Distribution Agreement

Signature:

______________________________________ ____________

Rongmei Lin Date

Exploring Invariance in Single and Multi-modal Deep Representation Learning

By

Rongmei Lin
Doctor of Philosophy

Computer Science and Informatics

Li Xiong, Ph.D.
Advisor

Lucila Ohno-Machado, Ph.D.
Committee Member

Joyce Ho, Ph.D.
Committee Member

Liang Zhao, Ph.D.
Committee Member

Accepted:

Kimberly Jacob Arriola, Ph.D.
Dean of the James T. Laney School of Graduate Studies

Date

Exploring Invariance in Single and Multi-modal Deep Representation Learning

By

Rongmei Lin
B.Eng., South China University of Technology, 2013
M.Sc., Emory University, 2017

Advisor: Li Xiong, Ph.D.

An abstract of
A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2022

Abstract

Exploring Invariance in Single and Multi-modal Deep Representation Learning
By Rongmei Lin

The recent successes in artificial intelligence have been largely attributed to the powerful and rich representation from deep neural networks. General-purpose representation learning has been well studied in the past decade. The ultimate goal of representation learning is to achieve a certain level of invariance. For example, for generic image recognition, we aim to learn features that are only sensitive to image labels and invariant to the intra-class variations such as backgrounds, object poses. We propose the single-modal / multi-modal generalization to handle different scenarios. Single-modal generalization is the classic setting of supervised learning where the training and testing data are drawn from the same distribution. Most deep neural network architectures and generalization techniques are designed towards this end. In contrast, multi-modal generalization considers the problem where the data are drawn from different modalities such as image/text or multi-sensor healthcare data. These variant modalities are differently represented yet complement each other simultaneously. It is worth considering the interactions between modalities rather than simply concatenating the information. My thesis focuses on the topic of learning task-driven invariant representations and the contributions can be summarized as follows: 1) We introduce an unified regularizer for invariant representation learning by promoting the angular diversity of neurons; 2) We propose a framework to fuse multimodal data homogeneously and learn features that are invariant to specific modality; 3) We further extend the multimodal framework with pre-training tasks on extensive vision-language and healthcare tasks, which leads to significant performance improvement.

Exploring Invariance in Single and Multi-modal Deep Representation Learning

By

Rongmei Lin
B.Eng., South China University of Technology, 2013
M.Sc., Emory University, 2017

Advisor: Li Xiong, Ph.D.

A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2022

Acknowledgments

Throughout the past four years of research and the writing of this dissertation, I have received a great deal of support and assistance. First of all, I would like to express my sincere gratitude to my advisor Prof. Li Xiong, who has supported me throughout these years of study. Your invaluable expertise has pushed me to think out of the box and brought my research to a higher level. I am also extremely grateful for your person support and understanding during the pandemic. You will always be my role model for a researcher, teacher and mentor. Further, I would like to thank the group members of AIMS group at Emory. They provided a friendly and cooperative atmosphere at work and also insightful feedback and comments. Last but not least, I especially thank my parent and friends for the unconditional love and care in these very intense academic years. Special thanks to my mother and father, they have scarified a lot for me during my childhood. My mother is more like my closest friend, who is always being there whenever I needed a listener and I am forever grateful. I cannot forget to thank Weiyang, this dissertation would not have been possible without your support and encouragement.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Background

The recent successes in artificial intelligence [39, 99, 116, 126, 127] have been largely attributed to the powerful and rich representation from deep neural networks. General-purpose representation learning [67, 39, 117, 40, 16] has been well studied in the past decade. Neural networks are a powerful class of nonlinear functions that can be trained end-to-end on various applications. While the over-parametrization nature in many neural networks renders the ability to fit complex functions and the strong representation power to handle challenging tasks, it also leads to highly correlated neurons that can hurt the generalization ability. Such high capacity model usually performs well on training set but poorly on the held-out validation set. We observe that deep models are susceptible to overfitting. This issue is further amplified when corrupted data and noisy labels exist in the training set. The dense model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as patterns by the model. As a result, how to avoid undesired representation learning becomes an important issue.

This phenomenon motivated us to encourage certain level of *invariance* in the deep representation learning. The model should learn diverse and non-redundant representations that are invariant to other distracting factors. For example, for generic image recognition, we aim to learn features that are only sensitive to image labels and invariant to the noise and the intra-class variations (e.g. background, pose, direction). Such unbiased representation also becomes increasingly important nowadays due to the fact that many existing data-driven machine learning models are biased towards certain group of people. Fair representation should be invariant to some protected attributes such as gender, race, etc.

On the other hand, the world naturally provides us with data of multiple modalities. Each of the modality presents a different view of the same instance. In addition to the distribution discrepancy between the training data and test data in single-modal setting, the multi-modal deep representation learning faces many unique challenges. The multi-source data is collected from diverse perspective and presents heterogeneous properties. Although there are many intrinsic associations among modalities, it can still be difficult to directly relate raw pixels from image data to wave forms from the speech data. The techniques used in single-modal deep representation learning cannot easily be transferred to multi-modal setting. In order to pursue the *invariance* in the multi-modal settings, we need to further explore the exclusive limitations and challenges in the multi-modal domain and propose specific solutions.

To this end, our research centers around the topic of invariant deep representation learning both in single and multi-modal settings. Under this unified umbrella, we consider various applications. The goal in common is to maximize the intra-class mutual agreement on heterogeneous views of data and improve the quality of latent representation.

Unified Framework. We first introduce a simple unified framework for invariant representation learning. We denote $\boldsymbol{S}$ as the set of intrinsic attributes and $\boldsymbol{P}$ as the set of invariant attributes. Then we define $\mathcal{D}^g_{s,p}$ (g is some feature transformation) as the feature distribution of intrinsic attribute $s \in \boldsymbol{S}$ and invariant attribute $p \in \boldsymbol{P}$. The high-level idea of invariant representation learning is to learn a parameterized feature transformation $g : \mathcal{X} \to \mathcal{Z}$ such that the distribution on the feature space $\mathcal{Z}$ satisfies that $d(\mathcal{D}^g_{s,p_i}, \mathcal{D}^g_{s,p_j}) = 0$ for any $p_i, p_j \in \mathcal{P}$ and any $s \in \mathcal{S}$. d denotes some distance measure of distributions. This unified framework enables us to conduct invariant representation learning in different tasks.

Single-modal Generalization. Single-modal generalization is the classic setting of supervised learning where the training and testing data are drawn from the same distribution. For example, image classification is a typical example of in-domain generalization. The goal here is to learn features that are only sensitive to labels and are invariant to intra-class variation such as background, object pose, etc. Most deep neural network architectures [67, 39, 117] are designed towards this end. In this sub branch, we address the central question: *how to regularize the network to avoid undesired representation redundancy and achieve certain level of invariance?*

Multi-modal Generalization. In contrast to single-modal generalization, multi-modal generalization considers the problem where the data are drawn from different modalities such as image/text or multi-sensor healthcare data. These variant modalities are differently represented yet complement each other simultaneously. It is worth considering the interactions between modalities rather than simply concatenating the information. We study the following question: *How to fuse the multi-modal data and maintain the desired "invariant" property in the single-modal representation learning?* In other words, the goal here is to learn intrinsic features that are invariant to heterogeneous modalities and discriminative in terms of labels.

1.2 Related Works

1.2.1 Generalizations

Diversity regularization. Diversity regularization is shown useful in sparse coding [92, 104], ensemble learning [75, 68], self-paced learning [57], metric learning [140], etc. Early studies in sparse coding [92, 104] show that the generalization ability of codebook can be improved via diversity regularization, where the diversity is often modeled using the (empirical) covariance matrix. More recently, a series of studies have featured diversity regularization in neural networks [137, 139, 138, 22, 109, 136], where regularization is mostly achieved via promoting large angle/orthogonality, or reducing covariance between bases. However, diversity is formulated on local variable, the scale and flexibility is somehow limited. Methods other than diversity-promoting regularization have been widely proposed to improve CNNs [121, 55, 94, 84] and generative adversarial nets (GANs) [12, 95]. Note that the generalization technique can be combined together if formulated properly, new techniques can be added and regarded as a complement that can be applied on top of these methods.

Relational regularizations. There are quite a number of relational regularizations that have been used in neural networks, such as orthogonality regularization[7, 84, 108, 52, 18], unitary constraint [58, 134, 5], decorrelation [108, 21, 139], spectral regularization [144], low-rank regularization [123], angular constraint [138, 72], etc. Most of these relational regularizations are either directly based on orthogonality or based on some notions related to orthogonality (*e.g.*, correlation). Such methods fall in the category of enforcing "hard orthogonality constraints" into optimization, and have to repeat singular value decomposition (SVD) during training, which is a computational expensive and time consuming operation.

Model Compression. The over-parametrization property in deep neural network motivated a series of influential works in network compression [36, 1] and parameter-efficient network architectures [48, 54, 147]. These works either compress the network by pruning redundant neurons or directly modify the network architecture, aiming to achieve comparable performance while using fewer parameters. Yet, it remains an open problem to find a unified and principled theory that guides the network compression in the context of optimal generalization ability.

1.2.2 Multi-modal Learning

Multi-modal Learning. Information surrounding us usually involves multiple modalities, where we consider environment that are observed using multiple sensors and each sensor output can be termed as *modality* associated with a single data set. The underlying motivation to use multimodal data is that complementary information could be extracted from each of the modalities considered for a given learning task, yielding a richer representation that could be used to produce much improved performance compared to using only a single modality. Recently, multimodal application has been explored with an emphasis on language and vision tasks. One of the most representative applications is image captioning where the task is to generate a text description of the input image [45]. This is motivated by the ability of such systems to help the visually impaired in their daily tasks [9]. The main challenges media description is evaluation: how to evaluate the quality of the predicted descriptions. The task of visual question-answering (VQA) was recently proposed to address some of the evaluation challenges [4], where the goal is to answer a specific question about the image. Early works in VQA focus on the design of the attention mechanism to merge information from image and text modality, such as the bilinear attention in [65]. The importance of words in the image to the VQA task was first recognized in [119] which proposed a new benchmark TextVQA dataset. Hu, et. al. [49] proposed to use transformer [126] to express a more general form of attention between image, image objects, image texts and questions. Recently [142] introduced pre-training tasks to this model architecture that boosted the state of the art of TextVQA benchmark significantly.

Multi-modal Fusion. Most approaches on multimodal fusion for medical signals can be classified into three categories: *(1) early fusion.* Signals from different modalities are pre-processed and concatenated in the early phrase. Features are extracted from such combined signals and feed into the downstream task like classification. Early fusion methods [131, 96, 97] require innovations in sensor synchronization, buffering, denoising and data normalization. *(2) late fusion.* Raw signals from each sensors are featurized separately and then fused for downstream task. Such fusion method requires feature selection [74] and feature normalization [2] to handle different time spans and signal scales. In addition, these separate features can be fused in different ways, such as naive concatenating before classifier, adding extra classifier for each modality and applying majority voting [111]. *(3) gated fusion.* LSTM [44] and GRU [17] have been widely used to process temporal multimodal data and extract the underlying patterns. A series of of attention-based fusion methods are proposed to model complex temporal correlations by adding the gated fusion. [145] considers each modality in isolation to learn view-specific interactions. It then uses an explicitly designed gated mechanism to find and store cross-view interactions over time. [47, 98] modified the LSTM cell and apply the gated attention to fuse multi-modal features. The first and the second fusion category is insufficient in terms of measuring the inter-modality correlations. The third category lacks flexibility while handling the gated fusion, only extracted or well aligned features are able to attend to the attention mechanism.

1.3 Research Contributions

Representation learning is the key components to ingest and process the original raw data in machine learning. The performance of subsequent learning task heavily relies on the quality of representation. Most representation learning problem face a trade-off between preserving as much information from the training data as possible and maintaining nice properties, such as *Independence* and the *Invariance* we highlighted in the thesis. This thesis explore different techniques and propose several solutions to handle the trade-off and learn the invariant representation in both single-modal and multi-modal scenarios. Our contributions can be grouped into two categories:

Single-modal Generalization. In the case of single-modal representation learning. We focus on how to regularize the network to avoid undesired representation. We have observed redundant and highly correlated neurons caused by the over-parametrization in deep neural networks. To reduce the redundancy and improve the generalization ability of neural networks, inspired by the Thomson problem in physics where the distribution of multiple propelling electrons on a unit sphere can be modeled via minimizing some potential energy, we propose a novel minimum hyperspherical energy (MHE) regularization framework, where the diversity of neurons is promoted by minimizing the hyperspherical energy in each layer. As verified by comprehensive experiments on multiple tasks, MHE is able to consistently improve the generalization power of neural networks. This line of work has been published in NeurIPS 2018, CVPR 2020, AISTATS 2021 and CVPR 2021. The detailed contributions are summarized as follows:

1. We propose MHE defined on Euclidean distance, as indicated in physics Thomson problem. We also consider minimizing hyperspherical energy defined with respect to angular distance. In addition, we provide theoretical insights of MHE regularization.

2. To address the drawbacks of MHE in high dimensional space, we propose compressive minimum hyperspherical energy (CoMHE) as a dynamic regularization to effectively minimize hyperspherical energy of neurons for better generalizability.

3. We design a novel over-parameterized training (OPT) framework with strong flexibility. OPT is the first training framework where the hyperspherical energy is provably minimized, leading to better empirical generalization. OPT reveals that learning a proper coordinate system is crucial to generalization, and the hyperspherical energy is sufficiently expressive to characterize relative neuron positions.

Multi-modal Generalization. In real world applications, data can be acquired from single or multiple modalities. In order to handle the multi-modal scenarios, we further explore the invariant representation learning to achieve high utility performance. In this branch, we focus on the representation learning that fused information from heterogeneous modalities. We propose a framework to fuse multimodal data homogeneously and learn features that are invariant to specific modality. We further extend the multimodal framework with pre-training tasks on extensive vision-language and healthcare tasks, which leads to significant performance improvement. Parts of this work has been published in KDD 2021 and we plan to summarize our findings and submit to NeurIPS 2022. The contributions are summarized as follows:

1. We propose a transformer-based sequence-to-sequence model to extract attribute values jointly from textual profile, visual information, and texts in images. To the best of our knowledge, this is the *first* work for multi-modal attribute value extraction.

2. We extend our basic solution to a cross-domain extraction model by equipping the model with an external dynamic vocabulary conditioned on domain priors

and multi-task training incorporated with our sequence-to sequence model.

3. We conduct extensive experiments to evaluate our solution on a dataset collected from a public e-commerce website across multiple product categories. Our approach consistently outperforms state-of-the-art solutions by 15% on recall and 10% on F1 metric.

4. On the healthcare applications, we propose the Multi-Sensor Fusion Framework along with Pre-training tasks designed for clinical timeseries data. Specifically, we introduce the attention-based halfway fusion mechanism utilizing the transformer layers, which enables modeling both inter- and intra- modality relations in a homogeneous way. Our framework along with the pre-training task improves the macro AUROC by significant margin. In particular, our method outperforms baseline models even if the training data is only 10% of baseline's training data.

Chapter 2

Single-modal Generalization on Hypersphere

2.1 Regularization on Hypersphere

2.1.1 Overview

Current deep networks are able to achieve impressive performance on large-scale problems. A steam of works seeks to further release the network generalization power by alleviating redundancy through diversification [139, 138, 22, 109] as rigorously analyzed by [137]. Most of these works address the redundancy problem by enforcing relatively large diversity between pairwise projection bases via regularization. Our work broadly falls into this category by sharing similar high-level target, but the spirit and motivation behind our proposed models are distinct. In particular, there is a recent trend of studies that feature the significance of angular learning at both loss and convolution levels [79, 82, 84, 87], based on the observation that the angles in deep embeddings learned by CNNs tend to encode semantic difference. The key intuition is that angles preserve the most abundant and discriminative information for visual recognition. As a result, hyperspherical geodesic distances between neurons naturally play a key role in this context, and thus, it is intuitively desired to impose discrimination by keeping their projections on the hypersphere as far away from each other as possible. While the concept of imposing large angular diversities was also considered in [137, 139, 138, 109], they do not consider diversity in terms of global equidistribution of embeddings on the hypersphere, which fails to achieve the state-of-the-art performances.

Given the above motivation, we draw inspiration from a well-known physics problem, called Thomson problem [125, 120]. The goal of Thomson problem is to determine the minimum electrostatic potential energy configuration of N mutually-repelling electrons on the surface of a unit sphere. We identify the intrinsic resemblance between the Thomson problem and our target, in the sense that diversifying neurons can be seen as searching for an optimal configuration of electron locations. Similarly, we characterize the diversity for a group of neurons by defining a generic hyperspherical potential energy using their pairwise relationship. Higher energy implies higher redundancy, while lower energy indicates that these neurons are more diverse and more uniformly spaced. To reduce the redundancy of neurons and improve the neural networks, we propose a novel *minimum hyperspherical energy* (MHE) regularization framework, where the diversity of neurons is promoted by minimizing the hyperspherical energy in each layer. As verified by comprehensive experiments on multiple tasks, MHE is able to consistently improve the generalization power of neural networks.

