so fails with probability $> \frac{1}{3}$.

[10] For two arbitrary distributions $\mathcal{P}'$ and $\mathcal{Q}'$, let the total variation distance between them be $\mathcal{D}_{TV}(\mathcal{P}' \parallel \mathcal{Q}') = \sup_{\mathcal{E}} |\mathcal{P}'(\mathcal{E}) - \mathcal{Q}'(\mathcal{E})| = \delta$, where $\mathcal{E}$ is an event. Consider an algorithm $\mathcal{A}$ that distinguishes samples from $\mathcal{P}'$ or $\mathcal{Q}'$, and an arbitrary sample $\mathbf{x}$. Let $\mathcal{E} = \Pr[\mathcal{A}(\mathbf{x}) = \mathcal{P}', \mathbf{x} \sim \mathcal{P}']$. If $\mathcal{A}$ succeeds with probability $\geq \frac{1}{2} + \frac{\delta}{2}$, then this implies $\Pr[\mathcal{A}(\mathbf{x}) = \mathcal{P}', \mathbf{x} \sim \mathcal{P}'] \geq \frac{1}{2} + \frac{\delta}{2}$, and $\Pr[\mathcal{A}(\mathbf{x}) = \mathcal{P}', \mathbf{x} \sim \mathcal{Q}'] \geq \frac{1}{2} + \frac{\delta}{2} - \delta = \frac{1}{2} - \frac{\delta}{2}$. This also implies $\Pr[\mathcal{A}(\mathbf{x}) = \mathcal{Q}', \mathbf{x} \sim \mathcal{Q}'] \leq 1 - (\frac{1}{2} - \frac{\delta}{2}) = \frac{1}{2} + \frac{\delta}{2}$, which means the success probability $\mathcal{A}$ is at most $\frac{1}{2} + \frac{\delta}{2}$.

Since $\Pr[\mathcal{E}] \geq 10\delta$, the overall failure probability of an algorithm for distinguishing $\mathcal{P}$ from $\mathcal{Q}$ is thus $10\delta \cdot \frac{1}{3} > \delta$. This implies that to achieve success probability at least $1 - \delta$,

$$q = \Omega(\frac{\log(1/\delta)}{\log\log(1/\delta)}).$$

□

3.2.7 Experiments

[11] We give sequential and parallel implementations of the non-adaptive trace estimation algorithm `NA-Hutch++` (**Algorithm** 2), the adaptive algorithm `Hutch++` (**Algorithm** 1) and `Hutchinson`'s method (Hutchinson, 1989). We specifically explore the benefits of the non-adaptive algorithm in a parallel setting, where all algorithms have parallel access to a matrix-vector oracle. All the code is included in the supplementary material and will be publicly released.

Metrics. We say an estimate failed if on input matrix $\boldsymbol{A}$, the estimate t returned by an algorithm falls into either case: $t < (1-\epsilon)\text{tr}(\boldsymbol{A})$ or $t > (1+\epsilon)\text{tr}(\boldsymbol{A})$. We measure the performance of each algorithm by: 1) the number of failed estimates across 100 random trials, 2) the total wall-clock time to perform 100 trials with sequential execution, and 3) the total wall-clock time to perform 100 trials with parallel execution.

Datasets and Applications. We consider different applications of trace estimation from synthetic to real-world datasets. In many applications, trace estimation is used to estimate not only $\text{tr}(\boldsymbol{A})$, but also $\text{tr}(f(\boldsymbol{A}))$ for some function $f : \mathbb{R} \to \mathbb{R}$. Letting $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ be the eigendecomposition of $\boldsymbol{A}$, we have $f(\boldsymbol{A}) := \boldsymbol{V}f(\boldsymbol{\Sigma})\boldsymbol{V}^\top$, where $f(\boldsymbol{\Sigma})$ denotes applying f to each of the eigenvalues. Due to the expensive computation of eigendecompositions of large matrices, the matrix-vector multiplication $f(\boldsymbol{A})\mathbf{v}$ is often estimated by polynomials implicitly computed via an oracle algorithm for a random vector $\mathbf{v}$. The Lanczos algorithm is a very popular choice due to its superior performance (e.g. (Dong et al., 2017; Ghorbani et al., 2019; Lin et al., 2013)). We compare the performance of our trace estimation algorithms on the following applications and datasets and use the Lanczos algorithm as the matrix-vector oracle on a random vector $\mathbf{v}$ in some particular cases.

- **Fast Decay Spectrum.** We first consider a `synthetic` dataset of size 5000 with a fast decaying spectrum, following (Meyer et al., 2020), which is a diagonal matrix $\boldsymbol{A}$ with i-th diagonal entry $\boldsymbol{A}_{ii} = 1/i^2$. Matrices with fast decaying spectrum will cause high variance in the estimated trace of `Huthinson`, but low variance for `Hutch++` and `NA-Hutch++`. The matrix-vector oracle is simply $\boldsymbol{A}\mathbf{v}$.
- **Graph Estrada Index.** Given a binary adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n\times n}$ of a graph, the Graph Estrada Index is defined as $\text{tr}(\exp(\boldsymbol{A}))$, which measures the strength of connectivity

[11] Our code is available at: https://github.com/11hifish/OptSketchTraceEst

within the graph. Following (Meyer et al., 2020), we use `roget`'s Thesaurus semantic graph[12] with 1022 nodes, which was originally studied in (Estrada and Hatano, 2008), and use the Lanczos algorithm with 40 steps to approximate $\exp(\boldsymbol{A})\mathbf{v}$ as the matrix-vector oracle.

- **Graph Triangle Counting.** Given a binary adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n\times n}$ of a graph, the number of triangles in the graph is $1/6 \cdot \mathrm{tr}(\boldsymbol{A}^3)$. This is an important graph summary with numerous applications in graph-mining and social network analysis (e.g. (Kolountzakis et al., 2010; Pavan et al., 2013)). We use `arxiv_cm`, the Condense Matter collaboration network dataset from arXiv [13]. This is a common benchmark graph with $23,133$ nodes and $173,361$ triangles. The matrix-vector oracle is $\boldsymbol{A}^3\mathbf{v}$. Note that $\boldsymbol{A}^3$, in this case, is not necessarily a PSD matrix.
- **Log-likelihood Estimation for Gaussian Process.** When performing maximum likelihood estimation (MLE) to optimize the hyperparameters of a kernel matrix $\boldsymbol{A}$ for Gaussian Processes, one needs to compute the gradient of the log-determinant of $\boldsymbol{A}$, which involves estimating $\mathrm{tr}(\boldsymbol{A}^{-1})$ (Dong et al., 2017). Following (Dong et al., 2017), we use the `precipitation`[14] dataset, which consists of the measured amount of precipitation during a day collected from 5,500 weather stations in the US in 2010. We sample 1,000 data points and construct a covariance matrix $\boldsymbol{A}$ using the RBF kernel with a length scale 1. We use the Lanczos algorithm with 40 steps as in (Dong et al., 2017) to approximate $\boldsymbol{A}^{-1}\mathbf{v}$ as the matrix-vector oracle.

Implementation. We use random vectors with i.i.d. $\mathcal{N}(0,1)$ entries as the query vectors for all algorithms. NA-Hutch++ requires additional hyperparameters to specify how the queries are split between random matrices $\boldsymbol{S}, \boldsymbol{R}, \boldsymbol{G}$ (see **Algorithm** 2). We set $c_1 = c_3 = \frac{1}{4}$ and $c_2 = \frac{1}{2}$ as Meyer et al. (2020) suggests. For each setting, we conduct 10 random runs and report the mean number of failed estimates across 100 trials and the mean total wall-clock time (in seconds) conducting 100 trials with one standard deviation. For all of our experiments, we fix the error parameter $\epsilon = 0.01$ and measure the performance of each algorithm with $\{10, 30, 50, \ldots, 130, 150\}$ queries on `synthetic`, `roget` and `precipitation`, and with $\{100, 200, \ldots, 700, 800\}$ queries on `arxiv_cm` which has a significantly larger size. The parallel versions are implemented using Python `multiprocessing`[15] package. Due to the large size of `arxiv_cm`, we use `sparse_dot_mkl`[16], a Python wrapper for Intel Math Kernel Library (MKL) which supports fast sparse matrix-vector multiplications, to implement the matrix-vector oracle for this dataset. During the experiments, we launch a pool of 40 worker processes in our parallel execution. All experiments are conducted on machines with 40 CPU cores.

[12] http://vlado.fmf.uni-lj.si/pub/networks/data/
[13] https://snap.stanford.edu/data/ca-CondMat.html
[14] https://catalog.data.gov/dataset/u-s-hourly-precipitation-data
[15] https://docs.python.org/3/library/multiprocessing.html
[16] https://github.com/flatironinstitute/sparse_dot

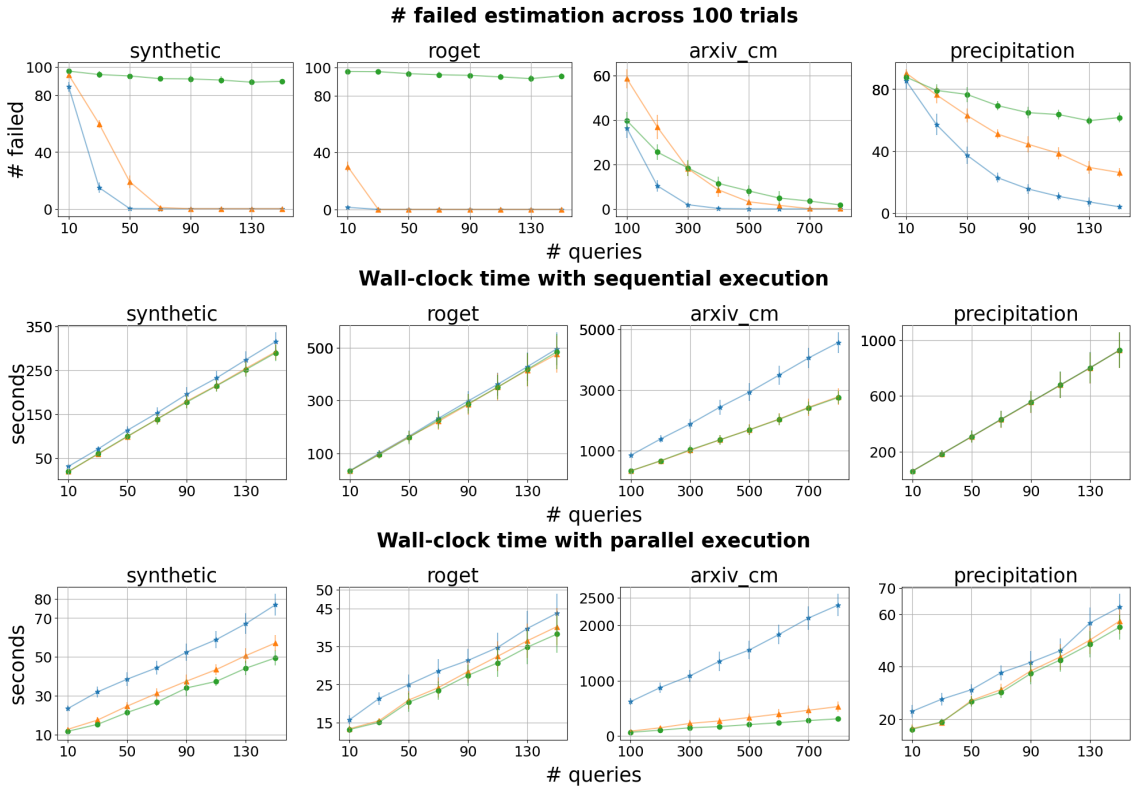


Figure 3.5: Performance comparison of `Hutch++`, `NA-Hutch++` and `Huthinson` over 4 datasets (mean $\pm$ 1 std. across 10 random runs). The approximation error for all settings is set at $\epsilon = 0.01$. Both `Hutch++` and `NA-Hutch++` outperform `Hutchinson` in terms of failed estimates. The parallel version of the non-adaptive `NA-Hutch++` is significantly faster than the adaptive `Hutch++`, making it more practical in real-world applications. *Legend:* `Hutch++` is —★—, `NA-Hutch++` is —▲—, and `Hutchinson` is —●—.