MHE faces different situations when it is applied to hidden layers and output layers. For hidden layers, applying MHE straightforwardly may still encourage some degree of redundancy since it will produce co-linear bases pointing to opposite directions (see Fig. 2.1 middle). In order to avoid such redundancy, we propose the half-space MHE which constructs a group of virtual neurons and minimize the hyperspherical energy of both existing and virtual neurons. For output layers, MHE aims

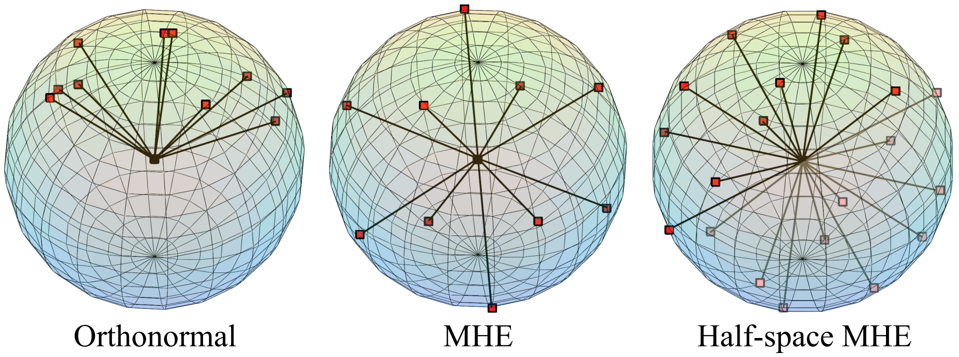


Figure 2.1: Orthonormal, MHE and half-space MHE regularization. The red dots denote the neurons optimized by the gradient of the corresponding regularization. The rightmost pink dots denote the virtual negative neurons. We randomly initialize the weights of 10 neurons on a 3D Sphere and optimize them with SGD.

to distribute the classifier neurons[1] as uniformly as possible to improve the inter-class feature separability. Different from MHE in hidden layers, classifier neurons should be distributed in the full space for the best classification performance [79, 82]. An intuitive comparison among the widely used orthonormal regularization, the proposed MHE and half-space MHE is provided in Fig. 2.1. One can observe that both MHE and half-space MHE are able to uniformly distribute the neurons over the hypersphere and half-space hypersphere, respectively. In contrast, conventional orthonormal regularization tends to group neurons closer, especially when the number of neurons is greater than the dimension.

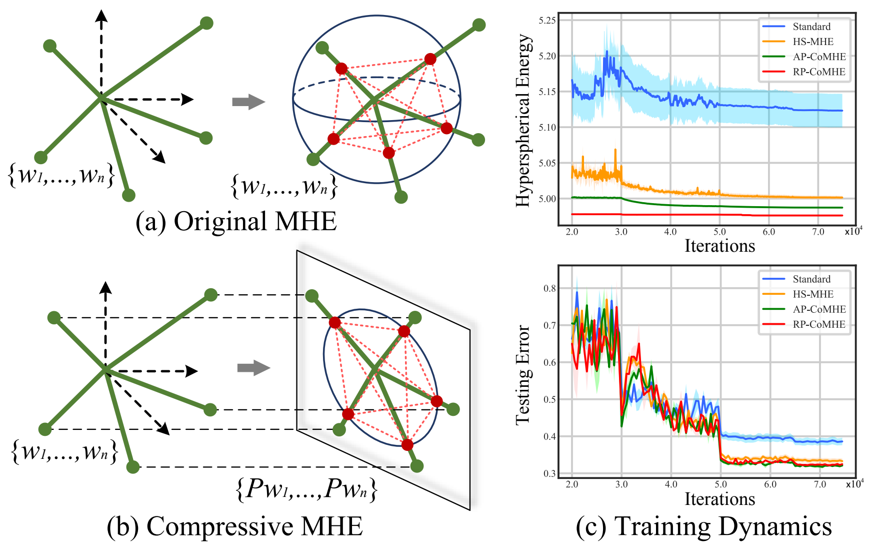
MHE is originally defined on Euclidean distance, as indicated in Thomson problem. However, we further consider minimizing hyperspherical energy defined with respect to angular distance, which we will refer to as angular-MHE (A-MHE) in the following chapters. In addition, we give some theoretical insights of MHE regularization, by discussing the asymptotic behavior and generalization error. Last, we apply MHE regularization to multiple vision tasks, including generic object recognition, class-imbalance learning, and face recognition. In the experiments, we show

[1] Classifier neurons are the projection bases of the last layer (*i.e.*, output layer) before input to softmax.

that MHE is architecture-agnostic and can considerably improve the generalization ability.

Although minimizing hyperspherical energy has already been empirically shown useful in a number of applications [85], two fundamental questions remain unanswered: *(1) what is the role that hyperspherical energy plays in training a well-performing neural network?* and *(2) How can the hyperspherical energy be effectively minimized?* To study the first question, we plot the training dynamics of hyperspherical energy (on CIFAR-100) in Fig. 2.2(c) for a baseline convolutional neural network (CNN) without any MHE variant, a CNN regularized by MHE [85] its variant. From the empirical results in Fig. 2.2(c), we find that MHE can achieve much lower hyperspherical energy and testing error than the baseline, showing the effectiveness of minimizing hyperspherical energy. It also implies that lower hyperspherical energy typically leads to better generalization. We empirically observe that a trained neural network with lower hyperspherical energy often generalizes better (*i.e.*, higher hyperspherical diversity leads to better generalization), and therefore we argue that hyperspherical energy is closely related to the generalization power of neural networks.

By adopting the definition of hyperspherical energy as the regularization objective and naively minimizing it with back-propagation, MHE suffers from a few critical problems which limit it to further unleash its potential. First, the original MHE objective has a huge number of local minima and stationary points due to its highly non-convex and non-linear objective function. The problem can get even worse when the space dimension gets higher and the number of neurons becomes larger [8, 13]. Second, the gradient of the original MHE objective *w.r.t* the neuron weight is deterministic. Unlike the weight decay whose objective is convex, MHE has a complex and non-convex regularization term. Therefore, deterministic gradients may make the solution quickly fall into one of the bad local minima and get stuck there. Third, MHE defines an ill-posed problem in general. When the number of neurons is smaller



(a) Original MHE

(b) Compressive MHE

(c) Training Dynamics

Figure 2.2: Comparison of original MHE and compressive MHE. In (c), the top figure shows the hyperspherical energy, and the bottom one shows the testing error (CIFAR-100).

than the dimension of the space (it is often the case in neural networks), it will be less meaningful to encourage the hyperspherical diversity since the neurons can not fully occupy the space. Last, in high-dimensional spaces, randomly initialized neurons are likely to be orthogonal to each other. Therefore, these high-dimensional neurons can be trivially "diverse", leading to small gradients in original MHE that cause optimization difficulties.

In order to address these problems and effectively minimize hyperspherical energy, we propose the compressive minimum hyperspherical energy (CoMHE) as a generic regularization for neural networks. The high-level intuition behind CoMHE is to project neurons to some suitable subspaces such that the hyperspherical energy can get minimized more effectively. Specifically, CoMHE first maps the neurons from a high-dimensional space to a low-dimensional one and then minimizes the hyperspherical energy of these neurons. Therefore, how to map these neurons to a low-dimensional space while preserving the desirable information in high-dimensional space is our major concern. Since we aim to regularize the directions of neurons, what we care most is the angular similarity between different neurons. To this end, we explore multiple novel methods to perform the projection and heavily study two main approaches: *random projection* and *angle-preserving projection*, which can reduce the dimensionality of neurons while still partially preserving the pairwise angles.

2.1.2 Proposed Method

(a) Standard MHE. MHE characterizes the diversity of N neurons ($\boldsymbol{W}_N=\{\boldsymbol{w}_1,\cdots,\boldsymbol{w}_N\in\mathbb{R}^{d+1}\}$) on a unit hypersphere using hyperspherical energy which is defined as

$$\begin{aligned}\boldsymbol{E}_{s,d}(\hat{\boldsymbol{w}}_i|_{i=1}^N) &= \sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} f_s\big(\,\|\hat{\boldsymbol{w}}_i-\hat{\boldsymbol{w}}_j\|\,\big)\\ &= \begin{cases}\sum_{i\neq j}\|\hat{\boldsymbol{w}}_i-\hat{\boldsymbol{w}}_j\|^{-s}, & s>0\\ \sum_{i\neq j}\log\big(\,\|\hat{\boldsymbol{w}}_i-\hat{\boldsymbol{w}}_j\|^{-1}\big), & s=0\end{cases}\end{aligned} \tag{2.1}$$

where $\|\cdot\|$ denotes ℓ_2 norm, $f_s(\cdot)$ is a decreasing real-valued function (we use $f_s(z)=z^{-s}, s>0$, *i.e.*, Riesz s-kernels), and $\hat{\boldsymbol{w}}_i=\frac{\boldsymbol{w}_i}{\|\boldsymbol{w}_i\|}$ is the i-th neuron weight projected onto the unit hypersphere $\mathbb{S}^d=\{\boldsymbol{v}\in\mathbb{R}^{d+1}|\,\|\boldsymbol{v}\|=1\}$. For convenience, we denote $\hat{\boldsymbol{W}}_N=\{\hat{\boldsymbol{w}}_1,\cdots,\hat{\boldsymbol{w}}_N\in\mathbb{S}^d\}$, and $\boldsymbol{E}_s=\boldsymbol{E}_{s,d}(\hat{\boldsymbol{w}}_i|_{i=1}^N)$. Note that, each neuron is a convolution kernel in CNNs. MHE minimizes the hyperspherical energy of neurons using gradient descent during back-propagation, and MHE is typically applied to the neural network in a layer-wise fashion. We first write down the gradient of $\boldsymbol{E}_2$ *w.r.t* $\hat{\boldsymbol{w}}_i$ and make the gradient to be zero:

$$\nabla_{\hat{\boldsymbol{w}}_i}\boldsymbol{E}_2=\sum_{j=1,j\neq i}^{N}\frac{-2(\hat{\boldsymbol{w}}_i-\hat{\boldsymbol{w}}_j)}{\|\hat{\boldsymbol{w}}_i-\hat{\boldsymbol{w}}_j\|^4}=0\Rightarrow\hat{\boldsymbol{w}}_i=\frac{\sum_{j=1,j\neq i}^{N}\alpha_j\hat{\boldsymbol{w}}_j}{\sum_{j=1,j\neq i}^{N}\alpha_j} \tag{2.2}$$

where $\alpha_j=\|\hat{\boldsymbol{w}}_i-\hat{\boldsymbol{w}}_j\|^{-4}$. We use toy and informal examples to show that high dimensional space (*i.e.*, d is large) leads to much more stationary points than low-dimensional one. Assume there are $K=K_1+K_2$ stationary points in total for $\hat{\boldsymbol{W}}_N$ to satisfy Eq. 2.2, where K_1 denotes the number of stationary points in which every element in the solution is distinct and K_2 denotes the number of the rest stationary points. We give two examples: *(i)* For $(d+2)$-dimensional space, we can extend the solutions in $(d+1)$-dimensional space by introducing a new dimension with zero value. The new solutions satisfy Eq. 2.2. Because there are $d+2$ ways to insert the zero, we have at least $(d+2)K$ stationary points in $(d+2)$-dimensional space. *(ii)* We denote $K_1'=\frac{K_1}{(d+1)!}$ as the number of unordered sets that construct the stationary points. In $(2d+2)$-dimensional space, we can construct $\hat{\boldsymbol{w}}_j^E=\frac{1}{\sqrt{2}}\{\hat{\boldsymbol{w}}_j;\hat{\boldsymbol{w}}_j\}\in\mathbb{S}^{2d+1},\forall j$ that satisfies Eq. 2.2. Therefore, there are at least $\frac{(2d+2)!}{2^{d+1}}K_1'+K_2$ stationary points for $\hat{\boldsymbol{W}}_N$ in $(2d+2)$-dimensional space, and besides this construction, there are much more stationary points. Therefore, MHE have far more stationary points in higher dimensions.

(b) MHE as Regularization for Neural Networks. Now that we have introduced the formulation of MHE, we propose MHE regularization for neural networks. In supervised neural network learning, the entire objective function is shown as follows:

$$\mathcal{L}=\underbrace{\frac{1}{m}\sum_{j=1}^{m}\ell(\langle\boldsymbol{w}_i^{\text{out}},\boldsymbol{x}_j\rangle_{i=1}^{c},\boldsymbol{y}_j)}_{\text{training data fitting}}+\underbrace{\lambda_{\text{h}}\cdot\sum_{j=1}^{L-1}\frac{1}{N_j(N_j-1)}\{\boldsymbol{E}_s\}_j}_{T_{\text{h}}\text{: hyperspherical energy for hidden layers}}+\underbrace{\lambda_{\text{o}}\cdot\frac{1}{N_L(N_L-1)}\boldsymbol{E}_s(\hat{\boldsymbol{w}}_i^{\text{out}}|_{i=1}^{c})}_{T_{\text{o}}\text{: hyperspherical energy for output layer}} \tag{2.3}$$

where $\boldsymbol{x}_i$ is the feature of the i-th training sample entering the output layer, $\boldsymbol{w}_i^{\text{out}}$ is the classifier neuron for the i-th class in the output fully-connected layer and $\hat{\boldsymbol{w}}_i^{\text{out}}$ denotes its normalized version. $\{\boldsymbol{E}_s\}_i$ denotes the hyperspherical energy for the neurons in the i-th layer. c is the number of classes, m is the batch size, L is the number of layers of the neural network, and N_i is the number of neurons in the i-th layer. $\boldsymbol{E}_s(\hat{\boldsymbol{w}}_i^{\text{out}}|_{i=1}^{c})$ denotes the hyperspherical energy of neurons $\{\hat{\boldsymbol{w}}_1^{\text{out}},\cdots,\hat{\boldsymbol{w}}_c^{\text{out}}\}$. The ℓ_2 weight decay is omitted here for simplicity, but we will use it in practice. MHE has different effects and interpretations in regularizing hidden layers and output layers.

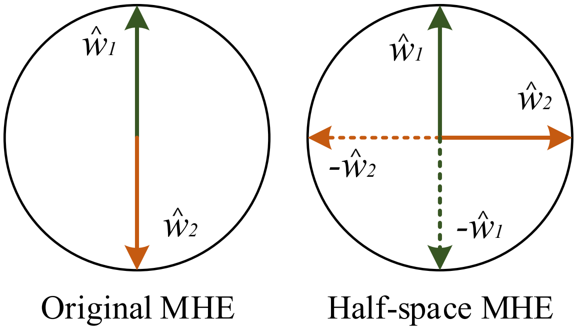


Figure 2.3: Half-space MHE.

MHE for hidden layers. To make neurons in the hidden layers more discriminative and less redundant, we propose to use MHE as a form of regularization. MHE encourages the normalized neurons to be uniformly distributed on a unit hypersphere, which is partially inspired by the observation in [84] that angular difference in neurons preserves semantic (label-related) information. To some extent, MHE maximizes the average angular difference between neurons (specifically, the hyperspherical energy of neurons in every hidden layer). For instance, in CNNs we minimize the hyperspherical energy of kernels in convolutional and fully-connected layers except the output layer.

MHE for output layers. For the output layer, we propose to enhance the inter-class feature separability with MHE to learn discriminative and well-separated features. For classification tasks, MHE regularization is complementary to the softmax cross-entropy loss in CNNs. The softmax loss focuses more on the intra-class compactness, while MHE encourages the inter-class separability. Therefore, MHE on output layers can induce features with better generalization power.

MHE in half space. Directly applying the MHE formulation may still encounter some redundancy. An example in Fig. 2.3, with two neurons in a 2-dimensional space, illustrates this potential issue. Directly imposing the original MHE regularization leads to a solution that two neurons are collinear but with opposite directions. To avoid such redundancy, we propose the half-space MHE regularization which constructs some virtual neurons and minimizes the hyperspherical energy of both original and virtual neurons together. Specifically, half-space MHE constructs a collinear virtual neuron with opposite direction for every existing neuron. Therefore, we end up with minimizing the hyperspherical energy with $2N_i$ neurons in the i-th layer (*i.e.*, minimizing $\boldsymbol{E}_s(\{\hat{\boldsymbol{w}}_k, -\hat{\boldsymbol{w}}_k\}|_{k=1}^{2N_i})$). This half-space variant will encourage the neurons to be less correlated and less redundant, as illustrated in Fig. 2.3. Note that, half-space MHE can only be used in hidden layers, because the collinear neurons do not constitute redundancy in output layers, as shown in [79]. Nevertheless, collinearity is usually not likely to happen in high-dimensional spaces, especially when the neurons are optimized to fit training data. This may be the reason that the original MHE regularization still consistently improves the baselines.

(c) General Framework of CoMHE. To overcome MHE's drawbacks in high dimensional space, we propose the compressive MHE that projects the neurons to a low-dimensional space and then minimizes the hyperspherical energy of the projected neurons. In general, CoMHE minimizes the following form of energy:

$$\boldsymbol{E}_s^C(\hat{\boldsymbol{W}}_N) := \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} f_s\big(\|g(\hat{\boldsymbol{w}}_i) - g(\hat{\boldsymbol{w}}_j)\| \big) \tag{2.4}$$

where $g:\mathbb{S}^d \rightarrow \mathbb{S}^k$ takes a normalized $(d+1)$-dimensional input and outputs a normalized $(k+1)$-dimensional vector. $g(\cdot)$ can be either linear or nonlinear mapping. We only consider the linear case here. Using multi-layer perceptrons as $g(\cdot)$ is one of the simplest nonlinear cases. Similar to MHE, CoMHE also serves as a regularization in neural networks.

(d) Random Projection for CoMHE. Random projection is in fact one of the most straightforward way to reduce dimensionality while partially preserving the angular information. More specifically, we use a random mapping $g(\boldsymbol{v}) = \frac{\boldsymbol{P}\boldsymbol{v}}{\|\boldsymbol{P}\boldsymbol{v}\|}$ where

$\boldsymbol{P}\in\mathbb{R}^{(k+1)\times(d+1)}$ is a Gaussian distributed random matrix (each entry follows i.i.d. normal distribution). In order to reduce the variance, we use C random projection matrices to project the neurons and compute the hyperspherical energy separately:

$$\boldsymbol{E}_s^R(\hat{\boldsymbol{W}}_N):=\frac{1}{C}\sum_{c=1}^{C}\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} f_s\Big(\Big\|\frac{\boldsymbol{P}_c\hat{\boldsymbol{w}}_i}{\|\boldsymbol{P}_c\hat{\boldsymbol{w}}_i\|}-\frac{\boldsymbol{P}_c\hat{\boldsymbol{w}}_j}{\|\boldsymbol{P}_c\hat{\boldsymbol{w}}_j\|}\Big\|\Big) \tag{2.5}$$

where $\boldsymbol{P}_c,\forall c$ is a random matrix with each entry following the normal distribution $\mathcal{N}(0,1)$. According to the properties of normal distribution [23], every normalized row of the random matrix $\boldsymbol{P}$ is uniformly distributed on a hypersphere $\mathbb{S}^d$, which indicates that the projection matrix $\boldsymbol{P}$ is able to cover all the possible subspaces. Multiple projection matrices can also be interpreted as multi-view projection, because we are making use of information from multiple projection views. In fact, we do not necessarily need to average the energy for multiple projections, and instead we can use maximum operation (or some other meaningful aggregation operations). Then the objective becomes $\max_c\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} f_s(\|\frac{\boldsymbol{P}_c\hat{\boldsymbol{w}}_i}{\|\boldsymbol{P}_c\hat{\boldsymbol{w}}_i\|}-\frac{\boldsymbol{P}_c\hat{\boldsymbol{w}}_j}{\|\boldsymbol{P}_c\hat{\boldsymbol{w}}_j\|}\|)$. Considering that we aim to minimize this objective, the problem is in fact a min-max optimization. Note that, we will typically re-initialize the random projection matrices every certain number of iterations to avoid trivial solutions. Most importantly, using RP can provably preserve the angular similarity.

(e) Angle-preserving Projection for CoMHE. Recall that we aim to find a projection to project the neurons to a low-dimensional space that best preserves angular information. We transform the goal to an optimization:

$$\boldsymbol{P}^\star=\arg\min_{\boldsymbol{P}}\mathcal{L}_P:=\sum_{i\neq j}(\theta_{(\hat{\boldsymbol{w}}_i,\hat{\boldsymbol{w}}_j)}-\theta_{(\boldsymbol{P}\hat{\boldsymbol{w}}_i,\boldsymbol{P}\hat{\boldsymbol{w}}_j)})^2 \tag{2.6}$$

where $\boldsymbol{P}\in\mathbb{R}^{(k+1)\times(d+1)}$ is the projection matrix and $\theta_{(\boldsymbol{v}_1,\boldsymbol{v}_2)}$ denotes the angle between $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. For implementation convenience, we can replace the angle with the cosine value (*e.g.*, use $\cos(\theta_{(\hat{\boldsymbol{w}}_i,\hat{\boldsymbol{w}}_j)})$ to replace $\theta_{(\hat{\boldsymbol{w}}_i,\hat{\boldsymbol{w}}_j)}$), so that we can directly use the inner product of normalized vectors to measure the angular similarity. With $\hat{\boldsymbol{P}}$ obtained in Eq. 2.6, we use a nested loss function:

$$\begin{aligned}\boldsymbol{E}_s^A(\hat{\boldsymbol{W}}_N, \boldsymbol{P}^\star) &:= \sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} f_s\Big(\Big\| \frac{\boldsymbol{P}^\star\hat{\boldsymbol{w}}_i}{\|\boldsymbol{P}^\star\hat{\boldsymbol{w}}_i\|} - \frac{\boldsymbol{P}^\star\hat{\boldsymbol{w}}_j}{\|\boldsymbol{P}^\star\hat{\boldsymbol{w}}_j\|} \Big\|\Big) \\ \text{s.t.} \quad \boldsymbol{P}^\star &= \arg\min_{\boldsymbol{P}} \sum_{i\neq j} (\theta_{(\hat{\boldsymbol{w}}_i, \hat{\boldsymbol{w}}_j)} - \theta_{(\boldsymbol{P}\hat{\boldsymbol{w}}_i, \boldsymbol{P}\hat{\boldsymbol{w}}_j)})^2\end{aligned} \tag{2.7}$$

for which we propose two different ways to optimize the projection matrix $\boldsymbol{P}$. We can approximate $\boldsymbol{P}^\star$ using a few gradient descent updates. Specifically, we use two different ways to perform the optimization. Naively, we use a few gradient descent steps to update $\boldsymbol{P}$ in order to approximate $\boldsymbol{P}^\star$ and then update $\boldsymbol{W}_N$, which proceeds alternately. The number of iteration steps that we use to update $\boldsymbol{P}$ is a hyperparameter and needs to be determined by cross-validation. Besides the naive alternate one, we also use a different optimization of $\boldsymbol{W}_N$ by unrolling the gradient update of $\boldsymbol{P}$.

Alternating optimization. The alternating optimization is to optimize $\boldsymbol{P}$ alternately with the network parameters $\boldsymbol{W}_N$. Specifically, in each iteration of updating the network parameters, we update $\boldsymbol{P}$ every number of inner iterations and use it as an approximation to $\boldsymbol{P}^\star$ (the error depends on the number of gradient steps we take). Essentially, we are alternately solving two separate optimization problems for $\boldsymbol{P}$ and $\boldsymbol{W}_N$ with gradient descent.

Unrolled optimization. Instead of naively updating $\boldsymbol{W}_N$ with approximate $\boldsymbol{P}^\star$ in the alternating optimization, the unrolled optimization further unrolls the update rule of $\boldsymbol{P}$ and embed it within the optimization of network parameters $\boldsymbol{W}_N$. If we denote the CoMHE loss with a given projection matrix $\boldsymbol{P}$ as $\boldsymbol{E}_s^A(\boldsymbol{W}_N, \boldsymbol{P})$ which takes $\boldsymbol{W}_N$ and $\boldsymbol{P}$ as input, then the unrolled optimization is essentially optimizing $\boldsymbol{E}_s^A(\boldsymbol{W}_N, \boldsymbol{P} - \eta \cdot \frac{\partial \mathcal{L}_P}{\partial \boldsymbol{P}})$. It can also be viewed as minimizing the CoMHE loss after a single step of gradient descent *w.r.t.* the projection matrix. This optimization includes the computation of second-order partial derivatives. Note that, it is also

Method	Error (%)
Baseline	28.03
Orthogonal	27.01
SRIP [7]	25.80
MHE	26.75
HS-MHE	25.96
G-CoMHE	25.08
Adv-CoMHE	25.09
RP-CoMHE	24.39
RP-CoMHE (max)	24.77
AP-CoMHE (alter.)	24.95
AP-CoMHE (unroll)	**24.33**

Table 2.1: MHE variants on CIFAR-100.

possible to unroll multiple gradient descent steps as in [29, 78, 24].

2.1.3 Experiments and Results

(a) Image Recognition. We perform image recognition to show the improvement of regularizing CNNs with MHE and CoMHE. The goal is to show the superiority of our proposed method rather than achieving state-of-the-art accuracies on particular tasks. For all the experiments on CIFAR-10 and CIFAR-100 in this section, we use the same data augmentation as [38, 87]. For ImageNet-2012, we use the same data augmentation in [84]. We train all the networks using SGD with momentum 0.9. All the networks use BN [55] and ReLU if not otherwise specified. By default, all CoMHE variants are built upon half-space MHE.

Variants of MHE. We compare different variants of MHE and CoMHE with the same plain CNN-9. Specifically, we evaluate the baseline CNN without any regularization, half-space MHE (HS-MHE) which is the best MHE variant from [85], random projection CoMHE (RP-CoMHE), RP-CoMHE (max) that uses max instead of average for loss aggregation, angle-preserving projection CoMHE (AP-CoMHE), adversarial projection CoMHE (Adv-CoMHE) and group CoMHE (G-CoMHE) on CIFAR-100. For RP, we set the projection dimension to 30 (*i.e.*, $k=29$) and the

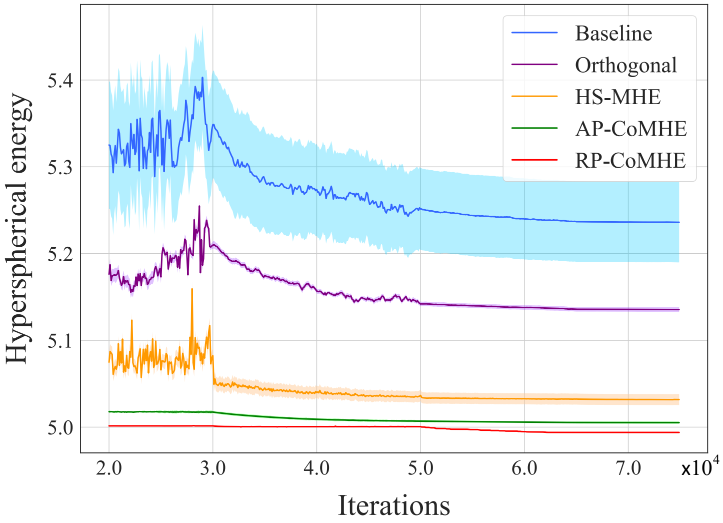


Figure 2.4: Hyperspherical energy during training. All networks are initialized with the same random weights, so the hyperspherical energy is the same before the training starts.

number of projection to 5 (*i.e.*, $C=5$). For AP, the number of projection is 1 and the projection dimension is set to 30. For AP, we evaluate both alternating optimization and unrolled optimization. In alternating optimization, we update the projection matrix every 10 steps of network update. In unrolled optimization, we only unroll one-step gradient in the optimization. For G-CoMHE, we construct a group with every 8 consecutive channels. All these design choices are obtained using cross-validation. We will also study how these hyperparameters affect the performance in the following experiments. The results in Table 2.1 show that all of our proposed CoMHE variants can outperform the original half-space MHE by a large margin. The unrolled optimization in AP-CoMHE shows the significant advantage over alternating one and achieves the best accuracy. Both Adv-CoMHE and G-CoMHE achieve decent performance gain over HS-MHE, but not as good as RP-CoMHE and AP-CoMHE. Therefore, we will mostly focus on RP-CoMHE and AP-CoMHE in the remaining experiments.

Effectiveness of optimization. To verify that our CoMHE can better minimize the hyperspherical energy, we compute the hyperspherical energy $\boldsymbol{E}_2$ (Eq. 2.1)

Method	Res-18	Res-34	Res-50
baseline	32.95	30.04	25.30
Orthogonal [109]	32.65	29.74	25.19
Orthnormal [84]	32.61	29.75	25.21
SRIP [7]	32.53	29.55	24.91
MHE	32.50	29.60	25.02
HS-MHE	32.45	29.50	24.98
RP-CoMHE	31.90	29.38	**24.51**
AP-CoMHE	**31.80**	**29.32**	24.53

Table 2.2: Top-1 center crop error on ImageNet.

for baseline CNN and CNN regularized by orthogonal regularization, HS-MHE, RP-CoMHE and AP-CoMHE during training. Note that, we compute the original hyperspherical energy rather than the energy after projection. All the networks use exactly the same initialization (the initial hyperspherical energy is the same). The results are averaged over five independent runs. We show the hyperspherical energy after the 20000-th iteration, because at the beginning of the training, hyperspherical energy fluctuates dramatically and is unstable. From Fig. 2.4, one can observe that both RP-CoMHE and AP-CoMHE can better minimize the hyperspherical energy. RP-CoMHE can achieve the lowest energy with smallest standard deviation. From the absolute scale, the optimization gain is also very significant. In the high-dimensional space, the variance of hyperspherical energy is usually small (already close to the smallest energy value) and is already difficult to minimize.

Large-scale recognition on ImageNet-2012. We evaluate our method for image recognition on ImageNet-2012 [110]. We perform the experiment using ResNet-18, ResNet-34 and ResNet-50, and then report the top-1 validation error in Table 2.2. Our results show consistent and significant performance gain in all ResNet variants. Compared to the baselines, MHE and its variants can reduce the top-1 error for more than 1%. Since the computational overhead is almost neglectable, the performance gain is obtained without many efforts. Most importantly, as a plug-in regularization, MHE is shown to be architecture-agnostic and produces considerable accuracy gain.

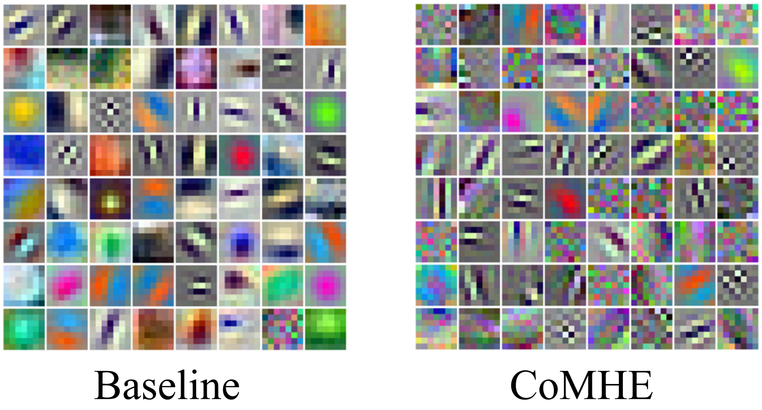


Figure 2.5: Visualized first-layer filters.

Besides the accuracy improvement, we also visualize in Fig. 2.5 the 64 filters in the first-layer learned by the baseline ResNet and the proposed CoMHE-regularized ResNet. The filters look quite different after we regularize the network using CoMHE. Each filter learned by baseline focuses on a particular local pattern (*e.g.*, edge, color and shape) and each one has a clear local semantic meaning. In contrast, filters learned by CoMHE focuses more on edges, textures and global patterns which do not necessarily have a clear local semantic meaning. However, from a representation basis perspective, having such global patterns may be beneficial to the recognition accuracy. We also observe that filters learned by CoMHE pay less attention to color.

(b) Point Cloud Recognition. We evaluate CoMHE on point cloud recognition. Our goal is to validate the effectiveness of CoMHE on a totally different network architecture with a different form of input data structure, rather than achieving state-of-the-art performance on point cloud recognition. To this end, we conduct experiments on widely used neural networks that handles point clouds: PointNet [101] (PN) and PointNet++ [102] (PN++). We combine half-space MHE, RP-CoMHE and AP-CoMHE into PN (without T-Net), PN (with T-Net) and PN++. We test the performance on ModelNet-40 [135]. Specifically, since PN can be viewed as 1×1 convolutions before the max pooling layer, we can apply all these MHE variants simi-

Method	PN	PN (T)	PN++
Original	87.1	89.20	90.07
MHE	87.31	89.33	90.25
HS-MHE	87.44	89.41	90.31
RP-CoMHE	87.82	89.69	90.52
AP-CoMHE	**87.85**	**89.70**	**90.56**

Table 2.3: Accuracy (%) on ModelNet-40.

Method	LFW	MegaFace
Softmax Loss	97.88	54.86
Softmax+Contrastive [122]	98.78	65.22
Triplet Loss [112]	98.70	64.80
L-Softmax Loss [79]	99.10	67.13
Softmax+Center Loss [132]	99.05	65.49
CosineFace [129, 128]	**99.10**	**75.10**
SphereFace	99.42	72.72
SphereFace+ (ours)	**99.47**	**73.03**

Table 2.4: Comparison to state-of-the-art face recognition methods.

larly to CNN. After the max pooling layer, there is a standard fully connected network where we can still apply the MHE variants. We compare the performance of regularizing PN and PN++ with half-space MHE, RP-CoMHE or AP-CoMHE. Table 2.3 shows that all MHE variants consistently improve PN and PN++, while RP-CoMHE and AP-CoMHE again perform the best among all. We demonstrate that CoMHE is generally useful for different types of input data (not limit to images) and different types of neural networks. CoMHE is also useful in graph neural networks.

(c) Face Recognition.