Results and Discussion. The results of `Hutch++`, `NA-Hutch++` and `Hutchinson` over the 4 datasets are presented in **Figure** 3.5. The performance of all algorithms is consistent across different datasets with different matrix-vector oracles, and even on a non-PSD instance from `arxiv_cm`. Given the same number of queries, `Hutch++` and `NA-Hutch++` both give significantly fewer failed estimates than `Hutchinson`, particularly on PSD instances. It is not surprising to see that `Hutchinson` fails to achieve a $(1 \pm \epsilon)$-approximation to the trace most of the time due to the high variance in its estimation, given a small number of queries and a high accuracy requirement ($\epsilon = 0.01$).

For computational costs, the difference in running time of all algorithms is insignificant in our sequential execution. In our parallel execution, however, `Hutch++` becomes significantly slower than the other two, `NA-Hutch++` and `Hutchinson`, which have very little difference in their parallel running time. `Hutch++` suffers from slow running time due to its adaptively

chosen queries, despite the fact that Hutch++ consistently gives the least number of failed estimates.

It is not hard to see that NA-Hutch++ gives the best trade-off between a high success probability in estimating an accurate trace with only a few numbers of queries, and a fast parallel running time due to the use of non-adaptive queries, which makes NA-Hutch++ more practical on large, real-world datasets. We remark that although the Lanczos algorithm is adaptive itself, even with a sequential matrix-vector oracle, our non-adaptive trace estimation can still exploit much more parallelism than adaptive methods, as shown by our experiments.

3.2.8 Conclusion

We determine an optimal $\Theta(\sqrt{\log(1/\delta)}/\epsilon + \log(1/\delta))$ bound on the number of queries to achieve $(1 \pm \epsilon)$ approximation of the trace with probability $1 - \delta$ for non-adaptive trace estimation algorithms, up to a $\log\log(1/\delta)$ factor. This involves both designing a new algorithm, as well as proving a new lower bound. We conduct experiments on synthetic and real-world datasets and confirm that our non-adaptive algorithm has a higher success probability compared to Hutchinson's method for the same sketch size, and has a significantly faster parallel running time compared to adaptive algorithms.

3.3 Task-based Mixture-of-Experts for Multitask Multilingual Transformer-based Models

Mixture-of-Experts (MoE) architecture has been proven a powerful method for diverse tasks in training deep models in many applications, especially when a large-scale deployment is required to serve a huge number of users. However, current MoE implementations are task agnostic, treating all tokens from different tasks in the same manner. In this work, we instead design a novel method that incorporates task information into MoE models at different granular levels with shared dynamic task-based adapters. Our experiments and analysis show the advantages of our approaches over the dense and canonical MoE models on multi-task multilingual machine translations. With task-specific adapters, our models can additionally generalize to new tasks efficiently.

This is the section where the contribution of the work can be put in either efficiency (Chapter 2) or scalability (Chapter 3) of representation learning. However, to conform with the original motivation of MoE revival in the era of deep learning to scale up already-very-large models in industry, this work is put in this chapter to underscore its main purpose of scaling up huge models in training, inference, as well as deployment. That emphasis, however, by no means discount the contribution of the model in incorporating task into MoE architecture, as well as the essential enablers at platform-level infrastructures such as Microsoft Deepspeed [17].

3.3.1 Introduction

Mixture-of-Experts (MoE), while not being a novel machine learning algorithm (Yüksel et al., 2012), has revived to combine with deep learning, particularly transformer (Vaswani et al., 2017) and has recently pushed forward various tasks such as natural language processing, computer vision, speech recognition, multimodal and multitask learning due to its advantage in scalability in distributed environments (Fedus et al., 2022). The main advantages of MoE is stemmed from its ensemble design while maintaining the sparsity in computation (Fedus et al., 2021). And with proper design such as using GShard (Lepikhin et al., 2021), the possibility for enterprise-level scalability is almost boundless. As a result, this method has been more and more widely adopted in many applications that require distributed and intensive workloads.

However, most of the current methods are task-agnostic, only optimizing for performance based on lower levels in the architecture such as at system or communication levels. In the case of multitasks learning where a single model is required to learn from heterogeneous tasks, however, the task-specific data could be inherently diverse and vary largely from one to another. As a result, treating data from such different sources the same makes the learning not effective, as also evidenced recently that the interference between different task data (Pfeiffer et al., 2022).

[17] https://www.microsoft.com/en-us/research/project/deepspeed/

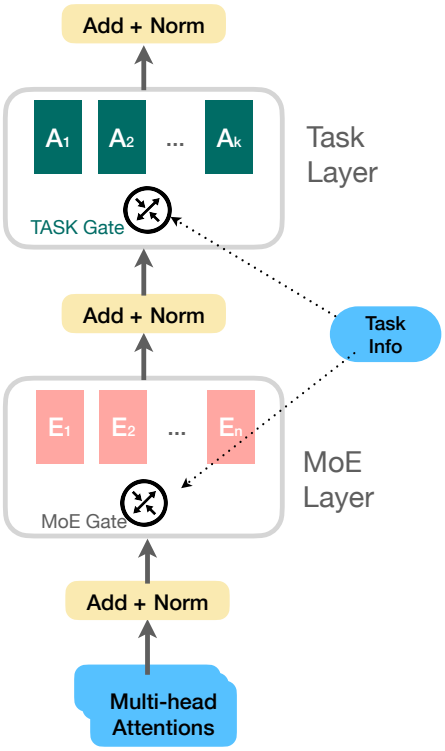


Figure 3.6: Extended from the typical MoE approaches that do not discriminate tokens from different tasks, we create shared task-related adapters that are trained to route tokens from similar tasks to the same shared adapters, and vice versa. Likewise, different task groups should be routed to different adapters, avoiding interference amongst different task data.

As a result, in this work, we design a novel MoE approach where task information is used during training and inference for assigning experts based on individual task information. The intuition is to make the training more task-aware in those similar tasks would be routed to the same group of experts and vice versa. From the architectural perspective, we incorporate high-level application-specific information with the system-level information to make the model become task-aware and hence have a better strategy in allocating experts based on the characteristics of distinct tasks, as also illustrated in Figure 3.6.

Our proposed architecture allows for grouping experts based on the similarity of tasks, i.e. similar tasks should use a similar group of experts and otherwise for different tasks, by using shared-task adapters. Our design of putting those adapters on top of MoE layers allows for flexibility in future extensions: if we want the model to acquire new tasks while still having similar resources, we only finetune new adapters, and if we want to scale the hardware resources, e.g. for more speed, we simply deal with MoE layers with such new resources.

Our experiments and analysis show the advantages of using task information in MoE architectures in multiple settings including multitask multilingual machine translations, as well as its

generalization in few-shot learning. In summary, our contributions are as follows.

- First, we design novel MoE architectures that dynamically allocate experts based on task information in the context of multilingual multitask machine translation, with many variations.
- Second, we thoroughly study the pros and cons of our approaches in training from scratch, finetuning as well as transfer learning.
- Third, we implement our models on top of well-proven infrastructures for practicality and scalability including `deepspeed` (Rasley et al., 2020), `fairseq` (Ott et al., 2019) and `transformer` (Vaswani et al., 2017), and will be releasing our code for public use.

3.3.2 Related Work

MoE Basic Transformer-based Mixture-of-Experts (MoE) architecture essentially sparsifies transformer architecture by replacing the heavy feed-forward network (FFN) with a sparse MoE layer with top-2 gates (Shazeer et al., 2017). However, increasing the number of experts does not simply increase the performance (Clark et al., 2022; Fedus et al., 2021), many approaches have been proposed together to tackle the large-scale MoE deployment, such as in (Kim et al., 2021). In large-scale deployment, however, additional techniques should also be employed to battle with memory issues such as "sharding" experts (Lepikhin et al., 2021) or stabilizing the training (Zoph et al., 2022), since the models are often deployed on separate nodes that mainly used GPUs with limited memory. The architecture in this work inherits all of those techniques, and in addition incorporates task information into MoE routing, which in turn directs data into separate task adapters. This kind of routing is, however, hardware-agnostic, unlike some other work such as in (Chen et al., 2023; Xiong et al.; Zheng et al., 2022).

MoE Routing Techniques Gating is critical to MoE layer, which works as a weighted sum of the experts, and serving for the ultimate purpose of load balancing of all available experts during both training and inference. Unlike the originally proposed top-k experts (Du et al., 2021; Shazeer et al., 2017), it was studied in SwitchTransformer that a single expert can preserve the quality if chosen properly, while significantly reducing the communication and computation cost (Fedus et al., 2021). In more detail, SwitchTransformer first divides evenly amongst all experts with an optional buffer for imbalanced cases, and then applies a auxiliary loss to enforce load balancing. Another alternative approach, which is more computationally efficient is to get rid of such extra-heavy complicated loss and instead use hash function to route every token to its matched expert, which tend to balance the output (Roller et al., 2021). Another interesting approach is to permit each token to appear in the top-k list of multiple experts (Zhou et al., 2022), which has been proven to help, although not applicable for auto-regressive applications. Yet because of the inherent problem of load imbalance, another approach is to replace gating mechanism by a stochastic selection method, which randomly activates an input during processing. The intu-

ition is somewhat similar to the hash approach, since it relies on the "fair" randomness to solve the balance problem while keeping the blueprint more lightweight than enforcing an auxiliary loss. Unlike all of those routing techniques which are application agnostic, our proposed model connects the application level (i.e. task information) with the lower-level MoE layers for better dealing with interference of different tasks in the context of multilingual multitask applications.

Task-level Routing Recently task information has been used for improving MoE, e.g. in (Zhili et al.). Our model is, however, is much simpler and can be trained end-to-end unlike their approach, which requires clustering to be made for off-the-shelf shared representation learning. And probably the most related work to ours is Mod-Squad (Chen et al., 2022) that shares the motivation with us while having several differences. First, their approach which has multiple aids to make the task-based MoE work with an additional loss for regularization, while we instead rely mainly on the simple motivation of incorporating task information into MoE. Second, we still stick to a single gate for routing, while they allocate multiple gates, each *per* task. Third, they additionally have MoE attention blocks, which make their architecture more complicated. Finally, our focused application is text-based machine translation, unlike computer vision settings in those both works mentioned.

3.3.3 Models

Transformer architecture (Vaswani et al., 2017) has been proven to be the core backbone of the pervasive successes in natural language processing, computer vision, and other artificial intelligent fields. The main bottleneck to this architecture is, however, its heavy blueprint leading to intensive resources in training and inference, and is difficult to scale up. MoE is one powerful method to alleviate those problems in transformer.

3.3.3.1 Sparse Mixture-of-Experts (MoE)

MoE, which was first introduced before the deep learning era (Jacobs et al., 1991), was recently borrowed to address those drawbacks in transformer architecture (Shazeer et al., 2017). In a nutshell, MoE creates an ensemble of experts in multi-layer transformer blocks in place of a single expert, typically in the form of a feed-forward neural network (FFN) that is dense with many parameters.

In terms of formality, given an original FFN layer called $\tilde{E}$, we clone it into another layer containing a set of N experts with exactly the same architecture $\{E_i\}_{i=1}^{N}$. Likewise, the number of parameters for this particular layer is increased by a factor of N.

The typical granular level of applying those experts in the context of natural language processing is the token level. Given a token x, its learned representation before MoE layer is a vector $\mathbf{x}$, its post-MoE output $\mathbf{y}$ is the weighted average of those experts' output

$$o_i = E_i(\mathbf{x}) \tag{3.58}$$

$$\mathbf{y} = \sum_{i=1}^{N} W_i o_i, \tag{3.59}$$

where W_i is the weight (importance) of the corresponding expert E_i.

The key to MoE power and its well-proven successes in tandem with transformer architecture is its sparsity design: only one or few experts are activated (i.e. having non-zero weight) at any point in time in spite of many more parameters just introduced due to the ensemble. Typically the component responsible for this sparsity is a gate that was co-trained with experts to route tokens to their target expert(s), and eventually assigns only a single or few non-zero weights across all experts *per* token to its output $G(\mathbf{x})$ typically using softmax and top-k method

$$g_{out} = \text{softmax}\left(W_g \mathbf{x}\right) \tag{3.60}$$

$$G(\mathbf{x}) = \text{Top\_K}\left(g_{out}\right) \tag{3.61}$$

With $G(\mathbf{x})$ being a set of K chosen experts, equation 3.59 becomes

$$\mathbf{y} = \sum_{i \in G(\mathbf{x})} W_i o_i \tag{3.62}$$

The main architectural problem with this design is its scalability: the memory will be quickly used up as we increase experts, given the limitation of current compute resources allocated to a single compute node in any distributed environment. GShard (Lepikhin et al., 2021) was born to fix this issue by trading the memory for communication: allocating each expert to a single node and only aggregating them when needed, e.g. gradient averaging in training or weight averaging when saving a model. This elegant design has unlocked MoE's unlimited scalability and practicality in enterprise-level deployments, especially with the following-up work in optimizing for better architecture in computation and communication, as mentioned in Section 3.3.2.

3.3.3.2 Task-based Adapters

Yet another problem on which we are focusing is not at the system level but more at the higher application level. As mentioned, in the multitask setting, the interference amongst tasks that are inherently different from each other could lead to the ineffectiveness of training. As a result, we employ *task-based adapters* to separate those different tasks to different adapters. Likewise, data (or tokens) from similar tasks should be routed to a similar group of adapters. There are three modes for those adapters.

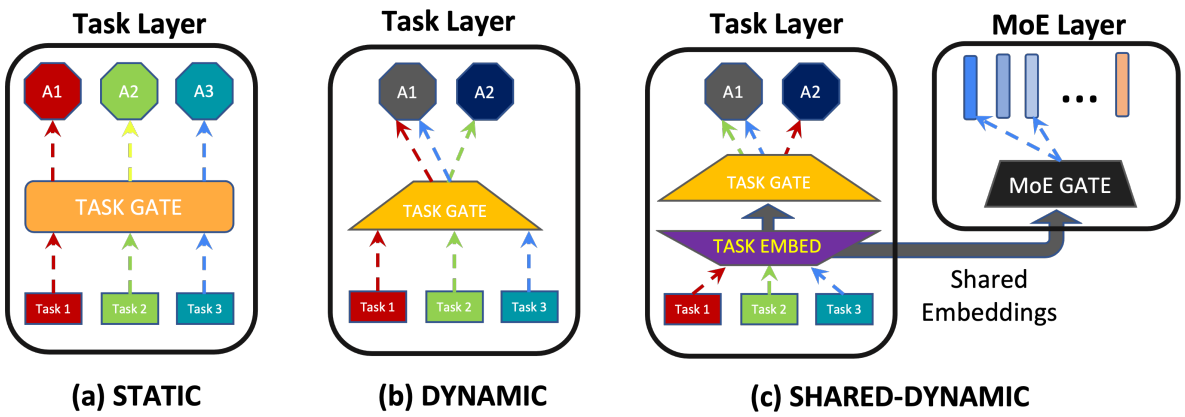


Figure 3.7: MoE models with variants. (a) `Static` means for each task, there is a separate adapter associated with it. (b) In the `Dynamic` mode, there is less number of adapters than the number of tasks, in order to learn the shared representation of similar tasks. (c) The last variant is `Shared-Dynamic` where the gates for task adapters and MoE share the same embedding for their routing decisions.

First and the simplest is to allocate each adapter for each individual task. Although this setting is straightforward and requires no additional computation for data routing, it has the drawback of acquiring new unseen tasks. The reason is the model has to allocate a new adapter for each new task and fine-tune it with some amount of new data. Another potential problem is memory limitation if we want to extend to many new tasks in the future. This mode is called `static adapters`, as shown in Figure 3.7a.

To enforce efficient learning of representation of similar task data, as well as alleviating memory problems, we have `dynamic adapters` (Figure 3.7b) where the number of adapters is less than the number of tasks. As a result, we intentionally guide the model to learn better cross-task representation in terms of similarity and dissimilarity. In other words, data from similar tasks should be routed to the same adapters and vice versa. In practice, we choose the number of adapters to be $\log_2(n)$ with n being the number of tasks.

3.3.3.3 Task-based Adapters with MoE

In this section, we formulate the task-based adapters mentioned in Section 3.3.3.2 in combination with MoE, both of which are our core architecture components.

Given M tasks, we allocate them into L shared-task adapters ($L < M$). For every single token x, we have the associated task information t that makes up an input tuple $(\mathbf{x}, \mathbf{t})$ per token. As before, $\mathbf{x}$ is the representation vector from input, and $\mathbf{t}$ is the task representation vector learned by task embedding.

Similarly to MoE, we use a learnable task gate G_t that is responsible for this routing with input being the concatenation of the input components

		Task Data								
Split	Unit	de-en	fr-en	cs-en	et-en	fi-en	gu-en	hi-en	lv-en	ro-en
Training	M	4.6	10	10.3	0.7	4.8	0.9	0.3	1.4	0.5
Validation	K	3.0	3.0	3.0	2.0	1.4	2.0	0.5	2.0	2.0
Testing	K	3.0	3.0	3.0	2.0	1.4	2.0	0.5	2.0	2.0

Table 3.2: Training, Validation, and Testing sizes for all XE tasks (the data for EX are exactly the same). Note that the unit for training is million (M) while that for both validation and testing are thousand (K), and the sizes are the same for validation and testing.

$$G_t(x, t) = \text{Top\_K}(\mathbf{x} \oplus \mathbf{t}) \tag{3.63}$$

$$\mathbf{y} = \sum_{i \in G_t(\mathbf{x}, \mathbf{t})} W_i o_i \tag{3.64}$$

And since the number of adapters $L < M$, the number of tasks, we call this setting `dynamic task adapters`, as demonstrated in Figure 3.7b, as opposed to `static task adapters` (Figure 3.7a), where each task will go to each individual adapter.

Our main model use the shared task embedding representation for the task gate as well as MoE gate, which we call `shared-dynamic task adapters`, as shown in Figure 3.7c.

3.3.4 Experiment Setup

3.3.4.1 Data

We tackle the problem of multitask multilingual machine translation using the data consisting of 10 different languages ranging from high-resources to low-resource ones including English (En), French (Fr), German (De), Czech (Cs), Finnish (Fi), Latvian (Lv), Estonian (Et), Romanian (Ro), Hindi (Hi), Turkish (Tr), and Gujarati (Gu). In more detail, the data for training, validation, and testing are listed in Table 3.2 where we can see besides the high-resource ones, we have low-resource languages such as Estonian, Hindi, or Gujarati.

Those data are in the form of Bitext in which there is always English. As a result, we denote EX as the translation from English (E) to another language (X), and similarly for the other way around, XE. Those data are populated from the popular WMT corpus [18]. For the given 1 English and 9 other languages, there are consequently 9 EX and 9XE tasks.

[18] https://www.statmt.org/wmt20/index.html

3.3.4.2 Task and Model Training

In this section, we describe the task information, evaluation metrics, and how we deal with data and models for training.

Task Our task is multi-task multilingual machine translation (MMMT) which use the EX and XE pairs. Our single model is trained with two main capacities. First, this single model can translate all the training pairs with high accuracy. Second, the model is able to quickly acquire new translation pairs with only zero or a few shots.

Evaluation While there are many evaluation metrics, we mainly use BLEU score due to its popularity and credibility in evaluating machine translation tasks. This evaluation is implemented by SacreBLEU. [19]. We note that unlike all available public implementations that we found, our implementation evaluates all BLEU scores on-the-fly along with the training, so there is no disruption for offline evaluation. That also helps in early stopping based on the BLEU scores on the validation sets.

Pre-Processing and Post-Processing In terms of preprocessing, we first encode the data using Byte-Pair encoding (BPE) method and generate shared dictionaries where all the language pairs use the same vocabulary of size 64K, before feeding to the model. To get accurate scores, for post-processing, we again use BPE decoding for reconstructing the whole translated sentences before comparing them with the original sentences before BPE pre-processing. Likewise, we treat all the processing and model manipulation as a black box for calculating the scores.

Model Configuration and Implementation We use transformer architecture (Vaswani et al., 2017) with 12 layers for both encoder and decoder phases, each of which uses a word embedding layer of dimension 1024 and a non-linear layer of dimension 4096. There are 16 attention heads and a dropout rate of 30%. For MoE, all jobs are trained on Azure cloud machines with 8 GPUs, each of which takes around 2 weeks for a model covering 18 aforementioned tasks to reach decent scores. We apply early stopping based on the validation BLEU scores, in which a non-increasing score after 2 epochs is the condition. For task-based information, we have a task embedding dimension of 64 and a task adapter hidden dimension of 256 for every single task adapter. Our implementation inherits the lower-level infrastructure code from Microsoft Deepspeed and Fairseq. [20]

As for the implementation, an important practical issue with MoE is load balancing among experts for the best utilization of the infrastructure systems. For enforcing the training to have a balanced load, as a result, we employ the auxiliary loss from Lepikhin et al. (2021).