We have shown that full-space MHE for output layers can encourage classifier neurons to distribute more evenly on hypersphere and therefore improve inter-class feature separability. Intuitively, the classifier neurons serve as the approximate center for features from each class, and can therefore guide the feature learning. We also observe that open-set face recognition (*e.g.*, face verification) requires the feature centers to be as separable as possible [82]. This connection inspires us to apply

MHE to face recognition. Specifically, we propose *SphereFace+* by applying MHE to SphereFace [82]. The objective of SphereFace, angular softmax loss (ℓ_{SF}) that encourages intra-class feature compactness, is naturally complementary to that of MHE. The objective function of SphereFace+ is defined as:

$$\mathcal{L}_{\text{SF+}} = \underbrace{\frac{1}{m}\sum_{j=1}^{m}\ell_{\text{SF}}(\langle \boldsymbol{w}_i^{\text{out}}, \boldsymbol{x}_j\rangle_{i=1}^{c}, \boldsymbol{y}_j, m_{\text{SF}})}_{\text{softmax: promoting } \textbf{intra-class compactness}} \underbrace{+\lambda_{\text{M}}\cdot\frac{1}{m(N-1)}\sum_{i=1}^{m}\sum_{j=1,j\neq y_i}^{N} f_s(\|\hat{\boldsymbol{w}}_{y_i}^{\text{out}} - \hat{\boldsymbol{w}}_j^{\text{out}}\|)}_{\text{MHE: promoting } \textbf{inter-class separability}} \tag{2.8}$$

where c is the number of classes, m is the mini-batch size, N is the number of classifier neurons, $\boldsymbol{x}_i$ the deep feature of the i-th face (y_i is its groundtruth label), and $\boldsymbol{w}_i^{\text{out}}$ is the i-th classifier neuron. m_{SF} is a hyperparameter for SphereFace, controlling the degree of intra-class feature compactness (*i.e.*, the size of the angular margin). Because face datesets usually have thousands of identities, we will use the data-dependent mini-batch approximation MHE as shown in Eq. 2.8 in the output layer to reduce computational cost. MHE completes a missing piece for SphereFace by promoting the inter-class separability. SphereFace+ consistently outperforms SphereFace, and achieves state-of-the-art performance on both LFW [50] and MegaFace [62] datasets. We compare our methods with some widely used loss functions. All these compared methods use SphereFace-64 network that are trained with CASIA dataset. All the results are given in Table 2.4 computed without model ensemble and PCA. Compared to the other state-of-the-art methods, SphereFace+ achieves the best accuracy on LFW dataset, while being comparable to the best accuracy on MegaFace dataset. Current state-of-the-art face recognition methods [128, 82, 129, 25, 89] usually only focus on compressing the intra-class features, which makes MHE a potentially useful tool in order to further improve these face recognition methods.

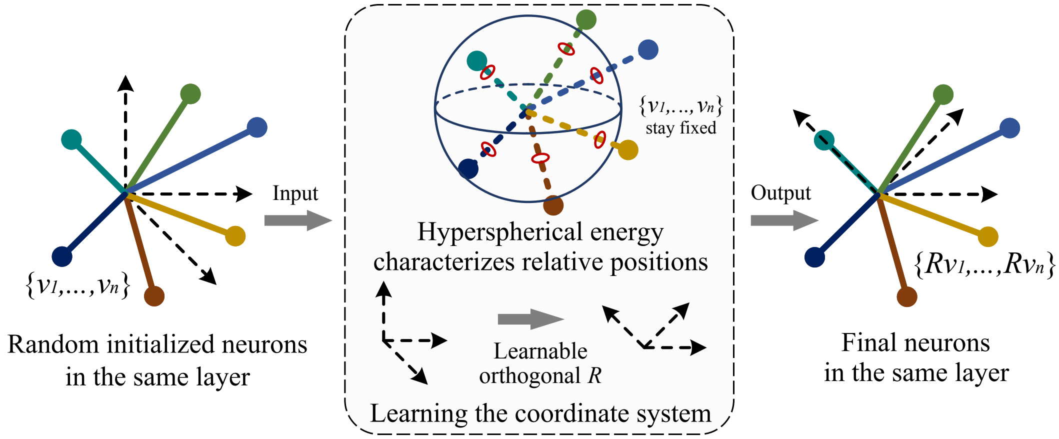

Input
Output
$\{v_1,\ldots,v_n\}$
Random initialized neurons in the same layer
$\{v_1,\ldots,v_n\}$ stay fixed
Hyperspherical energy characterizes relative positions
Learnable orthogonal R
Learning the coordinate system
$\{Rv_1,\ldots,Rv_n\}$
Final neurons in the same layer

Figure 2.6: Overview of the orthogonal over-parameterized training framework. OPT learns an orthogonal transformation for each layer in the neural network, while keeping the randomly initialized neuron weights fixed.

2.2 Orthogonal Training on Hypersphere

2.2.1 Overview

The inductive bias encoded in a neural network is generally determined by two major aspects: how the neural network is structured (*i.e.*, network architecture) and how the neural network is optimized (*i.e.*, training algorithm). For the same network architecture, using different training algorithms could lead to a dramatic difference in generalization performance [63, 105] even if the training loss is close to zero, implying that different training procedures lead to different inductive biases. Therefore, how to effectively train a neural network that generalize well remains an open challenge. Recent theories [33, 34, 61, 76] suggest the importance of over-parameterization in linear neural networks. There is also strong empirical evidence [26, 88] that over-parameterzing the convolutional filters under some regularity is beneficial to generalization. Our work aims to leverage the power of over-parameterization and explore more intrinsic structural priors in order to train a well-performing neural network.

Motivated by this goal, we propose a generic orthogonal over-parameterized training (OPT) framework for neural networks. Different from conventional neural training, OPT over-parameterizes a neuron $\boldsymbol{w}\in\mathbb{R}^d$ with the multiplication of a learnable layer-shared orthogonal matrix $\boldsymbol{R}\in\mathbb{R}^{d\times d}$ and a fixed randomly-initialized weight vector $\boldsymbol{v}\in\mathbb{R}^d$, and it follows that the equivalent weight for the neuron is $\boldsymbol{w}=\boldsymbol{R}\boldsymbol{v}$. Once each element of the neuron weight $\boldsymbol{v}$ has been randomly initialized by a zero-mean Gaussian distribution [37, 31], we fix them throughout the entire training process. Then OPT learns a layer-shared orthogonal transformation $\boldsymbol{R}$ that is applied to all the neurons (in the same layer). An illustration of OPT is given in Fig. 2.6. In contrast to standard neural training, OPT decomposes the neuron into an orthogonal transformation $\boldsymbol{R}$ that learns a proper coordinate system, and a weight vector $\boldsymbol{v}$ that controls the specific position of the neuron. Essentially, the weights $\{\boldsymbol{v}_1,\cdots,\boldsymbol{v}_n\in\mathbb{R}^d\}$ of different neurons determine the relative positions, while the layer-shared orthogonal matrix $\boldsymbol{R}$ specifies the coordinate system. Such a decoupled parameterization enables strong modeling flexibility.

Another motivation of OPT comes from an empirical observation that neural networks with lower *hyperspherical energy* generalize better [86]. Hyperspherical energy quantifies the diversity of neurons on a hypersphere, and essentially characterizes the relative positions among neurons via this form of diversity. [86] introduces hyperspherical energy as a regularization in the network but do not guarantee that the hyperspherical energy can be effectively minimized (due to the existence of data fitting loss). To address this issue, we leverage the property of hyperspherical energy that it is independent of the coordinate system in which the neurons live and only depends on their relative positions. Specifically, we prove that, if we randomly initialize the neuron weight $\boldsymbol{v}$ with certain distributions, these neurons are guaranteed to attain minimum hyperspherical energy in expectation. It follows that OPT maintains the minimum energy during training by learning a coordinate system (*i.e.*, layer-shared orthogonal matrix) for the neurons. Therefore, OPT is able to provably minimize the hyperspherical energy.

2.2.2 Proposed Method

(a) General Framework. OPT parameterizes the neuron as the multiplication of an orthogonal matrix $\boldsymbol{R}\in\mathbb{R}^{d\times d}$ and a neuron weight vector $\boldsymbol{v}\in\mathbb{R}^{d}$, and the equivalent neuron weight becomes $\boldsymbol{w}=\boldsymbol{R}\boldsymbol{v}$. The output $\hat{y}$ of this neuron can be represented by $\hat{y}=(\boldsymbol{R}\boldsymbol{v})^{\top}\boldsymbol{x}$ where $\boldsymbol{x}\in\mathbb{R}^{d}$ is the input vector. In OPT, we typically fix the randomly initialized neuron weight $\boldsymbol{v}$ and only learn the orthogonal matrix $\boldsymbol{R}$. In contrast, the standard neuron is directly formulated as $\hat{y}=\boldsymbol{v}^{\top}\boldsymbol{x}$, where the weight vector $\boldsymbol{v}$ is learned via back-propagation in training.

As an illustrative example, we consider a linear MLP with a loss function $\mathcal{L}$ (*e.g.*, the least squares loss: $\mathcal{L}(e_1,e_2)=(e_1-e_2)^2$). Specifically, the learning objective of the standard training is $\min_{\{\boldsymbol{v}_i,u_i,\forall i\}}\sum_{j=1}^{m}\mathcal{L}\big(y,\sum_{i=1}^{n}u_i\boldsymbol{v}_i^{\top}\boldsymbol{x}_j\big)$, while differently, our OPT is formulated as:

$$\min_{\{\boldsymbol{R},u_i,\forall i\}}\sum_{j=1}^{m}\mathcal{L}\big(y,\sum_{i=1}^{n}u_i(\boldsymbol{R}\boldsymbol{v}_i)^{\top}\boldsymbol{x}_j\big)\quad \text{s.t. } \boldsymbol{R}^{\top}\boldsymbol{R}=\boldsymbol{R}\boldsymbol{R}^{\top}=\boldsymbol{I} \tag{2.9}$$

where $\boldsymbol{v}_i\in\mathbb{R}^{d}$ is the i-th neuron in the first layer, and $\boldsymbol{u}=\{u_1,\cdots,u_n\}\in\mathbb{R}^{n}$ is the output neuron in the second layer. In OPT, each element of $\boldsymbol{v}_i$ is usually sampled from a zero-mean Gaussian distribution (*e.g.*, both Xavier [31] and Kaiming [37] initializations belong to this class), and is fixed throughout the entire training process. In general, OPT learns an orthogonal matrix that is applied to all the neurons instead of learning the individual neuron weight. Note that, we usually do not apply OPT to neurons in the output layer (*e.g.*, $\boldsymbol{u}$ in this MLP example, and the final linear classifiers in CNNs), since it makes little sense to fix a set of random linear classifiers. Therefore, the central problem is how to learn these layer-shared orthogonal matrices.

(b) Hyperspherical Energy Perspective. One of the most important properties of OPT is its invariance to hyperspherical energy. Based on [86], the hyperspherical energy of n neurons is defined as $\boldsymbol{E}(\hat{\boldsymbol{v}}_i|_{i=1}^{n})=\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\|\hat{\boldsymbol{v}}_i-\hat{\boldsymbol{v}}_j\|^{-1}$

in which $\hat{\boldsymbol{v}}_i = \frac{\boldsymbol{v}_i}{\|\boldsymbol{v}_i\|}$ is the i-th neuron weight projected onto the unit hypersphere $\mathbb{S}^{d-1} = \{\boldsymbol{v} \in \mathbb{R}^d | \, \|\boldsymbol{v}\| = 1\}$. Hyperspherical energy is used to characterize the diversity of n neurons on a unit hypersphere. Assume that we have n neurons in one layer, and we have learned an orthogonal matrix $\boldsymbol{R}$ for these neurons. The hyperspherical energy of these n OPT-trained neurons is:

$$\begin{aligned} \boldsymbol{E}(\hat{\boldsymbol{R}}\hat{\boldsymbol{v}}_i|_{i=1}^n) &= \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \|\boldsymbol{R}\hat{\boldsymbol{v}}_i - \boldsymbol{R}\hat{\boldsymbol{v}}_j\|^{-1} \\ \big(\text{since } \|\boldsymbol{R}\|^{-1} = 1\big) \quad &= \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \|\hat{\boldsymbol{v}}_i - \hat{\boldsymbol{v}}_j\|^{-1} = \boldsymbol{E}(\hat{\boldsymbol{v}}_i|_{i=1}^n) \end{aligned} \tag{2.10}$$

which verifies that the hyperspherical energy does not change in OPT. Moreover, [86] proves that minimum hyperspherical energy corresponds to the uniform distribution over the hypersphere. As a result, if the initialization of the neurons in the same layer follows the uniform distribution over the hypersphere, then we can guarantee that the hyperspherical energy is minimal in a probabilistic sense.

Theorem 2.2.1. *For the neuron $\boldsymbol{h}=\{h_1, \cdots, h_d\}$ where $h_i, \forall i$ are initialized i.i.d. following a zero-mean Gaussian distribution (i.e., $h_i \sim N(0, \sigma^2)$), the projections onto a unit hypersphere $\hat{\boldsymbol{h}}=\boldsymbol{h}/\|\boldsymbol{h}\|$ where $\|\boldsymbol{h}\|=(\sum_{i=1}^d h_i^2)^{1/2}$ are uniformly distributed on the unit hypersphere $\mathbb{S}^{d-1}$. The neurons with minimum hyperspherical energy attained asymptotically approach the uniform distribution on $\mathbb{S}^{d-1}$.*

Theorem 2.2.1 proves that, as long as we initialize the neurons in the same layer with zero-mean Gaussian distribution, the resulting hyperspherical energy is guaranteed to be small (*i.e.*, the expected energy is minimal). It is because the neurons are uniformly distributed on the unit hypersphere and hyperspherical energy quantifies the uniformity on the hypersphere in some sense. More importantly, prevailing neuron initializations such as [31] and [37] are zero-mean Gaussian distribution. Therefore, our neurons naturally have low hyperspherical energy from the beginning.

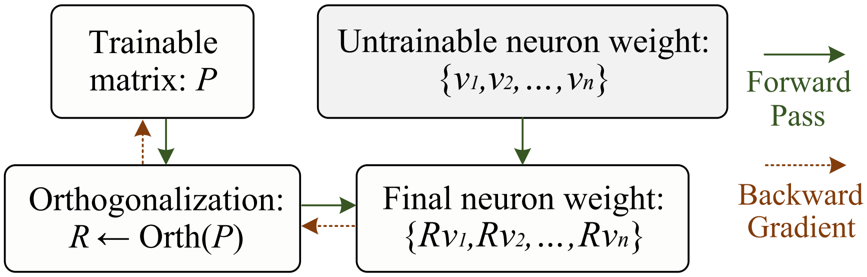

Trainable matrix: P
Untrainable neuron weight: $\{v_1, v_2, \ldots, v_n\}$
Forward Pass
Orthogonalization: $R \leftarrow \mathrm{Orth}(P)$
Final neuron weight: $\{Rv_1, Rv_2, \ldots, Rv_n\}$
Backward Gradient

Figure 2.7: Unrolled orthogonalization.

(c) Unrolling Orthogonalization Algorithms. In order to learn the orthogonal transformation, we unroll classic orthogonalization algorithms and embed them into the neural network such that the training can be performed in an end-to-end fashion. We need to make every step of the orthogonalization algorithm differentiable, as shown in Fig. 2.7.

Gram-Schmidt process. This method takes a linearly independent set and eventually produces an orthogonal set based on it. The Gram-Schmidt Process (GS) usually takes the following steps to orthogonalize a set of vectors $\{\boldsymbol{u}_1,\cdots,\boldsymbol{u}_n\}\in\mathbb{R}^{n\times n}$ and obtain an orthonormal set $\{\boldsymbol{e}_1,\cdots,\boldsymbol{e}_i,\cdots,\boldsymbol{e}_n\}\in\mathbb{R}^{n\times n}$. First, when $i=1$, we have $\boldsymbol{e}_1=\frac{\tilde{\boldsymbol{e}}_1}{\|\tilde{\boldsymbol{e}}_1\|}$ where $\tilde{\boldsymbol{e}}_1=\boldsymbol{u}_1$. Then, when $n\geq i\geq 2$, we have $\boldsymbol{e}_i=\frac{\tilde{\boldsymbol{e}}_i}{\|\tilde{\boldsymbol{e}}_i\|}$ where $\tilde{\boldsymbol{e}}_i=\boldsymbol{u}_i-\sum_{j=1}^{i-1}\mathrm{Proj}_{\boldsymbol{e}_j}(\boldsymbol{u}_i)$. Note that, $\mathrm{Proj}_{\boldsymbol{b}}(\boldsymbol{a})=\frac{\langle\boldsymbol{a},\boldsymbol{b}\rangle}{\langle\boldsymbol{b},\boldsymbol{b}\rangle}\boldsymbol{b}$ is defined as the projection operator.