3.3.4.3 Baselines

In order to show the performance of the task-based MoE models, the following baselines are selected:

[19] https://github.com/mjpost/sacrebleu
[20] https://github.com/facebookresearch/fairseq

Model	XE Tasks									
	de-en	fr-en	cs-en	et-en	fi-en	gu-en	hi-en	lv-en	ro-en	**Average**
1. Dense	29.9	31.2	28	22.4	21.4	22.3	21.4	24.5	36.1	26.4
2. MoE Token	27.9	29.5	26.3	19.9	19.6	18.9	17.7	22.3	33.8	24.0
3. MoE Sentence	27.9	29.9	26.2	21.4	19.9	17.9	15.9	23.2	34.4	24.1
4. MoE Task-Static	**32.1**	**33.3**	**30.7**	**24.3**	**23.4**	20.6	**22.5**	**27.2**	**38.8**	**28.1**
5.MoE Task-Dynamic	**31.4**	**32.0**	**29.1**	**23.4**	**22.1**	18.9	20.5	**25.5**	**37.2**	**26.7**
	EX Tasks									
	en-de	en-fr	en-cs	en-et	en-fi	en-gu	en-hi	en-lv	en-ro	
1. Dense	25.4	28.3	22.4	23.3	20.9	28.4	29.0	26.5	31.5	26.2
2. MoE Token	22.9	25.1	19.5	20.1	17.9	26.2	26.3	24.0	29.0	23.4
3. MoE Sentence	23.2	25.7	20.4	22.4	18.7	26.4	27.1	24.2	29.7	24.2
4. MoE Task-Static	**29.5**	**32.5**	**27.9**	**27.4**	**25.8**	**28.8**	**30.8**	**32.2**	**34.6**	**29.9**
5.MoE Task-Dynamic	**27.3**	**29.6**	**25.0**	**24.7**	**22.7**	27.7	**29.3**	**28.4**	**32.7**	**27.5**

Table 3.3: Performance comparison of task-based MoE models (models 4 & 5) to task-agnostic MoE models (models 2 & 3) and the non-MOE (Dense) model (model 1) in BLEU scores. With the help of task information, task-based MoE models show their outperforming BLEU scores over all other types across most of the tasks including both high-resource and low-resource ones.

Dense This is the traditional transformer model without any MoE layer, i.e., no change to the fully connected (FFN) layer in each layer of encoders or decoders.

MoE - Token This is the MoE model that is usually considered the default option where each FFN layer is replaced by an MoE layer. In our experiments, each MoE layer comprises 8 experts (each has the same size as the original FFN being replaced) and a gate for routing purposes.

MoE - Sentence This is yet another MoE architecture with exactly the same architecture configuration as the MoE - Token baseline. The difference is in the routing layer, which functions at a different granularity: sentences instead of tokens. In more detail, while the gate decides which expert for each token separately in MoE - Token model, it instead routes all tokens belonging to a single sentence to the same chosen expert.

3.3.5 Results and Discussions

3.3.5.1 Multitask Multilingual Machine Translation

We first present the main results for models capable of translating 18 tasks (see Section 3.3.4.2) concurrently. As shown in Table 3.3, our models that incorporate MoE layers and are enhanced with task information show great advantages over all the baseline models on most tasks, in both directions EX and XE, in accordance with our hypothesis that using task adapters in conjunction

Model	Design		Routing		Tasks				Average
	MoE	Task	MoE	Task	de-en	fr-en	et-en	fi-en	
MoE	Y	N	Token	-	32.4	33.7	24.2	23.6	28.5
Dense + Task Static	N	Y	Task	Static	32.2	33.7	21.0	22.8	27.4
Dense + Task Dynamic				Dynamic	31.9	33.0	22.0	22.5	27.4
MoE + Task Static	Y	Y		Static	30.7	32.0	19.9	20.8	25.9
MoE + Task Dynamic				Dynamic	**32.6**	**33.9**	24.0	**23.9**	**28.6**
MoE + Task Shared-Dynamic				Shared-Dynamic	32.2	33.3	**24.3**	**24.5**	**28.6**

Table 3.4: Performance of different models with changes on whether MoE layers exist, whether Task Adapters exist, and how routing for those components is undertaken. The scores better than the baseline are highlighted. Task-based MoE shows advantages, especially with shared-dynamic adapters between MoE and Task Adapters on the low-resource language pair.

with MoE is helpful in multilingual multitask translation.

An outstanding drawback with which the task-based MoE models are facing, however, is for the low-resource translation pairs, e.g. Gu-En, Hi-En, or En-Gu. We hypothesize the problem is due to the undersampling of the training data. Our training routine concatenates all the tasks' data in a single big dataset before drawing batches. However, without adjusting the sampling process, high-resource language pairs are being trained significantly more given their much larger data in place. In particular, for the case of Gujarati where the Task-Dynamic MoE model underperforms in comparison to the baselines, our hypothesis is that linguistically, this language is the most different from all other languages, which makes the models very hard to learn effective shared representation with any other pairs.

3.3.5.2 Ablation Study

In this study, we limit the number of tasks to four (De-En, Fr-En, Et-En, and Fi-En) to study the performance implications of different model variants when there is a task layer and/or MoE layer.

As illustrated in Table 3.4, we again see that combining MoE and Task Adapters yields the best models, the same trend as shown in Table 3.3, particularly when the dynamic adapters are used to enforce similar tasks to share the same representations.

However, when task adapters are not used in conjunction with MoE, the performance is worse than MoE alone. This also means MoE should be the foundational infrastructure, and on top of that, task adapters should be used. It concurs with the motivation that the interference of different tasks or languages makes the training of experts difficult. In other words, there is not so much help when there is only one expert for all the tasks (i.e. Dense models).

3.3.6 Conclusion

In the era of big data, large-scale models are more and more essential to big enterprises and institutions, where MoE in combination with transformer-based models have been proven its great advantages very recently. It is, however, complicated to enable that implementation in practice due to the difficulties of training a single model serving diverse tasks. The proposed task-based MoE, which employs both task adapters in tandem with MoE has shown its promising advantages in the application of multitask multilingual machine translations. This novel design enforces shared representation of similar tasks and separate different task data to counter the interference effects. In addition, it also offers the flexibility of changing adapters based on new tasks or changing the MoE infrastructure without affecting the application level. In the future, enforcing the shared representation learning explicitly using such additional techniques as contrastive learning or mutual information is also worth exploring.

Chapter 4

Conclusion and Future Work

Regardless of the type of problems in machine learning, one has to connect the given data with the targeted objective with a model that is able to learn useful representation from such data. Representation learning, consequently, is a broad topic that spans the whole field of machine learning, and artificial intelligence in general. And because of its nature as a being an overly expansive topic, this thesis is not ambitious to cover all possible aspects of representation learning. And in fact, it is probably impossible to do so, especially given the immensely fast-moving speed of the field.

This thesis, however, has explored and made contributions to the two prominent topics in representation learning, which are task-oriented efficiency and practical scalability. The applications–and the described works–that are used to study those two topics are unquestionably limited, yet cover diverse problems and data. The following sections will summarize the elaborated works in the main content of this thesis, with a highlight on their contributions and possible exploring directions in the future.

4.1 Thesis Contributions

This thesis makes several contributions in each of the two focuses in representation learning, namely efficiency and scalability, as the following sections will summarize.

4.1.1 Efficient Representation Learning

Throughout the thesis, it is shown that given knowledge about the task and the data itself, one can design an efficient model to learn the representation from that data to solve the task with high accuracy.

Chapter 2.1 introduces a novel approach to learning multimodal representation by borrowing machine translation to apply to cross-domain data. While a normal translation happens between texts of different languages, it can also be done efficiently between two different domains such

as between text and audio, or text and videos. This cross-domain type of translation is also strengthened by cyclic consistency loss, to help the representation learning to learn the translation symmetrically in both ways.

Chapter 2.2 tackles real-world data and a problem where the data is tampered with noise due to the scanning process, let alone it is very limited to be used in training deep models, thus making it very different from any available public datasets (and their successful models that can be borrowed). Given such challenges, the solution has to be also different. On one hand, new data annotation is needed with a focus on the helpful content (to limit the number of classes, which helps the model to learn faster), and so is various augmentation to enrich training data. On the other hand, a novel approach is introduced that is modularized with a multi-stage approach that is largely adapted from object detection and recognition in computer vision with enhancement from other fields such as CTC loss from speech recognition. Same as Chapter 2.1, this is yet another exemplar where traditional techniques in a field can be adapted and make novel contributions in a new field if applied properly.

Chapter 2.3 deals mainly with the expensive cost of attention in transformer-based architectures in a non-standard type of input that has multimodal documents where the content is long and the format is diverse. This thesis designs a novel technique where multimodal information can be used directly into attention, with an option to combine different modalities in separate phases. In addition, the reduced context with sparse global attention applied to all modalities helps make the approach efficient and also scalable in achieving new input length limits with state-of-the-art accuracy.

4.1.2 Scalable Representation Learning

Chapter 3.1 tackles a common problem in practice concerning Gaussian Processes (GP), where the training cost is cubic, making it the main hindrance to its popularity in dealing with high-dimensional data and practical problems in the real world. The novel contribution is to apply a sparse spectrum approach for GP with provable guarantees on the sample complexity, enhanced by a novel data clustering on the embedded representation of the data that facilitates the training with much saving in training and inference cost.

Chapter 3.2 deals with another popular application in practice: estimating a matrix trace, which is very challenging in high dimensional space such as neural networks. The thesis newly improves the query sample complexity theoretically and empirically shows that the non-adaptive estimation approach is more practical in the real world where the accuracy is high while having parallelization capabilities.

Chapter 3.3 is similar to Chapter 2.3 since each work covers both of this thesis focuses: efficiency and scalability of representation learning, and both use the increasingly popular transformer models that are heavy and thus always demand novel techniques satisfying those two focuses. Given the task of multitask learning for multilingual machine translation, this thesis

proposes the use of Mixture-of-Experts with "sharded-expert" distribution, enhanced by a novel use of task-embedded representation. This novel approach not only helps in facilitating the complicated routing with application-level information being utilized at the infrastructure level that is typically application-agnostic but also with separate task adapters the shared representation is learned efficiently to solve current tasks. Those shared adapters also unlock new capabilities in acquiring novel unseen tasks that have similar data to existing ones.

4.2 Future Directions

As shown in this thesis, there are several ways of designing efficient and scalable representation learning techniques. Current works, in addition, can be extended in meaningful ways with borrowing new techniques or mixing current methods in this thesis. For example, the multimodal sentiment analysis model 2.1 can also be extended into transformer instead of the older Seq2Seq models without affecting much of the main approach. Even more so, it can also be enhanced by approximation techniques for transformer-based models such as in Chapter 2.3 or Chapter 3.3 for gaining more scalability. Also, our practical model for long document understanding (Chapter 2.3) can be easily extended to vision input besides layout and text without affecting much the architecture. With that additional input, we can optionally borrow the cross-domain translation for learning multimodal aligned representation before feeding it to the transformer and thus can potentially reduce the computation significantly. In another example of Mixture-of-Experts (Chapter 3.3), the task-aware model can be enhanced by additional techniques such as using contrastive learning or mutual information approach for facilitating the similar and dissimilar representation of the task data. The transformer attention in this work or in document understanding, which is the main cost of this type of architecture, can borrow the sketching techniques such as from Chapter 3.1 or Chapter 3.2 to reduce this main cost while keeping hight accuracy.

Although it is not directly mentioned, domain knowledge and expertise are by no means less important–quite the opposite. For example, without deep knowledge of the nature of data given in Chapter 2.2, one can not easily design an effective approach for annotation and augmentation that facilitate the training. Or in Chapter 3.3, domain knowledge of multilingual machine translation greatly helps in debugging Mixture-of-Experts and Task Adapters with domain knowledge of what tasks are similar and what are not. All in all, it is always recommended to acquire domain knowledge given any problem, and such an essential phase could in turn save the cost immensely.