Householder Reflection. A Householder reflector is defined as $\boldsymbol{H}=\boldsymbol{I}-2\frac{\boldsymbol{u}\boldsymbol{u}^\top}{\|\boldsymbol{u}\|^2}$ where $\boldsymbol{u}$ is perpendicular to the reflection hyperplane. In QR factorization, Householder reflection (HR) is used to transform a (non-singular) square matrix into an orthogonal matrix and an upper triangular matrix. Given a matrix $\boldsymbol{U}=\{\boldsymbol{u}_1,\cdots,\boldsymbol{u}_n\}\in\mathbb{R}^{n\times n}$, we consider the first column vector $\boldsymbol{u}_1$. We use Householder reflector to transform $\boldsymbol{u}_1$ to $\boldsymbol{e}_1=\{1,0,\cdots,0\}$. Specifically, we construct an orthogonal matrix $\boldsymbol{H}_1$ with $\boldsymbol{H}_1=\boldsymbol{I}-2\frac{(\boldsymbol{u}_1-\|\boldsymbol{u}_1\|\boldsymbol{e}_1)(\boldsymbol{u}_1-\|\boldsymbol{u}_1\|\boldsymbol{e}_1)^\top}{\|\boldsymbol{u}_1-\|\boldsymbol{u}_1\|\boldsymbol{e}_1\|^2}$. The first column of $\boldsymbol{H}_1\boldsymbol{U}$ becomes $\{\|\boldsymbol{u_1}\|,0,\cdots,0\}$. At the k-th step, we can view the sub-matrix $\boldsymbol{U}_{(k:n,k:n)}$ as a new $\boldsymbol{U}$, and use the same procedure to construct the Householder transformation $\tilde{\boldsymbol{H}}_k\in\mathbb{R}^{(n-k)\times(n-k)}$. We construct the final Householder transformation as $\boldsymbol{H}_k=\mathrm{Diag}(\boldsymbol{I}_k,\tilde{\boldsymbol{H}}_k)$. Now we can gradually transform $\boldsymbol{U}$ to an upper triangular matrix with n Householder reflections. Therefore, we have that $\boldsymbol{H}_n \cdots \boldsymbol{H}_2 \boldsymbol{H}_1 \boldsymbol{U} = \boldsymbol{R}^{\text{up}}$ where $\boldsymbol{R}^{\text{up}}$ is an upper triangular matrix and the obtained orthogonal set is $\boldsymbol{Q}^\top = \boldsymbol{H}_n \cdots \boldsymbol{H}_2 \boldsymbol{H}_1$.

Löwdin's Symmetric Orthogonalization. Let the matrix $\boldsymbol{U} = \{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n\} \in \mathbb{R}^{n \times n}$ be a given set of linearly independent vectors in an n-dimensional space. A non-singular linear transformation $\boldsymbol{A}$ can transform the basis $\boldsymbol{U}$ to an orthogonal basis $\boldsymbol{R}$: $\boldsymbol{R} = \boldsymbol{U}\boldsymbol{A}$. The matrix $\boldsymbol{R}$ will be orthogonal if $\boldsymbol{R}^\top \boldsymbol{R} = (\boldsymbol{U}\boldsymbol{A})^\top \boldsymbol{U}\boldsymbol{A} = \boldsymbol{A}^\top \boldsymbol{M} \boldsymbol{A} = \boldsymbol{I}$ where $\boldsymbol{M} = \boldsymbol{U}^\top \boldsymbol{U}$ is the Gram matrix of the given set $\boldsymbol{U}$. We obtain a general solution to the orthogonalization problem via the substitution: $\boldsymbol{A} = \boldsymbol{M}^{-\frac{1}{2}} \boldsymbol{B}$ where $\boldsymbol{B}$ is an arbitrary unitary matrix. The specific choice $\boldsymbol{B} = \boldsymbol{I}$ gives the Löwdin's symmetric orthogonalization (LS): $\boldsymbol{R} = \boldsymbol{U}\boldsymbol{M}^{-\frac{1}{2}}$. We can analytically obtain the symmetric orthogonalization from the singular value decomposition: $\boldsymbol{U} = \boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{V}^\top$. Then LS gives $\boldsymbol{R} = \boldsymbol{W}\boldsymbol{V}^\top$ as the orthogonal set for $\boldsymbol{U}$. LS has a unique property which the other orthogonalizations do not have. The orthogonal set resembles the original set in a nearest-neighbour sense. More specifically, LS guarantees that $\sum_i \|\boldsymbol{R}_i - \boldsymbol{U}_i\|^2$ (where $\boldsymbol{R}_i$ and $\boldsymbol{U}_i$ are the i-th column of $\boldsymbol{R}$ and $\boldsymbol{U}$, respectively) is minimized. Intuitively, LS indicates the gentlest pushing of the directions of the vectors in order to get them orthogonal to each other.

(d) Orthogonal Parameterization. A convenient way to ensure orthogonality while learning the matrix $\boldsymbol{R}$ is to use a special parameterization that inherently guarantees orthogonality. The exponential parameterization use $\boldsymbol{R} = \exp(\boldsymbol{W})$ (where $\exp(\cdot)$ denotes the matrix exponential) to represent an orthogonal matrix from a skew-symmetric matrix $\boldsymbol{W}$. The Cayley parameterization (CP) is a Padé approximation of the exponential parameterization, and is a more natural choice due to its simplicity. CP uses the following transform to construct an orthogonal matrix $\boldsymbol{R}$ from a skew-symmetric matrix $\boldsymbol{W}$: $\boldsymbol{R} = (\boldsymbol{I} + \boldsymbol{W})(\boldsymbol{I} - \boldsymbol{W})^{-1}$ where $\boldsymbol{W} = -\boldsymbol{W}^\top$. We note that CP only produces the orthogonal matrices with determinant 1, which belong to the special orthogonal group and thus $\boldsymbol{R}\in SO(n)$. Specifically, it suffices to learn the upper or lower triangular of the matrix $\boldsymbol{W}$ with unconstrained optimization to obtain a desired orthogonal matrix $\boldsymbol{R}$. Cayley parameterization does not cover the entire orthogonal group and is less flexible in terms of representation power, which serves as an explicit regularization for the neurons.

(e) Orthogonality-Preserving Gradient Descent. An alternative way to guarantee orthogonality is to modify the gradient update for the matrix $\boldsymbol{R}$. The idea is to initialize $\boldsymbol{R}$ with an arbitrary orthogonal matrix and then ensure each gradient update is to apply an orthogonal transformation to $\boldsymbol{R}$. It is essentially conducting gradient descent on the Stiefel manifold [73, 133, 134, 70, 5, 41, 58]. Given a matrix $\boldsymbol{U}_{(0)}\in\mathbb{R}^{n\times n}$ that is initialized as an orthogonal matrix, we aim to construct an orthogonal transformation as the gradient update. We use the Cayley transform to compute a parametric curve on the Stiefel manifold $\mathcal{M}_s=\{\boldsymbol{U}\in\mathbb{R}^{n\times n}:\boldsymbol{U}^\top\boldsymbol{U}=\boldsymbol{I}\}$ with a specific metric via a skew-symmetric matrix $\boldsymbol{W}$ and use it as the update rule:

$$\boldsymbol{Y}(\lambda)=(\boldsymbol{I}-\frac{\lambda}{2}\boldsymbol{W})^{-1}(\boldsymbol{I}+\frac{\lambda}{2}\boldsymbol{W})\boldsymbol{U}_{(i)},\ \boldsymbol{U}_{(i+1)}=\boldsymbol{Y}(\lambda) \tag{2.11}$$

where $\hat{\boldsymbol{W}}=\nabla f(\boldsymbol{U}_{(i)})\boldsymbol{U}_{(i)}^\top-\frac{1}{2}\boldsymbol{U}_{(i)}(\boldsymbol{U}_{(i)}^\top\nabla f(\boldsymbol{U}_{(i)}\boldsymbol{U}_{(i)}^\top)$ and $\boldsymbol{W}=\hat{\boldsymbol{W}}-\hat{\boldsymbol{W}}^\top$. $\boldsymbol{U}_{(i)}$ denotes the orthogonal matrix in the i-th iteration. $\nabla f(\boldsymbol{U}_{(i)})$ denotes the original gradient of the loss function *w.r.t.* $\boldsymbol{U}_{(i)}$. We term this gradient update as orthogonal-preserving gradient descent (OGD). To reduce the computational cost of the matrix inverse in Eq. 2.11, we use an iterative method [73] to approximate the Cayley transform without matrix inverse. We arrive at the fixed-point iteration:

$$\boldsymbol{Y}(\lambda)=\boldsymbol{U}_{(i)}+\frac{\lambda}{2}\boldsymbol{W}\big(\boldsymbol{U}_{(i)}+\boldsymbol{Y}(\lambda)\big) \tag{2.12}$$

which converges to the closed-form Cayley transform with a rate of $o(\lambda^{2+n})$ (n is the iteration number). In practice, two iterations suffice for a reasonable approximation accuracy.

(f) Relaxation to Orthogonal Regularization. Alternatively, we also consider relaxing the original optimization with an orthogonality constraint to an unconstrained optimization with orthogonality regularization (OR). Specifically, we remove the orthogonality constraint, and adopt an orthogonality regularization for $\boldsymbol{R}$, *i.e.*, $\|\boldsymbol{R}^\top\boldsymbol{R}-\boldsymbol{I}\|_F^2$. However, OR cannot guarantee the energy stays unchanged. Taking Eq. 2.9 as an example, the objective becomes

$$\min_{\boldsymbol{R},u_i,\forall i}\sum_{j=1}^{m}\mathcal{L}\big(y,\sum_{i=1}^{n}u_i(\boldsymbol{R}\boldsymbol{v}_i)^\top\boldsymbol{x}_j\big)+\beta\|\boldsymbol{R}^\top\boldsymbol{R}-\boldsymbol{I}\|_F^2 \tag{2.13}$$

where β is a hyperparameter. This serves as an relaxation of the original OPT objective. Note that, OR is imposed to $\boldsymbol{R}$ instead of neurons and is quite different from the existing orthogonality regularization on neurons [83, 7, 52, 136, 11].

(g) Towards Better Scalablity for OPT.

If the dimension of neurons becomes extremely large, then the orthogonal matrix to transform the neurons will also be large. Therefore, it may take large GPU memory and time to train the neural networks with the original OPT. To address this, we propose a scalable variant – stochastic OPT (S-OPT). The key idea of S-OPT is to randomly select some dimensions from the neurons in the same layer and construct a small orthogonal matrix to transform these dimensions together. The selection of dimensions is stochastic in each outer iteration, so a small orthogonal matrix is sufficient to cover all the neuron dimensions. S-OPT aims to approximate a large orthogonal transformation for all the neuron dimensions with many small orthogonal transformations for random subsets of these dimensions, which shares similar spirits with Givens rotation. The approximation will be more accurate when the procedure is randomized over many times. Fig. 2.8 compares the size of the orthogonal matrix

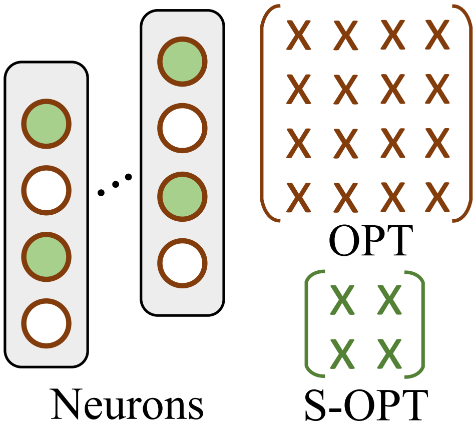


Figure 2.8: Illustration of S-OPT.

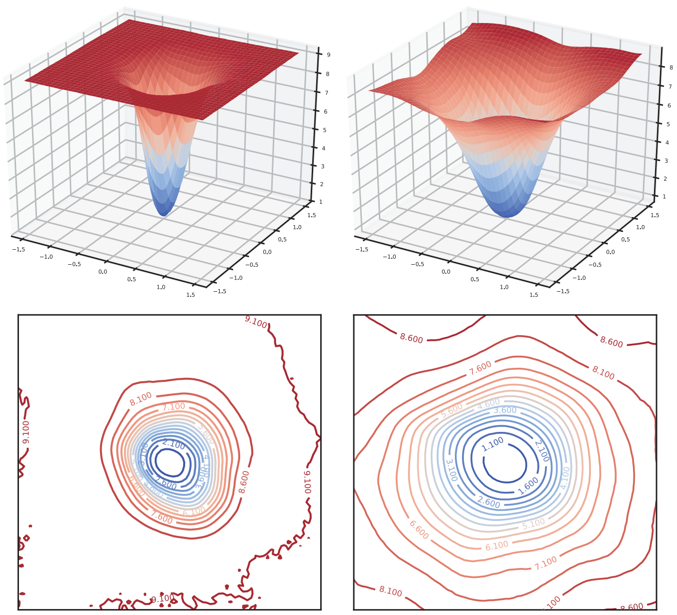
in OPT and S-OPT. The orthogonal matrix in OPT is of size $d \times d$, while the orthogonal matrix in S-OPT is of size $p \times p$ where p is usually much smaller than d. Most importantly, S-OPT can still preserve the low hyperspherical energy of neurons because of the following result.

Theorem 2.2.2. *For n d-dimensional neurons, selecting any p ($p \leq d$) dimensions and applying an shared orthogonal transformation ($p \times p$ orthogonal matrix) to these p dimensions of all neurons will not change the hyperspherical energy.*

A description of S-OPT is given in Algorithm 1. S-OPT has outer and inner iterations. In each inner iteration, the training is almost the same as OPT, except that the orthogonal matrix transforms a subset of the dimensions and the learnable orthogonal matrix has to be re-initialized to an identity matrix. The selection of neuron dimension is randomized in every outer iteration such that all neuron dimensions can be sufficiently covered as the number of outer iterations increases. Therefore, given sufficient number of iterations, S-OPT will perform comparably to OPT, as empirically verified in Section 2.2.3. This OPT variant explores structure priors in $\boldsymbol{R}$ to improve parameter efficiency.

Algorithm 1: Stochastic OPT

```
for $i = 1, 2, \cdots, N_{\text{out}}$ do
    for $j = 1, 2, \cdots, N_{\text{in}}$ do
        1. Randomly select $p$ dimensions from $d$-dimensional neurons in the same layer.;
        2. Construct an orthogonal matrix $\boldsymbol{R}_p \in \mathbb{R}^{p\times p}$ and initialize it as identity matrix.;
        3. Update $\boldsymbol{R}_p$ by applying OPT with one iteration.;
    end
    4. Multiply $\boldsymbol{R}_p$ back to the $p$-dim sub-vectors from the $d$-dim neurons to transform
      these neurons.;
end
```



Standard training

OPT

Figure 2.9: Training loss landscapes.

(h) Local Landscape.

We follow [71] to visualize the loss landscapes of both standard training and OPT in Fig. 2.9. For standard training, we perturb the parameter space of all the neurons (*i.e.*, filters). For OPT, we perturb the parameter space of all the trainable matrices (*i.e.*, $\boldsymbol{P}$ in Fig. 2.7), because OPT does not directly learn neuron weights. The general idea is to use two random vectors (*e.g.*, normal distribution) to perturb the parameter space and obtain the loss value with the perturbed network parameters. The loss landscape of standard training has extremely sharp minima. The red region is very flat, leading to small gradients. In contrast, the loss landscape of OPT is much more smooth and convex with flatter minima, well matching the finding that flat minimizers generalize well [43, 14, 56].

2.2.3 Experiments Results

(a) Ablation Study on Orthogonality. We evaluate whether orthogonality in OPT is necessary. We use 6-layer and 9-layer CNN on CIFAR-100. Then we compare OPT with unconstrained over-parameterized training (UPT) which learns an unconstrained matrix $\boldsymbol{R}$ (with weight decay) using the same network. In Table 2.5, "FN" denotes whether the randomly initialized neuron weights are fixed in training. "LR" denotes whether the learnable matrix $\boldsymbol{R}$ is unconstrained ("U") or orthogonal ("GS" for Gram-Schmidt process). Table 2.5 shows that without orthogonality, UPT performs much worse than OPT. From Table 2.5, we can see that using fixed neuron weights is consistently better than learnable neuron weights in both UPT and OPT. It indicates that fixing the neuron weights can well maintain low hyperspherical energy and is beneficial to empirical generalization.

Method	FN	LR	CNN-6	CNN-9
Baseline	-	-	37.59	33.55
UPT	✗	U	48.47	46.72
UPT	✓	U	42.61	39.38
OPT	✗	GS	37.24	32.95
OPT	✓	GS	**33.02**	**31.03**

Table 2.5: Error (%) on CIFAR-100.

(b) Empirical Evaluation on OPT.

Multi-layer perceptrons. We evaluate OPT on MNIST with a 3-layer MLP. Table 2.6 shows the testing error with normal initialization (MLP-N) or Xavier initialization [31] (MLP-X). GS/HR/LS denote different orthogonalization unrolling. CP denotes Cayley parameterization. OGD denotes orthogonal-preserving gradient descent. OR denotes relaxed orthogonal regularization. All OPT variants outperform

Method	MNIST		CIFAR-100			
	MLP-N	MLP-X	CNN-6	CNN-9	ResNet-20	ResNet-32
Baseline	6.05	2.14	37.59	33.55	31.11	30.16
Orthogonal [11]	5.78	1.93	36.32	33.24	31.06	30.05
SRIP [7]	-	-	34.82	32.72	30.89	29.70
HS-MHE	5.57	1.88	34.97	32.87	30.98	29.76
OPT (GS)	**5.11**	**1.45**	**33.02**	**31.03**	30.49	29.34
OPT (HR)	5.31	1.60	35.67	32.75	30.73	29.56
OPT (LS)	5.32	1.54	34.48	31.22	30.51	29.42
OPT (CP)	5.14	1.49	33.53	31.28	**30.47**	**29.31**
OPT (OGD)	5.38	1.56	33.33	31.47	30.50	29.39
OPT (OR)	5.41	1.78	34.70	32.63	30.66	29.47

Table 2.6: Testing error (%) of OPT for MLPs and CNNs.

the others by a large margin.