Finally, throughout the whole of this thesis, two central aspects of representation learning–efficiency and scalability–are underscored to be the main factors in having a proper and practical approach to a diverse set of data and problems. As we can also see, e.g. in Chapters 3.2 and 3.3, those two focuses are not necessarily conflicting but realistically, they can complement by inheriting the other's advantages in an integral solution. In fact, this thesis, by its diverse works, infers that a real-world solution should possess both aspects in a single model. As a result, ones have

to take both targets into account when designing a model in practice.

Bibliography

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. 2020. ETC: Encoding Long and Structured Inputs in Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 268–284. 2.3.2

Paavo Alku. 1992. Glottal wave analysis with pitch synchronous iterative adaptive inverse filtering. *Speech Communication*, 11(2-3):109–118. 2.1.4.2

Paavo Alku, Tom Bäckström, and Erkki Vilkman. 2002. Normalized amplitude quotient for parametrization of the glottal flow. *the Journal of the Acoustical Society of America*, 112(2):701–710. 2.1.4.2

Paavo Alku, Helmer Strik, and Erkki Vilkman. 1997. Parabolic spectral parameter - a new method for quantification of the glottal flow. *Speech Communication*, 22(1):67–79. 2.1.4.2

Rakshit Allamraju and Girish Chowdhary. 2017. Communication efficient decentralized Gaussian Process fusion for multi-UAS path planning. In *American Control Conference*, pages 4442–4447. 3.1.1

T. J. Ansell et al. 2006. Daily Mean Sea Level Pressure Reconstructions for the European-North Atlantic Region for the Period 1850-2003. *J. Climate*, 19(12):2717–2742. 3.1.1

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha. 2021. DocFormer: End-to-End Transformer for Document Understanding. *arXiv preprint arXiv:2106.11539*. 2.3.1, 2.3.1, 2.3.2, 2.3.4.3

H. Avron, M. Kapralov, C. Musco, C. Musco, A. Velingker, and A. Zandieh. 2017. Random Fourier Features for Kernel Ridge Regression: Approximation Bounds and Statistical Guarantees. In *Proceeding of ICML*, pages 253–262. 3.1.2.2, 3.1.3, 3.1.3.1, 3.1.3.2, 6

Haim Avron. 2010. Counting triangles in large graphs using randomized matrix trace estimation. 3.2.1

Haim Avron and Sivan Toledo. 2011. Randomized Algorithms for Estimating the Trace of an Implicit Symmetric Positive Semi-Definite Matrix. *J. ACM*, 58(2). 3.2.2

Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. 2019. Character

Region Awareness for Text Detection. In *CVPR*, pages 9365–9374. 2.2.4.4

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*. 2.1.3.3

Claus Bahlmann and Hans Burkhardt. 2004. The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic time warping. *IEEE Transactions on pattern analysis and machine intelligence*. 2.2.2

Tadas Baltrusaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2017. Multimodal machine learning: A survey and taxonomy. *CoRR*, abs/1705.09406. 2.1.2

Irwan Bello. 2021. Lambdanetworks: Modeling long-range interactions without attention. *arXiv preprint arXiv:2102.08602*. 2.3.2

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*. 2.3.2, 2.3.3.1, 2.3.3.3, 2.12, 2.3.4.2, 2.3.4.4

Christos Boutsidis, David P. Woodruff, and Peilin Zhong. 2016. Optimal principal component analysis in distributed and streaming models. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, page 236–249, New York, NY, USA. Association for Computing Machinery. 3.2.1, 3.2.3

Vladimir Braverman, Stephen R. Chestnut, Robert Krauthgamer, Yi Li, David P. Woodruff, and Lin F. Yang. 2018. Matrix Norms in Data Streams: Faster, Multi-Pass and Row-Order. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 648–657. 3.2.1

Vladimir Braverman, Robert Krauthgamer, Aditya Krishnan, and Roi Sinoff. 2020. Schatten Norms in Matrix Streams: Hello Sparsity, Goodbye Dimension. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, pages 1100–1110. 3.2.1

Leo Breiman. 2001. Random Forests. *Machine Learning*, 45(1):5–32. 2.1.4.5

Horst Bunke and Tamás Varga. 2007. Off-line Roman cursive handwriting recognition. In *Digital Document Processing*, pages 165–183. Springer. 2.2.2

Javier Burgues. Gas Sensor Array Temperature Modulation Dataset. 3.2, 3.1.4, 3.4

Javier Burgués, Juan Manuel Jiménez-Soto, and Santiago Marco. 2018. Estimation of the limit of detection in semiconductor gas sensors through linearized calibration models. *Analytica chimica acta*, 1013:13–25. 3.1.4

Javier Burgués and Santiago Marco. 2018. Multivariate estimation of the limit of detection by orthogonal partial least squares in temperature-modulated MOX sensors. *Analytica chimica acta*, 1019:49–64. 3.1.4

Lei Cai, Zhengyang Wang, Hongyang Gao, Dinggang Shen, and Shuiwang Ji. 2018. Deep Adver-

sarial Learning for Multi-Modality Missing Data Completion. In *KDD '18*, pages 1158–1166. 2.1.2

Nannan Cao, Kian Hsiang Low, and John M. Dolan. 2013. Multi-Robot Informative Path Planning for Active Sensing of Environmental Phenomena: A Tale of Two Algorithms. In *Proceeding of AAMAS*, pages 7–14. 3.1.1

Linda Chamakh, Emmanuel Gobet, and Zoltán Szabó. 2020. Orlicz Random Fourier Features. *Journal of Machine Learning Research*, 21:145:1–145:37. 3.1.2.2

Chang-Qin Chen, Min Li, Zhihua Wu, Dianhai Yu, and Chao Yang. 2023. TA-MoE: Topology-Aware Large Scale Mixture-of-Expert Training. *ArXiv*, abs/2302.09915. 3.3.2

J. Chen, N. Cao, K. H. Low, R. Ouyang, C. K.-Y. Tan, and P. Jaillet. 2013a. Parallel Gaussian Process Regression with Low-Rank Covariance Matrix Approximations. In *Proceeding of UAI*, pages 152–161. 3.1.1

J. Chen, K. H. Low, C. K.-Y. Tan, A. Oran, P. Jaillet, J. M. Dolan, and G. S. Sukhatme. 2012. Decentralized Data Fusion and Active Sensing with Mobile Sensors for Modeling and Predicting Spatiotemporal Traffic Phenomena. In *Proceeding of UAI*, pages 163–173. 3.1.1

Jie Chen, Kian Hsiang Low, and Colin Tan. 2013b. Gaussian Process-Based Decentralized Data Fusion and Active Sensing for Mobility-on-Demand System. *Robotics: Science and System*. 3.1.1

Minghai Chen, Sen Wang, Paul Pu Liang, Tadas Baltrušaitis, Amir Zadeh, and Louis-Philippe Morency. 2017. Multimodal Sentiment Analysis with Word-Level Fusion and Reinforcement Learning. *ICMI*. 2.1.2

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *European Conference on Computer Vision*, pages 104–120. Springer. 2.3.2

Z. Chen, Yikang Shen, Mingyu Ding, Zhenfang Chen, Hengshuang Zhao, Erik G. Learned-Miller, and Chuang Gan. 2022. Mod-Squad: Designing Mixture of Experts As Modular Multi-Task Learners. *ArXiv*, abs/2212.08066. 3.3.2

Yong Cheng, Fei Huang, Lian Zhou, Cheng Jin, Yuejie Zhang, and Tao Zhang. 2017. A Hierarchical Multimodal Attention-based Neural Network for Image Captioning. In *SIGIR '17*. 2.1.2

H. Chernoff. 1952. A measure of asymptotic efficiency for tests of hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–509. 3.1.3.1

Donald G Childers and CK Lee. 1991. Vocal quality factors: Analysis, synthesis, and perception. *the Journal of the Acoustical Society of America*, 90(5):2394–2410. 2.1.4.2

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. 2020. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*. 2.3.2

Aidan Clark, Diego de las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, et al. 2022. Unified Scaling Laws for Routed Language Models. *arXiv preprint arXiv:2202.01169*. 3.3.2

Kenneth L. Clarkson and David P. Woodruff. 2009. Numerical Linear Algebra in the Streaming Model. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 205–214, New York, NY, USA. Association for Computing Machinery. 3.2.3, 3.2.5.2

Guillem Collell, Ted Zhang, and Marie-Francine Moens. 2017. Imagined Visual Representations as Multimodal Embeddings. *AAAI*. 2.1.2

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297. 2.1.4.5

Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. 2016. R-FCN: Object Detection via Region-based Fully Convolutional Networks. In *NIPS*, pages 379–387. 2.2.1, 2.2.2, 2.2.3.2, 2.7, 2.2.5.1, 2.11, 2.2.5.2, 2.9

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *arXiv preprint arXiv:1901.02860*. 2.3.2

Fred J Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176. 2.2.4.3

Gilles Degottex, John Kane, Thomas Drugman, Tuomo Raitio, and Stefan Scherer. 2014. COVAREP - A collaborative voice analysis repository for speech technologies. In *ICASSP*. IEEE. 2.1.4.2

Dan Deng, Haifeng Liu, Xuelong Li, and Deng Cai. 2018. Pixellink: Detecting scene text via instance segmentation. In *AAAI*. 2.2.4.4, 2.7, 2.11

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv*, abs/1810.04805. 1.3, 2.3.4.4

Ilias Diakonikolas, Themis Gouleakis, Daniel M. Kane, John Peebles, and Eric Price. 2020. Optimal Testing of Discrete Distributions with High Probability. *CoRR*, abs/2009.06540. 3.2.1

Siyu Ding, Junyuan Shang, Shuohuan Wang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE-DOC: The Retrospective Long-Document Modeling Transformer. *arXiv preprint arXiv:2012.15688*. 2.3.2

J. M. Dolan, G. Podnar, S. Stancliff, K. H. Low, A. Elfes, J. Higinbotham, J. C. Hosler, T. A. Moisan, and J. Moisan. 2009. Cooperative Aquatic Sensing using the Telesupervised Adaptive Ocean Sensor Fleet. In *Proceeding of SPIE Conference on Remote Sensing of the Ocean, Sea Ice, and Large Water Regions*, volume 7473. 3.1.1

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. 2016. Adversarial Feature Learning. *arXiv*

preprint arXiv:1605.09782. 2.1.2

Kun Dong, David Eriksson, Hannes Nickisch, David Bindel, and Andrew G Wilson. 2017. Scalable Log Determinants for Gaussian Process Kernel Learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc. 3.2.7

Thomas Drugman and Abeer Alwan. 2011. Joint Robust Voicing Detection and Pitch Estimation Based on Residual Harmonics. In *Interspeech*, pages 1973–1976. 2.1.4.2

Thomas Drugman, Mark Thomas, Jon Gudnason, Patrick Naylor, and Thierry Dutoit. 2012. Detection of glottal closure instants from speech signals: A quantitative review. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(3):994–1006. 2.1.4.2

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. 2021. GLaM: Efficient Scaling of Language Models with Mixture-of-Experts. *arXiv preprint arXiv:2112.06905*. 3.3.2

John Duchi. Derivations for Linear Algebra and Optimization. 3.2.5

John Duchi. 2021. Lecture Notes for Statistics 311/Electrical Engineering 377. 3.2.7

Kartik Dutta, Praveen Krishnan, Minesh Mathew, and CV Jawahar. 2018. Improving CNN-RNN Hybrid Networks for Handwriting Recognition. In *ICFHR*, pages 80–85. IEEE. 2.2.2

Ernesto Estrada and Naomichi Hatano. 2008. Communicability in complex networks. *Phys. Rev. E*, 77:036111. 3.2.1, 3.2.7

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2010. The PASCAL Visual Object Classes (VOC) Challenge. *International journal of computer vision*, 88(2):303–338. 2.2.4.3

William Fedus, Jeff Dean, and Barret Zoph. 2022. A Review of Sparse Expert Models in Deep Learning. *ArXiv*, abs/2209.01667. 3.3.1

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *arXiv preprint arXiv:2101.03961*. 3.3.1, 3.3.2

Paraskevi Fika and Christos Koukouvinos. 2017. Stochastic estimates for the trace of functions of matrices via Hadamard matrices. *Communications in Statistics-Simulation and Computation*, 46(5):3491–3503. 3.2.1

Andreas Fischer, Andreas Keller, Volkmar Frinken, and Horst Bunke. 2012. Lexicon-free handwritten word spotting using character HMMs. *Pattern Recognition Letters*. 2.2.2

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*. 2.3.4.3

Yarin Gal and Richard Turner. 2015. Improving the Gaussian process sparse spectrum approximation by representing uncertainty in frequency inputs. 3.1.2.2

Yarin Gal, Mark van der Wilk, and Carl E. Rasmussen. 2014. Distributed Variational Inference in Sparse Gaussian Process Regression and Latent Variable Models. In *Proceeding of NIPS*. 3.1.1

Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. 2019. An Investigation into Neural Net Optimization via Hessian Eigenvalue Density. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2232–2241. PMLR. 3.2.1, 3.2.7

Anna C. Gilbert, Hung Q. Ngo, Ely Porat, Atri Rudra, and Martin J. Strauss. 2013. L2/L2-foreach sparse recovery with low risk. *CoRR*, abs/1304.6232. 3.2.1

Ross B. Girshick. 2015. Fast R-CNN. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448. 2.2.3.2

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in neural information processing systems*, pages 2672–2680. 1, 2.1.2

Filip Graliński, Tomasz Stanisławek, Anna Wróblewska, Dawid Lipiński, Agnieszka Kaliska, Paulina Rosalska, Bartosz Topolski, and Przemysław Biecek. 2020. Kleister: A novel task for Information Extraction involving Long Documents with Complex Layout. *arXiv preprint arXiv:2003.02356*. 2.3.4.1, 2.3.4.3

A. Graves, A. r. Mohamed, and G. Hinton. 2013. Speech recognition with deep recurrent neural networks. In *ICASSP*. 2.1.4.5

Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. 2009. A Novel Connectionist System for Unconstrained Handwriting Recognition. *IEEE Transactions on pattern analysis and machine intelligence*. 2.2.1, 2.2.3.3

Patrick J Grother and Kayee K Hanaoka. 1995. NIST Special Database 19 Handprinted Forms and Characters Database. *Handprinted forms and characters database, National Institute of Standards and Technology*. 2.2.1

Jean-Philippe Thiran Guillaume Jaume, Hazim Kemal Ekenel. 2019. FUNSD: A Dataset for Form Understanding in Noisy Scanned Documents. In *Accepted to ICDAR-OST*. 2.3.4.1

Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. 2016. Synthetic Data for Text Localisation in Natural Images. In *CVPR*, pages 2315–2324. 2.2.4.4

Adam W. Harley, Alex Ufkes, and Konstantinos G. Derpanis. 2015. Evaluation of deep convolutional nets for document image classification and retrieval. *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 991–995. 2.13, 2.15, 2.3.4.1

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask R-CNN. *arXiv preprint arXiv:1703.06870*. 2.2.2

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for

Image Recognition. In *CVPR*, pages 770–778. 2.2.3.3, 3.2.1

J. Hensman, N. Fusi, and N. D. Lawrence. 2013. Gaussian Processes for Big Data. In *Proceeding of UAI*, pages 282–290. 3.1.1

Felix Hill, Roi Reichart, and Anna Korhonen. 2014. Multi-Modal Models for Concrete and Abstract Concept Meaning. *TACL*. 2.1.2

M. Hoang and C. Kingsford. 2020. Optimizing Dynamic Structures with Bayesian Generative Search. In *Internation Conference on Machine Learning*. 3.1.1

Q. M. Hoang, T. N. Hoang, and K. H. Low. 2017. A Generalized Stochastic Variational Bayesian Hyperparameter Learning Framework for Sparse Spectrum Gaussian Process Regression. In *Proceeding of AAAI*, pages 2007–2014. 3.1.1, 3.1.2.2, 3.1.2.2

Q. M. Hoang, T. N. Hoang, K. H. Low, and C. Kingsford. 2019a. Collective Model Fusion for Multiple Black-Box Experts. In *Proceeding of ICML*. 3.1.1

Quang Minh Hoang, Trong Nghia Hoang, Hai Pham, and David P Woodruff. 2020. Revisiting the Sample Complexity of Sparse Spectrum Approximation of Gaussian Processes. *arXiv preprint arXiv:2011.08432*. 1.3

T. N. Hoang, Q. M. Hoang, and K. H. Low. 2015. A Unifying Framework of Anytime Sparse Gaussian Process Regression Models with Stochastic Variational Inference for Big Data. In *Proceeding of ICML*, pages 569–578. 3.1.1

T. N. Hoang, Q. M. Hoang, and K. H. Low. 2016. A Distributed Variational Inference Framework for Unifying Parallel Sparse Gaussian Process Regression Models. In *Proceeding of ICML*, pages 382–391. 3.1.1

T. N. Hoang, Q. M. Hoang, K. H. Low, and J. P. How. 2019b. Collective Online Learning of Gaussian Processes in Massive Multi-Agent Systems. In *Proceeding of AAAI*. 3.1.1

T. N. Hoang, Q. M. Hoang, O. Ruofei, and K. H. Low. 2018. Decentralized High-Dimensional Bayesian Optimization with Factor Graphs. In *Proceeding of AAAI*. 3.1.1

T. N. Hoang, K. H. Low, P. Jaillet, and M. Kankanhalli. 2014a. Nonmyopic ϵ-Bayes-Optimal Active Learning of Gaussian Processes. In *Proceeding of ICML*, pages 739–747. 3.1.1

T. N. Hoang, K. H. Low, P. Jaillet, and M. S. Kankanhalli. 2014b. Active Learning Is Planning: Nonmyopic ϵ-Bayes-Optimal Active Learning of Gaussian Processes. In *Proceeding of ECML-PKDD Nectar Track*, pages 494–498. 3.1.1

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780. 2.1.4.5, 2.2.2

Wassily Hoeffding. 1963. Probability Inequalities for the Sum of Bounded Random Variables. *Journal of the American Statistical Association*, 58:13–30. 3.1.3.1

Teakgyu Hong, Donghyun Kim, Mingi Ji, Wonseok Hwang, Daehyun Nam, and Sungrae Park.

2022. BROS: A Pre-trained Language Model Focusing on Text and Layout for Better Key Information Extraction from Documents. *AAAI*. 2.3.1, 2.3.2

Jianying Hu, Sok Gek Lim, and Michael K Brown. 2000. Writer independent on-line handwriting recognition using an HMM approach. *Pattern Recognition*. 2.2.2

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking. *arXiv preprint arXiv:2204.08387*. 2.3.1, 2.3.2

Michael F Hutchinson. 1989. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076. 3.2.1, 3.2.3, 3.2.7

iMotions. 2017. Facial Expression Analysis. 2.1.4.2

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive Mixtures of Local Experts. *Neural Computation*, 3:79–87. 3.3.3.1

Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition. *arXiv preprint arXiv:1406.2227*. 2.2.4.4

T. S. Jayram and David P. Woodruff. 2011. Optimal Bounds for Johnson-Lindenstrauss Transforms and Streaming Problems with Sub-Constant Error. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1–10. 3.2.1, 3.2.3

Shuli Jiang, Hai Pham, David Woodruff, and Richard Zhang. 2021. Optimal Sketching for Trace Estimation. *Advances in Neural Information Processing Systems*, 34. 1.3

William B. Johnson and Joram Lindenstrauss. 1984. Extensions of Lipschitz mappings into Hilbert space. *Contemporary Mathematics*, 26:189–206. 3.2.2

Akshay Kamath, Eric Price, and David P. Woodruff. 2021. A Simple Proof of a New Set Disjointness with Applications to Data Streams. 3.2.1

John Kane and Christer Gobl. 2013. Wavelet Maxima Dispersion for Breathy to Tense Voice Discrimination. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(6):1170–1179. 2.1.4.2

Lei Kang, J Ignacio Toledo, Pau Riba, Mauricio Villegas, Alicia Fornés, and Marçal Rusinol. 2018. Convolve, Attend and Spell: An Attention-based Sequence-to-Sequence Model for Handwritten Word Recognition. In *German Conference on Pattern Recognition*. Springer. 2.2.2, 2.2.4.4, 2.7

Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar,

Shijian Lu, et al. 2015. ICDAR 2015 Competition on Robust Reading. In *ICDAR*, pages 1156–1160. IEEE. 2.7, 2.2.4.4

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR. 2.3.2

Young Jin Kim, Ammar Ahmad Awan, Alexandre Muzio, Andres Felipe Cruz Salinas, Liyang Lu, Amr Hendy, Samyam Rajbhandari, Yuxiong He, and Hany Hassan Awadalla. 2021. Scalable and Efficient MoE Training for Multitask Multilingual Models. *arXiv preprint arXiv:2109.10465*. 3.3.2

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *NIPS*. 2.1.2

Diederik P Kingma and Max Welling. 2013. Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*. 3.1.3.3, 3.1.3.3

Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. 2014. Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models. *arXiv preprint arXiv:1411.2539*. 2.1.2

Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. 2017. Self-Normalizing Neural Networks. In *NIPS*. 2.2.2, 2.9

Mihail N. Kolountzakis, Gary L. Miller, Richard Peng, and Charalampos E. Tsourakakis. 2010. Efficient Triangle Counting in Large Graphs via Degree-Based Vertex Partitioning. *Lecture Notes in Computer Science*, page 15–24. 3.2.7

A. Krause and C. Guestrin. 2007. Nonmyopic Active Learning of Gaussian Processes: An Exploration–Exploitation Approach. In *Proceeding of ICML*, pages 449–456. 3.1.1

Praveen Krishnan, Kartik Dutta, and CV Jawahar. 2016. Deep Feature Embedding for Accurate Recognition and Retrieval of Handwritten Text. In *ICFHR*, pages 289–294. IEEE. 2.2.2

Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning Multiple Layers of Features from Tiny Images. 3.2.1

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-Based & Neural Unsupervised Machine Translation. *CoRR*, abs/1804.07755. 2.1.3.3

B. Laurent and P. Massart. 2000. Adaptive estimation of a quadratic functional by modelselection. *The Annals of Statistics*, 28(5):1302 – 1338. 3.2.1