Convolutional networks. We evaluate OPT with 6/9-layer plain CNNs and ResNet-20/32 [39] on CIFAR-100. All neurons (*i.e.*, convolution kernels) are initialized by [37]. BatchNorm is used by default. Table 2.6 shows that all OPT variants outperform both baseline and HS-MHE by a large margin. HS-MHE puts the hyperspherical energy into the loss function and naively minimizes it along with the CNN. We observe that OPT (HR) performs the worse among all OPT variants partially because of its intensive unrolling computation. OPT (GS) achieves the best testing error on CNN-6/9, while OPT (CP) achieves the best testing error on ResNet-20/34, implying that different OPT encodes different inductive bias.

Training dynamics. We look into how hyperspherical energy and testing error changes in OPT. Fig. 2.10 shows that the energy of the baseline will increase dramatically at the beginning and then gradually go down, but it still stays in a high value in the end. HS-MHE well reduces the energy at the end of the training. In contrast, OPT variants always maintain very small energy in training. OPT with GS, CP and OGD keep exactly the same energy as the random initialization, while OPT (OR) slightly increases the energy due to relaxation. All OPT variants converge efficiently and stably.

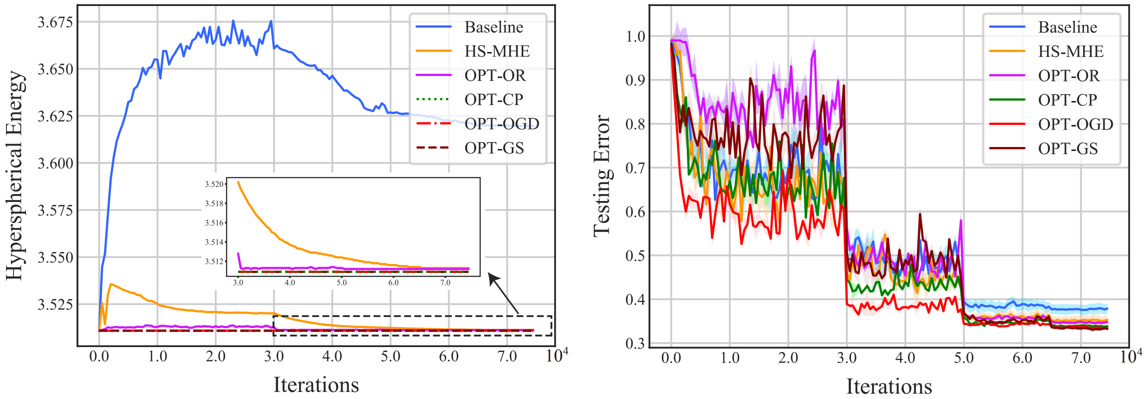


Figure 2.10: Training dynamics on CIFAR-100. Left: Hyperspherical energy vs. iteration. Right: Testing error vs. iteration.

Method	GCN		PointNet
	Cora	Pubmed	MN-40
Baseline	81.3	79.0	87.1
OPT (GS)	81.9	79.4	87.23
OPT (CP)	82.0	79.4	87.81
OPT (OGD)	**82.3**	**79.5**	**87.86**

Table 2.7: Geometric networks.

Geometric learning. We apply OPT to graph convolution network (GCN) [66] and point cloud network (PointNet) [101] for graph node and point cloud classification, respectively. The training of GCN and PointNet is conceptually similar to MLP. For GCN, we evaluate OPT on Cora and Pubmed datsets [113]. For PointNet, we conduct experiments on ModelNet-40 dataset [135]. Table 2.7 shows that OPT effectively improves both GCN and PointNet.

(c) Empirical Evaluation on S-OPT. S-OPT is a scalable OPT variant, and we evaluate its performance in terms of number of *trainable parameters* and testing error. Training parameters are learnable variables in training, and are different from model parameters in testing. In testing, all methods have the same number of model parameters. We perform classification on CIFAR-100 with CNN-6 and wide CNN-9. We also evaluate S-OPT with standard ResNet-18 on ImageNet. For S-OPT, we set the sampling dimension as 25% of the original neuron dimension in each layer. Ta-

Method	CIFAR-100				ImageNet	
	CNN-6	Params	Wide CNN-9	Params	ResNet-18	Params
Baseline	37.59	258K	28.03	2.99M	32.95	11.7M
HS-MHE [86]	34.97	258K	25.96	2.99M	32.50	11.7M
OPT (GS)	**33.02**	1.36M	OOM	16.2M	OOM	46.5M
S-OPT (GS)	33.70	**90.9K**	**25.59**	**1.04M**	**32.26**	**3.39M**

Table 2.8: OPT vs. S-OPT on CIFAR-100 & ImageNet.

$p =$	Error (%)	Params
d	OOM	16.2M
$d/4$	**25.59**	1.04M
$d/8$	28.61	278K
$d/16$	32.52	88.7K
16	33.03	27.0K
3	45.22	26.0K
0	60.64	**25.6K**

Table 2.9: Sampling dim.

ble 2.8 shows that S-OPT achieves a good trade-off between accuracy and scalability. More importantly, S-OPT can be applied to large neural networks, making OPT more useful in practice.

We study how the sampling dimension p affect the performance by performing classification with wide CNN-9 on CIFAR-100. In Table 2.9, $p = d/4$ means that we randomly sample 1/4 of the original neuron dimension in each layer, so p may vary in different layer. $p = 16$ means that we sample 16 dimensions in each layer. Note that there are 25.6K parameters used for the final classification layer, which can not be saved in S-OPT. Table 2.9 shows that S-OPT can achieve highly competitive accuracy with a reasonably large p.

(d) Large Categorical Training. Previously, OPT is not applied to the final classification layer, since it makes little sense to fix random classifiers and learn an orthogonal matrix to transform them. However, learning the classification layer can be costly with large number of classes. The number of trainable parameters of the classification layer grows linearly with the number of classes. To address this, OPT

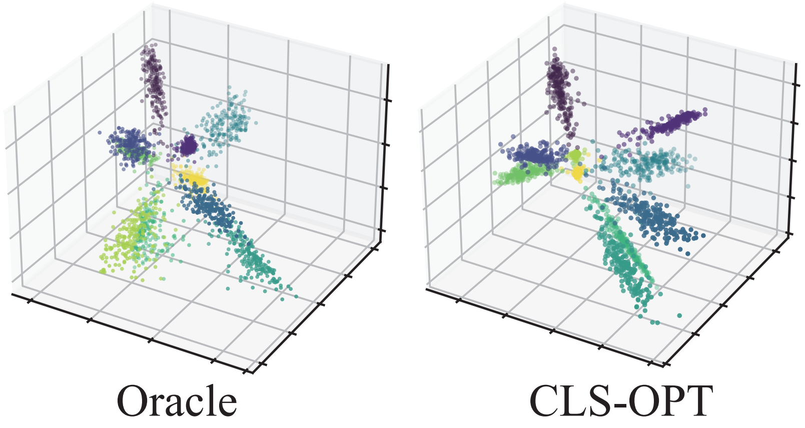


Figure 2.11: Feature visualization.

can be used to learn the classification layer, because its number of trainable parameters only depends on the classifier dimension. To be fair, we *only* learn the last classification layer with OPT and the other layers are normally learned (CLS-OPT). The oracle learns the entire network normally.

We intuitively compare the oracle and CLS-OPT by visualizing the deep MNIST features following [80]. The features are the direct outputs of CNN by setting the output dimension as 3. Figure 2.11 shows that even if CLS-OPT fixes randomly initialized classifiers, it can still learn discriminative and separable deep features.

We evaluate its performance on ImageNet with 1K classes. We use ResNet-18 with different output dimensions (A:128, B:512). Table 2.10 gives the top-5 test error (%) and "Params" denotes the number of trainable parameters in the classification layer. CLS-OPT performs well with far less trainable parameters.

Since face datasets usually contain large number of identities [35], it is natural to apply CLS-OPT to learn face embeddings. We train on CASIA [143] which has 0.5M face images of 10,572 identities, and test on LFW [51]. Since the training and testing sets do not overlap, the task well evaluates the generalizability of learned features. All methods use CNN-20 [81] and standard softmax loss. We set the output feature dimension as 512 or 1024. Table 2.11 validates CLS-OPT's effectiveness.

Method	ResNet-18A		ResNet-18B	
	Error	Params	Error	Params
Oracle	**18.08**	64.0K	12.12	512K
CLS-OPT	21.12	**8.13K**	**12.05**	**131K**

Table 2.10: CLS-OPT on ImageNet.

Method	512 Dim.		1024 Dim.	
	Error	Params	Error	Params
Oracle	**95.7**	5.41M	**96.4**	10.83M
CLS-OPT	94.9	**131K**	95.8	**524K**

Table 2.11: Verification (%) on LFW.

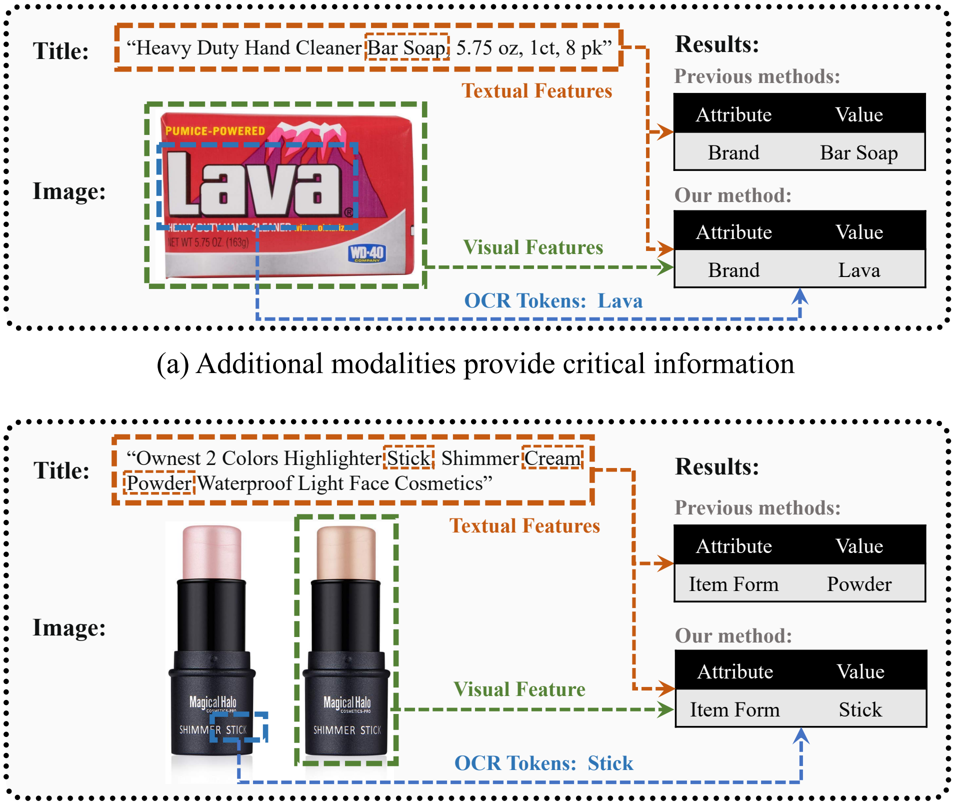
Chapter 3

Multi-modal Learning on Vision-and-Language Tasks

3.1 Problem Definition and Challenges

3.1.1 Overview

Multi-modal learning on Vision-and-Language task requires invariant representation of visual concepts and language semantics, and most importantly, the alignment and fusion between modalities. We propose a co-learning framework that fuse cross-modal knowledge and provide an industry-level application on large-scale e-commerce platform. Product attributes, such as brand, color and size, are critical product features that customers typically use to differentiate one product from another. Detailed and accurate attribute values can make it easier for customers to find the products that suit their needs and give them a better online shopping experience. It may also increase the customer base and revenue for e-commerce platforms. Therefore, accurate product attributes are essential for e-commerce applications such as product search and recommendation. Due to the manual and tedious nature of entering product information [27], the product attributes acquired from a massive number of



(a) Additional modalities provide critical information

(b) Different modalities cross-validates retrieved information

Figure 3.1: Compared to existing attribute value extraction works which focused on text-only input features, our approach is able to take extra multi-modal information (Visual Features and OCR Tokens) from product image as input.

retailers and manufacturers on e-commerce platforms are usually incomplete, noisy and prone to errors. To address this challenge, there has been a surge of interest in automatically extracting attribute values from readily available product profiles [148, 141, 60, 130, 27, 146]. Most of the existing works rely only on the textural cues obtained from the text descriptions in the product profiles, which is often far from sufficient to capture the target attributes.

In this work, we propose to leverage the product images in the attribute value extraction task. Besides the commonly used textural features from product descriptions, our approach simultaneously utilizes the generic visual features and the textual content hidden in the images, which are extracted through Optical Character

Recognition (OCR). We studied 30 popular product attributes applicable to different product categories, including electronics, home innovation, clothes, shoes, grocery, and health. We observed that over 20% of the attribute values that are missing from the corresponding Amazon web pages can only be identified from the product images. We illustrate this with an intuitive example in Figure 3.1. In general, we identified two main areas where the product images can be particularly useful:

- **Additional information:** Important cues are sometimes absent in the product text descriptions. In Figure 3.1(a), the brand of the product is *"Lava"*, which is prominently displayed in the product image but not mentioned in the product title. In this case, OCR could perfectly recover the missing brand from the product image.

- **Cross validation:** The product title is likely to contain multiple possible values for one target attribute and the product image could help in disambiguating the correct value. In Figure 3.1(b), for the attribute *"Item Form"*, the product title contains the word *"Stick"*, *"Cream"* and *"Powder"*. However, both the product shape and the word *"Stick"* in the image strongly indicate the correct value should be *"Stick"*.

3.1.2 Challenges

Despite the potential, leveraging product images for attribute value extraction remains a difficult problem and faces three main challenges:

- **C1: Cross-modality connections:** There are many intrinsic associations between product titles and images. An effective model needs to seamlessly and effectively make use of information from three modalities, including product images, texts in the images, and texts from product profiles.

- **C2: Domain-specific expressions:** The texts in the product images and product profiles are usually packed with certain phrases that are unique to a specific retail domain. For example, *"cleaning ripples"* in the category of toilet paper is a special wavy pattern to help with cleaning. *"free and clear"* in the category of detergent means that it is scent-free. In general, language models or word embeddings pre-trained on public corpora are unable to accurately capture and ground these domain-specific phrases.

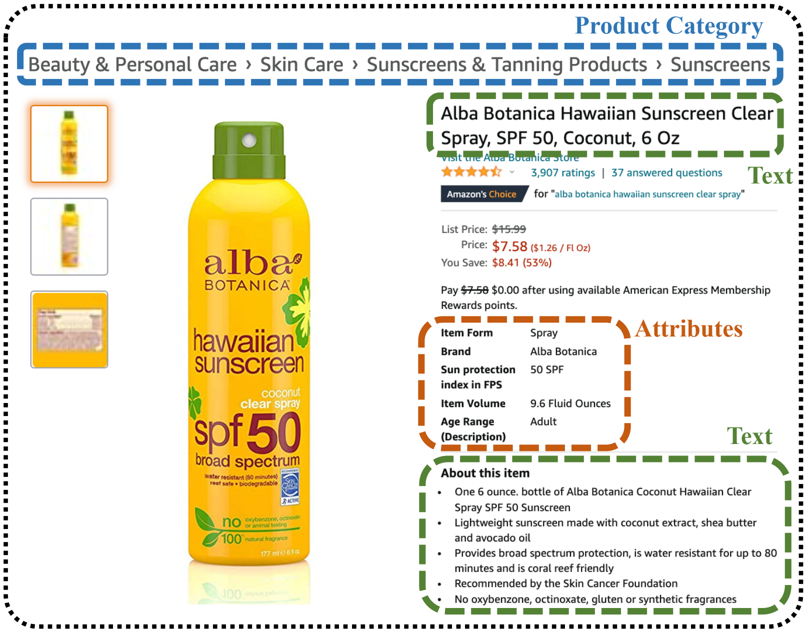
- **C3: Multiple categories:** For the same attribute, there may be little overlap between the attribute values for different product categories. For example, the vocabulary for the attribute *"size"* in T-shirts (*i.e.*, small, median, large, x-large) is completely different from baby diapers (*i.e.*, newborn, 1, 2, 3, etc.). Therefore, a model trained on one product category may generalize poorly to other categories.

Existing solutions for multi-modal information extraction [4, 65, 90, 124, 119] fall short in the e-commerce domain, as it cannot address challenges **C2** and **C3**. On the other hand, text extraction solutions that manage to extract attribute values across multiple product categories [60] are text focused, and the techniques cannot easily be transferred to image information extraction. A comparison between these models are summarized in Table 3.1. In this work, we address the central question: *how can we perform multi-modal product attribute extraction across various product categories?*

Methods	**C1**			**C2**	**C3**
	Text	**Image**	**OCR**	**Domain**	**Category**
BAN[65]	√	√			
LXMERT[124]	√	√			
LoRRA[119]	√	√	√		
OpenTag[148]	√			√	
TXtract[60]	√			√	√
Ours	√	√	√	√	√

Table 3.1: Comparison between Different Methods

3.1.3 Problem Definition



Beauty & Personal Care › Skin Care › Sunscreens & Tanning Products › Sunscreens

Alba Botanica Hawaiian Sunscreen Clear Spray, SPF 50, Coconut, 6 Oz

Visit the Alba Botanica Store

3,907 ratings | 37 answered questions

Amazon's Choice for "alba botanica hawaiian sunscreen clear spray"

List Price: ~~$15.99~~
Price: $7.58 ($1.26 / Fl Oz)
You Save: $8.41 (53%)

Pay ~~$7.58~~ $0.00 after using available American Express Membership Rewards points.

Item Form	Spray
Brand	Alba Botanica
Sun protection index in FPS	50 SPF
Item Volume	9.6 Fluid Ounces
Age Range (Description)	Adult

About this item

- One 6 ounce. bottle of Alba Botanica Coconut Hawaiian Clear Spray SPF 50 Sunscreen
- Lightweight sunscreen made with coconut extract, shea butter and avocado oil
- Provides broad spectrum protection, is water resistant for up to 80 minutes and is coral reef friendly
- Recommended by the Skin Cancer Foundation
- No oxybenzone, octinoxate, gluten or synthetic fragrances

Figure 3.2: Example of product profiles.

A product profile displayed on an e-commerce website usually looks like Figure 3.2. A navigation bar (top in Figure 3.2) describes the category that the product belongs to. Left side is product image and right side are product texts, including a title and several bullet points. We consider products in a set of N product categories $C = \{c_1, \cdots, c_N\}$; a category can be coffee, skincare, as shown in Table 3.2. We formally define the problem as follows:

Problem definition: We take as input a target attribute *attr* and a product with the following information:

1. a phrase describing the product category;
2. the text in the product profile (*i.e.,* title, bullet points), denoted by a sequence of M words $T = \{w_1^{text}, \cdots, w_M^{text}\}$;
3. the product image [1].

[1] we use the first image shown on the website. For certain attributes such as nutrition information, later images such as nutrition label are used instead.

Our goal is to predict the value for the target attribute *attr*. Figure 3.2 displayed a few such attribute values for a sunscreen product. Specifically, considering the target attribute *"Item Form"* shown in Figure 3.2, our objective is to extract the attribute value *"Spray"*. If the target attribute is *"Brand"*, the objective is to extract the attribute value *'Alba Botanica'*.

3.2 Domain-Aware Attribute Extraction Model

3.2.1 Proposed Method

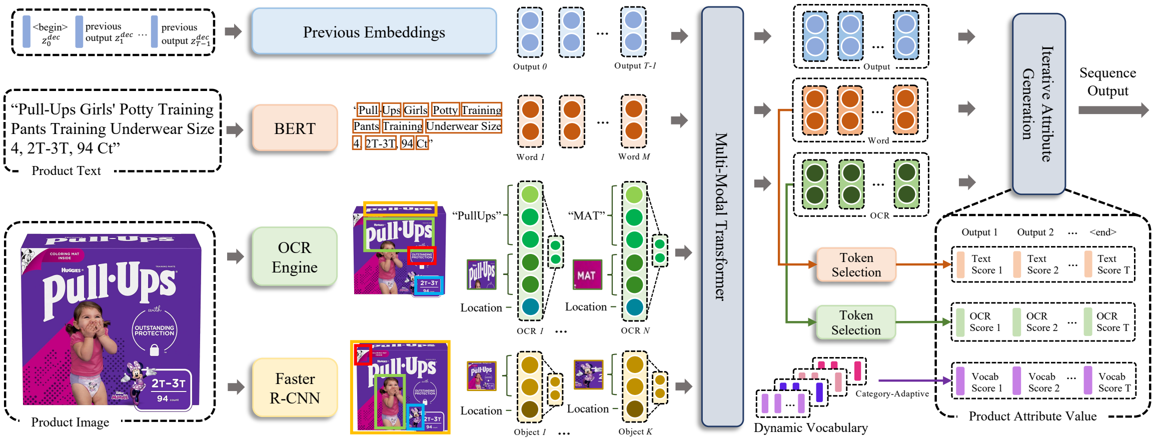


Figure 3.3: Overview of the proposed framework. 1) Input modalities: Product title and product image. 2) Token selection: Tokens could be selected from product title, OCR tokens identified in product image and a dynamic vocabulary conditioned on the product category. We consider edit distance between candidates and existing attribute values while selecting the next token. 3) Target sequence: We ask the decoder to first decode product category, and then decode attribute value.

(a) Overall Architecture. As shown in Figure 3.3, the overall model architecture is a sequence-to-sequence generation model. The encoder and decoder are implemented with one transformer, denoted as "Multi-Model Transformer" in Figure 3.3, where attention masks are used to separate the encoder and decoder computations from each other internally. The input from the different modalities and the previously decoded tokens are each converted into vector representations, the details can be found in next section about input representation. These representations are transformed into vectors of the same dimension, then concatenated as a single sequence of embeddings that is fed to the transformer. Therefore, an input position in each input modality is free to attend to other positions within the same modality, or positions from a different modality. The decoder operates recursively and outputs a vector representation z_t^{dec} at the tth step. The decoder output is based on the intermediate representations of the different encoder layers, along with the embeddings of the previously decoded tokens z_{t-1}^{dec} at step $0 \cdots t-1$, denoted by "Previous Embeddings" in Figure 3.3. A token selection module then chooses the token to output at step t based on z_t^{dec} from a candidate token set.

(b) Input Representation. The product profile texts are fed into the first three layers of a BERT model. The outputs of the pre-processing steps are then converted into three embeddings with the same dimension. The input image is pre-processed with object detection and OCR recognition. For object detection, the product image is fed into the Faster R-CNN object detection model [106], which returns bounding boxes of detected objects, denoted as "Location" in Figure 2.6, and fixed length vector representation extracted through RoI-Pooling among each detected region. The OCR engine provides the detected texts, along with their bounding boxes. We also extract visual features over each OCR token's region using the same Faster R-CNN detector. We experimented with two different OCR solutions: *1)* Public Amazon OCR API. [2] *2)* Mask TextSpotter [77] which is more capable of detecting texts that are organized in unconventional shapes (such as a circle).

(c) Output Selection. The traditional sequence-to-sequence generation model is known to suffer from text degeneration [46], in which the decoder outputs repetitive word sequences that are not well formed linguistically. To fix this problem, the output

[2] https://aws.amazon.com/rekognition

of the decoder in our model is constrained to a set of candidate words. At the tth decoding step, the decoder can return a token from the product text profile, the OCR engine, or from an external pre-defined vocabulary. The external vocabulary is composed of the words from the set of target attribute values, obtained from the training dataset. It is useful in the cases where *1)* the true attribute value is not mentioned in the product profile or image, *2)* the words in the image are not properly captured by the OCR algorithm, or *3)* the true attribute value is implied by the different inputs (from the product profile and images), but not explicitly mentioned. An example for the third case is predicting the target age of a "hair clips for young girl" product, where the target values include "kid", "teenage", or "adult" only.

Dynamic vocabulary. The vocabulary of target attribute values could contain very different words for different product categories. For example, it is unlikely for sunscreen brands to share common words with coffee brands except for words *"company"* or *"inc"*. Hence for each (product category, attribute type) pair, we pre-identify a specific vocabulary of frequent attribute value words, denoted by $V_{i,j}$ for the ith product category and jth attribute type. We also added the words that appear in the product category name to the vocabulary. In addition, the word *"unknown"* is added to the vocabulary, so the model can output *"unknown"* when the input product profile does not contain a value for this attribute. The goal is to capture patterns for products where the attribute value is not usually conveyed. During the training process, the model will query the vocabulary $V_{i,j}$ according to the input product category i, which provides a more precise prior knowledge. The vector representation for each word in $V_{i,j}$ is obtained with pre-trained fastText [10] embedding. This is because there are not enough training data in some product categories to compute a good word representation during the learning process.

Scoring function. The lack of domain-specific embedding hurts the accuracy of the output token selection module. We therefore utilize the edit distance between the candidates and existing attribute values as a supplemental feature. We denote the edit distance based similarity ratio [3] for specific word token w compared with the vocabulary $V_{i,j} = \{v_1, \cdots, v_L\}$ with L words as:

$$f^e(w) = \max_{l \in L} \text{ similarity\_ratio}(w, v_l) \tag{3.1}$$

With the decoded embedding z_t^{dec} and the edit distance function f^e, we calculate the final score for candidates from the different modalities as follows, where (3.2), (3.3), and (3.4) are for tokens in the dynamic vocabulary $V_{i,j}$, OCR tokens, and tokens from product profile texts, respectively.

$$y_{t,l}^{voc} = (W^{voc} z_l^{voc} + b^{voc})^T (W^{dec} z_t^{dec} + b^{dec}) \tag{3.2}$$

$$y_{t,n}^{ocr} = (W^{ocr} z_n^{ocr} + b^{ocr})^T (W^{dec} z_t^{dec} + b^{dec}) + \lambda f^e(w_n^{ocr}) \tag{3.3}$$

$$y_{t,m}^{text} = (W^{text} z_m^{text} + b^{text})^T (W^{dec} z_t^{dec} + b^{dec}) + \lambda f^e(w_m^{text}) \tag{3.4}$$

If d denotes the dimension of the encoder's output embedding, then W^{text}, W^{ocr} and W^{dec} are $d \times d$ projection matrices, W^{voc} is a $d \times 300$ matrix (300 is the dimension of the fastText embeddings). z_m^{text}, z_n^{ocr} are the output embeddings for text tokens and OCR tokens computed by the encoder, respectively. z_l^{voc} is the fastText embedding of the frequent vocabulary words. b^{text}, b^{ocr}, b^{voc}, b^{dec} are all d-dimensional bias vectors and λ is the hyper-parameter to balance the score. Finally, the auto-regressive decoder will choose the candidate tokens with the highest score from the concatenated list $[y_{t,m}^{text},\ y_{t,n}^{ocr},\ y_{t,l}^{voc}]$ at each time step t.

(d) Multi-Task Training. We use the multi-task learning setup to incorporate the product categories in the overall model. We experiment with two methods of multi-task training.

[3] The FuzzyWuzzy ratio implemented in https://github.com/seatgeek/fuzzywuzzy

Embed the Product Category in Target Sequence. The first multi-task training method is based on prefixing the target sequence the decoder is expected to generate during training with the product category name. For example, the target sequence of product shown in Figure 3.2 would be *"sunscreen spray"* for attribute *"Item Form"* and *"sunscreen alba botanica"* for attribute *"Brand"*. The category name prefix serves as an auxiliary task that encourages the model to learn the correlation between the product category and attribute value. At inference time, the decoder output at the tth step depends on its previous outputs. But since the ground truth value of the product category is known, there is no need to depend on product category estimated by the decoder. We simply replace it with the true product category value. We have seen empirically that this modification improves the precision of the model. Let the target label during the tth decoding step be $[y_{t,m}^{text},\ y_{t,n}^{ocr},\ y_{t,l}^{voc}]$, which takes the value 1 if the token from text, OCR, or external vocabulary is the correct token and 0 otherwise. More than one token could have label 1 if they are identical but from different sources. We use the multi label cross entropy loss between the target label list $[y_{t,m}^{text},\ y_{t,n}^{ocr},\ y_{t,l}^{voc}]$ and the predicted score list $[\hat{y}_{t,m}^{text},\ \hat{y}_{t,n}^{ocr},\ \hat{y}_{t,l}^{voc}]$ given by (3.2)-(3.4). The loss function hence contains two term: the loss contributed by the category name prefix, and the loss contributed by the attribute value in the target sequence:

$$Loss = Loss_{\text{attribute value}} + \lambda_{cat} Loss_{\text{category name prefix}} \tag{3.5}$$

where λ_{cat} is the tunable hyper-parameter to balance between these two losses.

Auxiliary Task of Category Prediction. The target sequence method is by no means the only possible design of multi-task training. It is also possible to introduce a separate classifier $f^{cat}(z)$ to predict the product category. A specific classification token `<CLS>` is inserted as the first entity in the input to the encoder.

Category	# Samples	# Attr1	# Attr2
cereal	3056	7	631
dishwasher detergent	741	8	114
face shaping makeup	6077	16	926
fish	2517	11	391
herb	6592	19	1220
honey	1526	20	472
insect repellent	994	20	373
jerky	3482	9	475
sauce	4218	10	878
skin cleaning agent	10904	22	3016
skin foundation concealer	8564	17	744
sugar	1438	10	347
sunscreen	5480	26	1295
tea	5719	14	1204

Table 3.2: Dataset Statistics

After concatenating and fusing with the other multimodal contexts in the transformer encoding layer, the enriched representation z^{cls} corresponding to the classification token `<CLS>` will be passed to the feed-forward neural network $f^{cat}(z)$ to predict the product category.

$$f^{cat}(z) = softmax(W^{cat} z^{cls} + b^{cat}) \tag{3.6}$$

where W^{cat} and b^{cat} are trainable parameters.

The training of the end-to-end model is jointly supervised by the sequence generation task and product category prediction tasks as described in (3.7).

$$Loss = Loss_{\text{attribute value}} + \lambda_{cat} Loss_{cat} \tag{3.7}$$

where $Loss_{cat}$ is the loss for product category prediction task.

3.2.2 Experiments Results

(a) Dataset. We evaluate our approach on 61,308 samples that cover 14 product categories. For each product category, we randomly collect the product texts, attribute values and images from the `amazon.com` web pages. We split the dataset into 56,843 samples as training/validation set and 4,465 samples as held-out testing set. The attribute values shown on the web page are used as training label after basic pre processing, which handle the symbol and morphology issues. The attribute values labeled by annotators are used as benchmark testing label. Assuming the attribute type is applicable to the products, if the attribute value information can not be observed from the given product profile (text description, image, OCR tokens), we will assign *"unknown"* as the corresponding value. In terms of the target attribute, we focus on two different types of attribute in the experiments. One of the criteria to determine the attribute type is the size of the value space. We consider attribute 1 *"Item Form"* with around 20 values as an attribute with closed vocabulary, and attribute 2 *"Brand"* with more than 100 values as an attribute with open vocabulary. Table 3.2 summarizes the statistics of our dataset, where "# Samples" denotes the number of samples, "# Attr1" denotes the number of unique values for attribute 1 *"Item Form"* and "# Attr2" denotes the number of unique values for attribute 2 *"Brand"*.

(b) Evaluation Metrics. We use *Precision*, *Recall* and *F1* score as the evaluation metrics. We compute *Precision* (denoted as P) as percentage of "match" value generated by our framework; *Recall* (denoted as R) as percentage of ground truth value retrieved by our framework; *F1* score (denoted as $F1$) as harmonic mean of *Precision* and *Recall*. We determine whether the extraction result is a "match" using the exact match criteria, in which the full sequence of words are required to be correct.

(c) Baselines. To evaluate our proposed framework, we choose the following models as baselines: BiLSTM-CRF [53], OpenTag [148], BUTD [3] and M4C [49]. Our attribute value extraction task is highly related to the visual question answering tasks. Thus, among the four baselines, BiLSTM-CRF and OpenTag are attribute value extraction models, BUTD and M4C are visual question answering models. Some variants of our model and baselines are also included to make fair comparison on input sources. The details of baselines are listed below:

- BiLSTM-CRF [53]: the hidden states generated by the BiLSTM model are fed into the CRF as input features, the CRF will capture dependency between output tags. Text modality is used in this model.

- OpenTag [148]: on top of the BiLSTM-CRF, attention mechanism is introduced to highlight important information. Text modality is used in this model.

- BUTD [3]: Bottom-Up and Top-Down (BUTD) attention encodes question with GRU [20], then attends to object region of interest (ROI) features to predict answer. Text and Image modalities are used in this model.

- M4C [49]: cross-modality relationships are captured using multimodal transformer, the model will then generate the answer by iterative sequence decoding. Text, image and OCR modalities are used in this model. The answer could be selected from OCR tokens and frequent vocabulary.

- M4C full: to accommodate to the attribute value extraction task in e-commerce applications, extra input source of product title are added directly in the decoding process. Text, image and OCR modalities are used in this model. The answer could be selected from product title, OCR tokens and frequent vocabulary.

- PAM text-only: the text-only variant of our framework, image visual features and OCR tokens extracted from the product image are excluded from the input embeddings. Text modality is used in this model.