Angeliki Lazaridou, Nghia The Pham, and Marco Baroni. 2015. Combining Language and Vision with a Multimodal Skip-gram Model. *arXiv preprint arXiv:1501.02598*. 2.1.2

M. Lázaro-Gredilla, J. Quiñonero-Candela, C. E. Rasmussen, and A. R. Figueiras-Vidal. 2010. Sparse Spectrum Gaussian Process Regression. *Journal of Machine Learning Research*, pages 1865–1881. 3.1.1, 3.1.2.2, 3.1.3

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding. *ICLR*. 3.3.1, 3.3.2, 3.3.3.1, 3.3.4.2

Chenliang Li, Bin Bi, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si. 2021. StructuralLM: Structural Pre-training for Form Understanding. *ACL*. 2.3.1, 2.3.2

Chongxuan Li, Kun Xu, Jun Zhu, and Bo Zhang. 2017. Triple Generative Adversarial Nets. *arXiv preprint arXiv:1703.02291*. 2.1.2

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks. In *European Conference on Computer Vision*, pages 121–137. Springer. 2.3.2

Yi Li, Huy L. Nguyen, and David P. Woodruff. 2014. On Sketching Matrix Norms and the Top Singular Vector. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1562–1581. 3.2.1

Yi Li and David P. Woodruff. 2016. On approximating functions of the singular values in a stream. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 726–739. 3.2.1

Paul Pu Liang, Ziyin Liu, AmirAli Bagher Zadeh, and Louis-Philippe Morency. 2018. Multimodal language analysis with recurrent multistage fusion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2.1.2, 2.1.3.1, 2.1.3.2, 2.1.4.2, 2.1.4.3, 2.1.4.5

Lin Lin, Yousef Saad, and Chao Yang. 2013. Approximating Spectral Densities of Large Matrices. *SIAM Review*, 58. 3.2.7

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal Loss for Dense Object Detection. In *ICCV*, pages 2980–2988. 2.2.2

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*. 2.3.4.4

Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, AmirAli Bagher Zadeh, and Louis-Philippe Morency. 2018. Efficient Low-rank Multimodal Fusion With Modality-Specific Factors. In *ACL*. 2.1.2, 2.1.3.2, 2.1.4.5

Marcus Liwicki, Alex Graves, Horst Bunke, and Jürgen Schmidhuber. 2007. A Novel Approach to On-Line Handwriting Recognition Based on Bidirectional Long Short-Term Memory Networks. In *ICDAR*, volume 1, pages 367–371. 2.2.1, 2.2.3.3

K. H. Low, J. Yu, J. Chen, and P. Jaillet. 2015. Parallel Gaussian Process Regression for Big Data: Low-Rank Representation Meets Markov Approximation. In *Proceeding of AAAI*, pages 2821–2827. 3.1.1

Canjie Luo, Lianwen Jin, and Zenghui Sun. 2019. MORAN: A Multi-Object Rectified Attention Network for Scene Text Recognition. *Pattern Recognition*, 90:109–118. 2.2.4.4, 2.7, 2.11, 2.10

Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Xilin Chen, and Ming Zhou. 2020. UniVL: A Unified Video and Language Pre-Training Model for Multimodal Understanding and Generation. *arXiv preprint arXiv:2002.06353*. 2.3.2

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. *arXiv preprint arXiv:1508.04025*. 2.2.3.3

Xuezhe Ma, Xiang Kong, Sinong Wang, Chunting Zhou, Jonathan May, Hao Ma, and Luke Zettlemoyer. 2021. Luna: Linear Unified Nested Attention. *Advances in Neural Information Processing Systems*, 34. 2.3.2

C. Mario Christoudias, Raquel Urtasun, Mathieu Salzmann, and Trevor Darrell. 2010. Learning to Recognize Objects from Unseen Modalities. In *ECCV*. 2.1.2

U-V Marti and Horst Bunke. 2002. The IAM-database: an English sentence database for offline handwriting recognition. *IJDAR*. 2.2.1, 2.7

Emile Mathieu, Tom Rainforth, Siddharth Narayanaswamy, and Yee Whye Teh. 2019. Disentangling Disentanglement in Variational Autoencoders. In *ICML*. 3.1.3.3

Raphael A. Meyer, Cameron Musco, Christopher Musco, and David P. Woodruff. 2020. Hutch++: Optimal Stochastic Trace Estimation. 3.2.1, 3.2.1, 3.2.2, 3.2.3, 3.2.3, 3.2.5, 3.2.5.1, 3.2.10, 3.2.7, 3.2.7

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. *Foundations of Machine Learning*. MIT press. 3.1.2.2, 3.1.3, 3, 3.1.3.1

Louis-Philippe Morency, Rada Mihalcea, and Payal Doshi. 2011. Towards multimodal sentiment analysis: Harvesting opinions from the web. In *ICMI*. ACM. 2.1.1, 2.1.4.1, 2.1.4.5

Louis-Philippe Morency, Ariadna Quattoni, and Trevor Darrell. 2007. Latent-Dynamic Discriminative Models for Continuous Gesture Recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE. 2.1.4.5

C. Musco and C. Musco. 2016. Recursive Sampling for the Nystrom Method. In *Proceeding of NIPS*. 3.1.2.2, 3.1.3.1, 3.1.3.2, 6

S. Muthukrishnan. 2005. Data Streams: Algorithms and Applications. *Found. Trends Theor. Comput. Sci.*, 1(2). 3.2.1

Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. 2011. Multimodal deep learning. In *ICML*. 2.1.2

Laura Nguyen, Thomas Scialom, Jacopo Staiano, and Benjamin Piwowarski. 2021. Skim-Attention: Learning to Focus via Document Layout. *arXiv preprint arXiv:2109.01078*. 2.3.1, 2.3.2

Behnaz Nojavanasghari, Deepak Gopinath, Jayanth Koushik, Tadas Baltrušaitis, and Louis-Philippe Morency. 2016. Deep multimodal fusion for persuasiveness prediction. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, ICMI 2016. 2.1.4.5

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. FAIRSEQ: A Fast, Extensible Toolkit for Sequence Modeling. In *North American Chapter of the Association for Computational Linguistics*. 3.3.1

Gaurav Pandey and Ambedkar Dukkipati. 2017. Variational methods for conditional multimodal deep learning. In *IJCNN*. IEEE. 2.1.2

Bo Pang and Lillian Lee. 2008. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135. 2.1.2

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs Up?: Sentiment Classification Using Machine Learning Techniques. EMNLP. 2.1.1, 2.1.2

Sunghyun Park, Han Suk Shim, Moitreya Chatterjee, Kenji Sagae, and Louis-Philippe Morency. 2014. Computational Analysis of Persuasiveness in Social Multimedia: A Novel Dataset and Multimodal Prediction Approach. In *ICMI '14*. 2.1.4.5

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. 2018. Image Transformer. In *International Conference on Machine Learning*, pages 4055–4064. PMLR. 2.3.2

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. 3.1.4

A. Pavan, Kanat Tangwongsan, Srikanta Tirthapura, and Kun-Lung Wu. 2013. Counting and sampling triangles from a graph stream. *Proceeding of VLDB Endow.*, 6(14):1870–1881. 3.2.7

Barak A. Pearlmutter. 1994. Fast Exact Multiplication by the Hessian. *Neural Computation*, 6:147–160. 3.2.1

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A. Smith, and Lingpeng Kong. 2021. Random Feature Attention. *ArXiv*, abs/2103.02143. 2.3.2

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*. 2.1.4.2

Jonas Pfeiffer, Naman Goyal, Xi Victoria Lin, Xian Li, James Cross, Sebastian Riedel, and Mikel Artetxe. 2022. Lifting the Curse of Multilinguality by Pre-training Modular Transformers. In *North American Chapter of the Association for Computational Linguistics*. 3.3.1

Hai Pham, Paul Pu Liang, Thomas Manzini, Louis-Philippe Morency, and Barnabás Póczos. 2019. Found in translation: Learning robust joint representations by cyclic translations between modalities. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages

6892–6899. 1.3

Hai Pham, Thomas Manzini, Paul Pu Liang, and Barnabas Poczos. 2018. Seq2Seq2Sentiment: Multimodal Sequence to Sequence Models for Sentiment Analysis. In *Challenge-HML*. ACL. 2.1.2

Hai Pham, Amrith Setlur, Saket Dingliwal, Kang Huang, Tzu Hsiang Lin, Zhuo Li, Jae Lim, Collin McCormack, Tam Vu, David Johnson, Nguyen Lam, and Barnabás Póczos. 2020. Robust Handwriting Recognition with Limited Dataset. *International Conference on Frontiers in Handwriting Recognition*. 1.3

Réjean Plamondon and Sargur Srihari. 2000. Online and Offline Handwriting Recognition: A Comprehensive Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2.2.1, 2.2.2

Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2015. Deep Convolutional Neural Network Textual Features and Multiple Kernel Learning for Utterance-Level Multimodal Sentiment Analysis. In *EMNLP*. 2.1.4.5

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, Navonil Majumder, Amir Zadeh, and Louis-Philippe Morency. 2017. Context-Dependent Sentiment Analysis in User-Generated Videos. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. 2.1.4.5

Arik Poznanski and Lior Wolf. 2016. CNN-N-Gram for HandwritingWord Recognition. In *CVPR*, pages 2305–2314. 2.2.2

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W. Black. 2018. Style Transfer Through Back-Translation. In *ACL*. 2.1.3.3

Joan Puigcerver. 2017. Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition? In *ICDAR*. 2.2.2

Ariadna Quattoni, Sybor Wang, Louis-Philippe Morency, Michael Collins, and Trevor Darrell. 2007. Hidden Conditional Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1848–1852. 2.1.4.5

J. Quiñonero-Candela and C. E. Rasmussen. 2005. A Unifying View of Sparse Approximate Gaussian Process Regression. *Journal of Machine Learning Research*, 6:1939–1959. 3.1.1

J. Quiñonero-Candela, C. E. Rasmussen, and C. K. I. Williams. 2007. Approximation Methods for Gaussian Process Regression. *Large-Scale Kernel Machines*, pages 203–223. 3.1.1

A. Rahimi and B. Recht. 2007. Random Features for Large-Scale Kernel Machines. In *Proceeding of NIPS*. 3.1.2.2, 3.1.3, 3.1.3.1

Shyam Sundar Rajagopalan, Louis-Philippe Morency, Tadas Baltrušaitis, and Goecke Roland. 2016. Extending Long Short-Term Memory for Multi-View Structured Learning. In *European Conference on Computer Vision*. 2.1.4.5

Cyrus Rashtchian, David P. Woodruff, and Hanlin Zhu. 2020. Vector-Matrix-Vector Queries for Solving Linear Algebra, Statistics, and Graph Problems. In *APPROX-RANDOM*. 3.2.1

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. DeepSpeed: System Optimizations Enable Training Deep Learning Models with Over 100 Billion Parameters. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3.3.1

C. E. Rasmussen and C. K. I. Williams. 2006. *Gaussian Processes for Machine Learning*. MIT Press. 3.1.1, 3.1.2.1

Joseph Redmon and Ali Farhadi. 2018. YOLOv3: An Incremental Improvement. *arXiv*. 2.2.2, 2.2.3.2, 2.9

Stephen Roller, Sainbayar Sukhbaatar, Jason Weston, et al. 2021. Hash layers for large sparse models. *Advances in Neural Information Processing Systems*, 34. 3.3.2

Farbod Roosta-Khorasani and Uri Ascher. 2015. Improved Bounds on Sample Size for Implicit Matrix Trace Estimators. *Found. Comput. Math.*, 15(5):1187–1212. 3.2.1, 3.2.2

Mark Rudelson and Roman Vershynin. 2010. Non-asymptotic theory of random matrices: extreme singular values. In *Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures*, pages 1576–1602. World Scientific. 3.2.4

M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *Trans. Sig. Proceeding of*, 45(11):2673–2681. 2.1.4.5

Matthias W. Seeger, Christopher K. I. Williams, and Neil D. Lawrence. 2003. Fast Forward Selection to Speed Up Sparse Gaussian Process Regression. In *International Conference on Artificial Intelligence and Statistics*. 3.1.1

Irene Rogan Shaffer. 2018. Exploring the Performance of Facial Expression Recognition Technologies on Deaf Adults and Their Children. In *SIGACCESS Conference on Computers and Accessibility*. 2.1.1

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. *arXiv preprint arXiv:1701.06538*. 3.3.2, 3.3.3.1

Max Simchowitz, Ahmed El Alaoui, and Benjamin Recht. 2018. Tight Query Complexity Lower Bounds for PCA via Finite Sample Deformed Wigner Law. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1249–1259. 3.2.12

K. Simonyan and A. Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1556. 2.2.3.3

Ray Smith. 2007. An Overview of the Tesseract OCR Engine. In *ICDAR*. IEEE. 2.2.1

J. Snoek, L. Hugo, and R. P. Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Proceeding of NIPS*, pages 2960–2968. 3.1.1

Aleksandros Sobczyk and Mathieu Luisier. 2022. Approximate Euclidean lengths and distances beyond Johnson-Lindenstrauss. *ArXiv*, abs/2205.12307. 3.2.2

Joram Soch and Carsten Allefeld. 2016. Kullback-Leibler Divergence for the Normal-Gamma Distribution. 3.2.5

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *EMNLP*. 2.1.2

Kihyuk Sohn, Wenling Shang, and Honglak Lee. 2014. Improved Multimodal Deep Learning with Variation of Information. In *NIPS*. 2.1.2

Yale Song, Louis-Philippe Morency, and Randall Davis. 2012. Multi-View Latent Variable Discriminative Models For Action Recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2120–2127. IEEE. 2.1.4.5

Yale Song, Louis-Philippe Morency, and Randall Davis. 2013. Action Recognition by Hierarchical Sequence Summarization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3562–3569. 2.1.4.5

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. 2023. Consistency Models. *arXiv preprint arXiv:2303.01469*. 1

Yang Song and Stefano Ermon. 2019. Generative Modeling by Estimating Gradients of the Data Distribution. *ArXiv*, abs/1907.05600. 1

N. Srinivas, A. Krause, S. Kakade, and M. Seeger. 2010. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In *Proceeding of ICML*, pages 1015–1022. 3.1.1

Bharath K. Sriperumbudur and Zoltán Szabó. 2015. Optimal Rates for Random Fourier Features. In *NIPS*. 3.1.2.2

Nitish Srivastava and Ruslan Salakhutdinov. 2014. Multimodal Learning with Deep Boltzmann Machines. *JMLR*, 15. 2.1.2

Tomasz Stanisławek, Filip Graliński, Anna Wróblewska, Dawid Lipiński, Agnieszka Kaliska, Paulina Rosalska, Bartosz Topolski, and Przemysław Biecek. 2021. Kleister: Key Information Extraction Datasets Involving Long Documents with Complex Layouts. *arXiv preprint arXiv:2105.05796*. 2.3.4.1

Xiaoming Sun, David P. Woodruff, Guang Yang, and Jialin Zhang. 2021. Querying a Matrix through Matrix-Vector Products. *ACM Trans. Algorithms*, 17(4). 3.2.1

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural

Networks. In *Advances in neural information processing systems*, pages 3104–3112. 2.1.1

Michalis Titsias. 2009. Variational Learning of Inducing Variables in Sparse Gaussian Processes. In *Proceeding of AISTATS*. 3.1.1

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR. 2.3.4.3

Luan Tran, Xiaoming Liu, Jiayu Zhou, and Rong Jin. 2017. Missing Modalities Imputation via Cascaded Residual Autoencoder. In *CVPR*. 2.1.2

Yao-Hung Hubert Tsai, Paul Pu Liang, Amir Zadeh, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2018. Learning Factorized Multimodal Representations. *arXiv preprint arXiv:1806.06176*. 2.1.2, 2.1.3.2

Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2016. Neural Machine Translation with Reconstruction. *CoRR*, abs/1611.01874. 2.1.1

Shashanka Ubaru and Yousef Saad. 2018. Applications of Trace Estimation Techniques. In *High Performance Computing in Science and Engineering*, pages 19–33, Cham. Springer International Publishing. 3.2.1

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605. 2.1.5.2, 3.3

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *arXiv preprint arXiv:1706.03762*. 2.3.1, 2.3.3.3, 3.3.1, 3.3.1, 3.3.3, 3.3.4.2

Paul Voigtlaender, Patrick Doetsch, and Hermann Ney. 2016. Handwriting Recognition with Large Multidimensional Long Short-Term Memory Recurrent Neural Networks. In *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 228–233. IEEE. 2.2.3.3

Paul Voigtlaender, Patrick Doetsch, Simon Wiesler, Ralf Schlüter, and Hermann Ney. 2015. Sequence-discriminative training of recurrent neural networks. In *ICASSP*. IEEE. 2.2.2

Jonas Wacker. 2022. *Random features for dot product kernels and beyond*. Ph.D. thesis, Sorbonne Université. 2.3.2

Haohan Wang, Aaksha Meghawat, Louis-Philippe Morency, and Eric P Xing. 2016. Select-Additive Learning: Improving Cross-individual Generalization in Multimodal Sentiment Analysis. *arXiv preprint arXiv:1609.05244*. 2.1.4.5

Jiapeng Wang, Lianwen Jin, and Kai Ding. 2022. LiLT: A Simple yet Effective Language-Independent Layout Transformer for Structured Document Understanding. *ACL*. 2.3.1, 2.3.2

Sam Waugh. Abalone Dataset. 3.1.4, 3.1

Martin Wöllmer, Felix Weninger, Tobias Knaup, Björn Schuller, Congkai Sun, Kenji Sagae, and Louis-Philippe Morency. 2013. YouTube Movie Reviews: Sentiment Analysis in an Audio-Visual Context. *IEEE Intelligent Systems*. 2.1.4.1

David P Woodruff. 2014. Sketching as a Tool for Numerical Linear Algebra. *arXiv preprint arXiv:1411.4357*. 3.2.3, 3.2.6

Yihong Wu. 2020. Lecture Notes on: Information-Theoretic Methods for High-Dimensional Statistics. 3.2.6

Deyi Xiong et al. SCoMoE: Efficient Mixtures of Experts with Structured Communication. In *The Eleventh International Conference on Learning Representations*. 3.3.2

N. Xu, K. H. Low, J. Chen, K. K. Lim, and E. B. Özgül. 2014. GP-Localize: Persistent Mobile Robot Localization using Online Sparse Gaussian Process Observation Model. In *Proceeding of AAAI*, pages 2585–2592. 3.1.1

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. 2020a. LayoutLMv2: Multi-modal Pre-training for Visually-Rich Document Understanding. *arXiv preprint arXiv:2012.14740*. 2.1.1, 2.3.2, 2.3.3.1, 2.3.4.3

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020b. LayoutLM: Pre-training of Text and Layout for Document Image Understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1192–1200. 2.1.1, 2.3.1, 2.3.1, 2.3.2, 2.3.3.1, 2.3.4.2, 2.3.4.4

Yiheng Xu, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, and Furu Wei. 2021. LayoutXLM: Multimodal Pre-training for Multilingual Visually-rich Document Understanding. *arXiv preprint arXiv:2104.08836*. 2.3.1

Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael Mahoney. 2020. PyHessian: Neural Networks Through the Lens of the Hessian. 3.2.1

Jiahong Yuan and Mark Liberman. 2008. Speaker identification on the SCOTUS corpus. *Journal of the Acoustical Society of America*. 2.1.4.3

Seniha Esen Yüksel, Joseph N. Wilson, and Paul D. Gader. 2012. Twenty Years of Mixture of Experts. *IEEE Transactions on Neural Networks and Learning Systems*, 23:1177–1193. 3.3.1

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor Fusion Network for Multimodal Sentiment Analysis. In *EMNLP*, pages 1114–1125. 2.1.2, 2.1.4.5, 2.1.5.2

Amir Zadeh, Paul Pu Liang, Navonil Mazumder, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018. Memory Fusion Network for Multi-view Sequential Learning. *arXiv preprint arXiv:1802.00927*. 2.1.2, 2.1.3.2, 2.1.4.5

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago On-

tanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big Bird: Transformers for Longer Sequences. *arXiv preprint arXiv:2007.14062*. 2.3.2, 2.3.3.3, 2.3.4.2, 2.3.4.4

Y. Zhang, T. N. Hoang, K. H. Low, and M. Kankanhalli. 2016. Near-Optimal Active Learning of Multi-Output Gaussian Processes. In *Proceeding of AAAI*, pages 2351–2357. 3.1.1

Y. Zhang, T. N. Hoang, K. H. Low, and M. Kankanhalli. 2017. Information-Based Multi-Fidelity Bayesian Optimization. In *NIPS Workshop BayesOpt*. 3.1.1

Yuchen Zhang, Martin Wainwright, and Michael Jordan. 2015. Distributed estimation of generalized matrix rank: Efficient algorithms and lower bounds. In *International Conference on Machine Learning*, pages 457–465. PMLR. 3.2.1

Lianmin Zheng, Zhuohan Li, Hao Zhang, Yonghao Zhuang, Zhifeng Chen, Yanping Huang, Yida Wang, Yuanzhong Xu, Danyang Zhuo, Joseph E Gonzalez, et al. 2022. Alpa: Automating Inter- and Intra-Operator Parallelism for Distributed Deep Learning. *OSDI*. 3.3.2

Liu Zhili, Kai Chen, Jianhua Han, Hong Lanqing, Hang Xu, Zhenguo Li, and James Kwok. Task-customized Masked Autoencoder via Mixture of Cluster-conditional Experts. In *The Eleventh International Conference on Learning Representations*. 3.3.2

Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. 2017. EAST: an efficient and accurate scene text detector. In *CVPR*, pages 5551–5560. 2.2.4.4

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew Dai, Zhifeng Chen, Quoc Le, and James Laudon. 2022. Mixture-of-Experts with Expert Choice Routing. *arXiv preprint arXiv:2202.09368*. 3.3.2

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2017. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *CoRR*, abs/1703.10593. 2.1.3.3

Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng, and Shai Avidan. 2006. Fast Human Detection Using a Cascade of Histograms of Oriented Gradients. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1491–1498. IEEE. 2.1.4.2

Matthias Zimmermann and Horst Bunke. 2002. Automatic segmentation of the IAM off-line database for handwritten English text. In *Pattern Recognition*, volume 4, pages 35–39. IEEE. 2.2.2

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. ST-MoE: Designing Stable and Transferable Sparse Expert Models. *arXiv preprint arXiv:2202.08906*. 3.3.2