(d) Comparison between Model Architectures. We first show the performance comparisons between our approach, baselines and some variants on two different attributes *"Item Form"* and *"Brand"* in Table 3.3. As can be seen from these comparison results, PAM could consistently outperform the other baseline methods on Recall and F1 score. For example, for the *"Item Form"* attribute, the Recall of PAM increases by 15% compared with the text-only variant of PAM and increases by 22% compared with the M4C model. For the *"Brand"* attribute, the Recall of PAM increases by 5% compared with the text-only variant of PAM and increases by 15% compared with the M4C model. Note that PAM could achieve higher score on all metrics compared with the M4C full variant (full access to all modalities in the decoding process), which demonstrate the effectiveness of our task-specific designs on the framework. There are two main reasons contribute to these improvements: *1)* PAM utilizes rich information from naturally fused text, image and OCR modalities that could significantly improve Recall. These three modalities could help each other by providing important cues while information might be missing in specific modality. *2)* PAM utilizes product category inputs in the decoding process. Attribute value is highly related to product category. By considering such crucial information, our model is able to learn enriched embeddings that could discriminate targeted attribute values from distracting values that belong to other product categories.

(e) Ablation Study. In order to quantify the impact of each modality, we further conduct ablation study on the input sources. We evaluate following variants of PAM:

- PAM *w/o* text is the variant that removes product texts modality from inputs.
- PAM *w/o* image is the variant where features of detected objects are removed from inputs.
- PAM *w/o* OCR is the variant that removes the OCR tokens from inputs.

From Table 3.4 we can see that all the metrics on the attribute *'Item Form'* degrade by removing any modality from the PAM framework, which demonstrates the necessity

Attributes	**Models**	***P(%)***	***R(%)***	***F1(%)***
Item Form	BiLSTM-CRF	90.8	60.2	72.3
	OpenTag	95.5	59.8	73.5
	BUTD	83.3	53.7	65.3
	M4C	89.4	52.6	66.2
	M4C full	90.9	63.4	74.6
	PAM (ours) text-only	94.5	60.1	73.4
	PAM (ours)	91.3	75.3	82.5
Brand	BiLSTM-CRF	81.8	71.0	76.1
	OpenTag	82.3	72.9	77.3
	BUTD	79.7	62.6	70.1
	M4C	72.0	67.8	69.8
	M4C full	83.1	74.5	78.6
	PAM (ours) text-only	81.2	78.4	79.8
	PAM (ours)	86.6	83.5	85.1

Table 3.3: Comparison between proposed framework PAM and baselines

of combining all the modalities in our attribute extraction task. Closer inspection on the table shows that the text modality plays the most important role in this task. On the other hand, the image modality which represents the appearance of the product might be less effective compared to the other two modalities. The first possible reason is that the image could contain noisy information. In addition, similar shape of product might have different semantic meanings among various product categories. Finally, different attribute types also affect the performance, image modality could contributes more if the attribute type is related to color or obvious shape.

We also conduct experiments by removing each individual model design component from our framework to evaluate its effectiveness. The variants are listed below:

- PAM *w/o* target sequence is the variant that will generate attribute value without first generating the product category name.

- PAM *w/o* dynamic vocabulary is the variant that uses a large vocabulary of words shared by multiple categories instead of a dynamic vocabulary conditioned on product category.

Models	***P(%)***	***R(%)***	***F1(%)***
PAM *w/o* text	79.9	63.4	70.7
PAM *w/o* image	88.7	72.1	79.5
PAM *w/o* OCR	82.0	69.4	75.1
PAM	91.3	75.3	82.5

Table 3.4: Usefulness of Image, text, OCR inputs

Models	***P(%)***	***R(%)***	***F1(%)***
PAM *w/o* target sequence	88.5	72.9	80.0
PAM *w/o* dynamic vocabulary	89.1	69.5	78.1
PAM	91.3	75.3	82.5

Table 3.5: Impact of different components in the model

Table 3.5 presents the extraction results on the *"Item Form"* attribute. Without the category specific vocabulary set, the model has to search on a much larger space for possible attribute values. The target sequence could enforce the category information via back-propagation loss. It is apparent from the results that these two category modules contribute to the final gains on Recall/F1 score.

Our approach is able to accommodate various product categories in one model. In order to verify such generalization ability on single category, we perform the category-level individual training tasks using the following baselines:

- OpenTag[148]: the setting is the same as described in Section 3.2.2 (c), except the training and evaluation are performed on single product category.

- Word Matching (WM): this is a brute-force matching baseline. *1)* all the possible attribute values will be collected as attribute value dictionary ({*"value"* : *count*}), the count represents the popularity of corresponding attribute value; *2)* manually exclude some distracting words like "tea" from the dictionary of the "tea" product category; *3)* extract title tokens and OCR tokens for each sample; *4)* compare the extracted tokens with attribute value dictionary in a popularity ascending sequence; *5)* identify the attribute value if exact match is

OCR Detectors	**average # OCR tokens extracted**	***F1(%)***
Mask TextSpotter	13	71.1
Amazon Rekognition	42	80.3

Table 3.6: Impact of OCR component on model performance

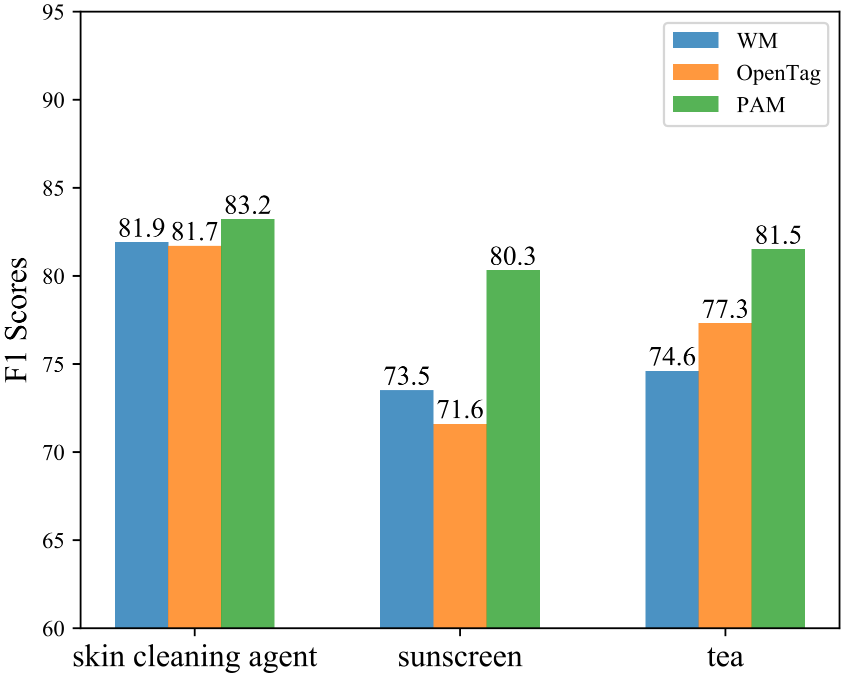


Figure 3.4: Comparison between different methods on a single product category.

found in the attribute value dictionary. This baseline method does not require training, it is evaluated on the testing data directly.

For the purpose of performing category-level individual training, we choose three categories that contain enough number of samples: *skin cleaning agent*, *sunscreen* and *tea*. Figure 3.4 demonstrates the comparison of two baselines and our model on single category. Our method consistently improves the extraction metric. Although the WM baseline could also produce a good F1 score for *"skin cleaning agent"*, it requires manual efforts to create a good list to exclude words that are impossible to appear in attribute values, which is expensive to scale up to many product categories.

Under the single category settings, we also implement experiments to evaluate the impact of the external OCR components on end-to-end performance. As introduced in Section 3.2.1, we use Amazon Rekognition and Mask TextSpotter to extract OCR tokens from the product image. $F1(\%)$ is the extraction performance on attribute *Item Form* and category *Sunscreen* of using corresponding detectors. It can be seen from Table 3.6 that Rekognition is more suitable for our task. This is because Mask TextSpotter is trained on a public dataset that is different from the product image dataset. Therefore, Rekognition on average identifies more OCR tokens and hence lead to better end-to-end $F1$ scores.

3.2.3 Conclusions

To sum up, we explored a multimodal learning task that involves textual, visual and image text collected from product profiles. We presented a unified framework for the multimodal attribute value extraction task in the e-commerce domain. Multimodal transformer based encoder and decoder are used in the framework. The model is trained to simultaneously predict product category and attribute value and its output vocabulary is conditioned on the product category as well, resulting in a model capable of extracting attributes across different product categories. Extensive experiments are implemented on a multi categories/multi attributes dataset collected from public web page. The experimental results demonstrate both the rich information contained within the image/OCR modality and the effectiveness of our product category aware multimodal framework.

For future works, pre-training task from [142] could be useful in our attribute value extraction scenario. It is also valuable to scale from 14 product categories to thousands of product categories and model the complex tree structure of product categories properly [60]. The dynamic selection of vocabulary in this framework could be incorporated into the training process as in the RAG architecture [69]. Finally, it is useful to design a model that extracts different attributes with one model in which case the attribute generation order could be part of the learning process too [28].

Chapter 4

Multi-modal Learning with Pre-training Tasks on Healthcare Data

4.1 Introduction

Deep neural networks have made tremendous success in solving increasingly complex real-life problems, which often involve different modalities. Due to the powerful representation learning ability and high performance computing resources, deep learning-based multimodal representation learning has attracted much attention. Such learning method needs to be able to integrate and fuse multimodal signals together and narrow the heterogeneity gap among various modalities.

In recent years, with the advancement of the efficient monitoring wearable devices and the adoptions of electronic health record (EHR) in hospitals, there has been an increased focus on developing multimodal representation learning for processing the medical signals. Comprehensively utilizing these rich and diverse signals will benefit the medical services in many ways, application scenarios in the healthcare system include real-time monitoring and surveillance on specific symptom, early detection for certain disease and providing timely suggestions on necessary medication and measurement.

Despite the potentials, multimodal representation learning on medical signals presents several challenges. Symptoms and diseases are usually complicated and multi-factorial. Multiple overlapping diseases might affect different signals simultaneously. Thus, fusing information from multimodal signals is especially crucial for healthcare applications due to the underlying causal correlations between EHR data, sensor signals and other structured data, including the vital signals, laboratory values, demographic information, medications and clinical notes. In addition, such multimodal signals conveys different aspects of human physiology and might have different features, format and frequency. For example, the wearable electronic sensors provides high-frequency measurement on user's heart rate, while some important lab results related to serum creatinine are measured on much lower frequency since it requires blood samples from patients. Appropriate representation tailored for each single modality are required to accurately combine multimodal information. The final challenge is the alignment between multiple modalities. It is very usual that specific modality has worse resources compares to others. Common issues include unreliable labels, sparse and noisy input, and lack of annotated data. By exploiting and learning knowledge from other modalities belong to same instance during the training process, it is possible to bridge modalities and improve overall performance.

Most approaches on multimodal fusion for medical signals can be classified into two categories: *(1) early fusion.* Signals from different modalities are pre-processed and concatenated in the early phrase. Features are extracted from such combined signals and feed into the downstream task like classification. This method requires innovations in sensor synchronization, buffering, denoising and data normalization. *(2) late fusion.* Raw signals from each sensors are featurized separately and then fused for downstream task. Such fusion method requires feature selection and feature normalization to handle different time spans and signal scales. In addition, these separate features can be fused in different ways, such as naive concatenating before classifier, adding extra classifier for each modality and applying majority voting.

In this work, we propose a new framework that can intelligently fuses multi-sensor and multi-source data in an attention-based halfway manner. Specifically, we investigate the transformer encoder-decoder network with attention mechanisms for more powerful intra-modality modeling. Features are first extracted from raw sensor data using corresponding model (e.g. Gated Recurrent Unit (GRU) [19] for temporal signals) and projected into a common embedding space. The list of projected features is fused halfway using a stack of multi-modal transformer layers. Unlike existing works on multi-modal fusion, through the multi-head self-attention mechanism in our transformer layers, each entity is able to freely attend to all other entities, regardless of whether they are from the same sensor or not. This property enables modeling both inter- and intra- modality relations in a homogeneous way through the same set of transformer parameters.

4.2 Proposed Method

4.2.1 Problem Formulation

We formulate the proposed framework with clinical multi-sensor application in the multimodal representation learning settings. Assuming we have temporal sensor data in our data cohorts and data instances have medical signals with various time length. A maximum length T is pre-defined for all modalities. Instances with shorter length will get zero-pad up to the maximum length T. Corresponding data mask $M_{pad} = \{m_0, m_1, \cdots, m_T\}$ is set to 1 for timestamps with real values and 0 for timestamps with paddings. We initially have multiple raw data with time length T from N different sensors: $X^0 = \{x^0_0, x^0_1, \cdots, x^0_T\}$, $X^1 = \{x^1_0, x^1_1, \cdots, x^1_T\}$ and

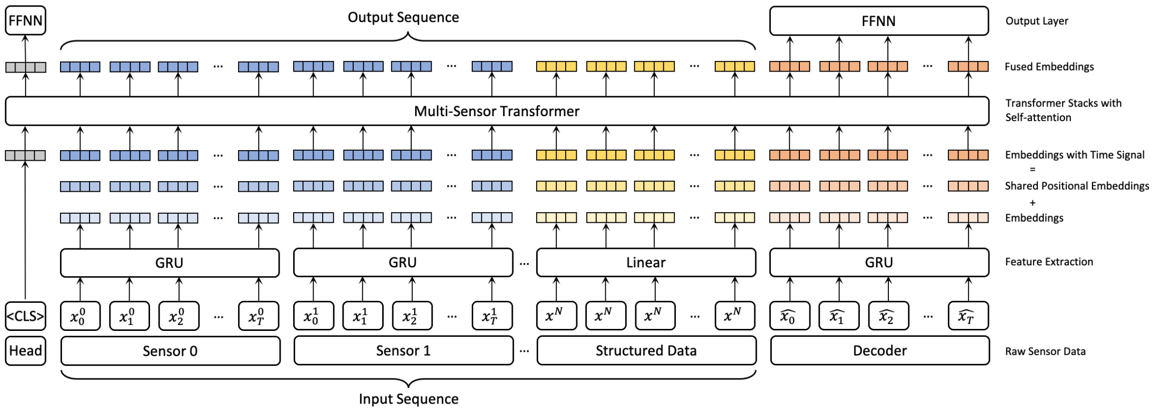


Figure 4.1: An overview of the Multi-Sensor Fusion Framework

$X^N = \{x_0^N, x_1^N, \cdots, x_T^N\}$.

We consider the following two types of clinical prediction applications. The prediction here refers to classification task or real-value regression task. In this work, we mainly focus on classification task. *(1) one-time classification.* Given the entire multi-modal trajectory of instance, the model is required to make the one-time final prediction. We consider 1 label y for each instance. Such label could be $[positive, negative]$ as the event indicator for certain disease / symptom. *(2) real-time classification.* To fulfill the needs on applications like real-time monitoring and healthcare surveillance. The model should be able to make real-time prediction given consecutive window (window size varies from 1 to T) of multi-sensor signals. For each instance, we consider T labels $Y = \{y_0, y_1, \cdots, y_T\}$ corresponds to event on each timestamp. We further consider a prediction horizon with time interval τ. The label at time t is set to positive if disease / symptom occurs within the next time interval τ and set to negative otherwise.

4.2.2 Multi-Sensor Fusion Framework

Most existing fusion methods simply concatenate features in the early fusion manner or generate ensemble prediction results in the late fusion manner. We propose a more flexible fusion framework for healthcare applications with multisensor data. Figure 4.1 shows the overview of our framework. As we mentioned earlier, there are three major challenges while ingesting the multi-sensor multi-source clinical EHR data: *representation*, *fusion* and *alignment*.

In order to get appropriate *representation* from the multi-sensor raw data, we construct the embeddings in the following two steps: *(1) Construct input sequence.* To capture the underlying temporal pattern for each modality that contains timeseries data, we use the GRU model to extract embeddings with hidden size d from the raw data collected by sensors. On the other hand, the structured data are projected into the same d-dimensional space using learned linear layers. These extracted multi-sensor features are grouped in modality and concatenated as the "input sequence" we denoted in Figure 4.1 *(2) Add various heads.* The first steps could also be viewed as the construction of extraction head. To incorporate our downstream task, in addition to the "input sequence", we add the classification head at the begin and decoder head at the end. The classification head contains the `<CLS>` token, the final embedding correspond to this position can be used in classification task. The decoder head consumes concatenated information from all modalities. The rich information is denoted as $\hat{x_t} = [x_t^0, x_t^1, \cdots, x_t^N]$. The GRU model is again used here to extract embeddings for entities in decoder.

To handle the second challenge *fusion*, we utilize the Multi-Sensor Transformer as the attention-based fusion module. For our setting that contains N modalities and each modality has maximum sequence T, the Multi-Sensor Transformer is a stack of L transformer layers [126] with hidden size d and input length $1+T*(N+1)$. In this way, all entities will be fused halfway in the self-attention layers. By adopting appropriate masking pattern, such attention mechanism provides a much flexible inter- and intra-fusion. Specifically, before feeding the embeddings into the transformer stacks, we apply a shared positional embedding on each modality and the decoder with T as embedding number and d as embedding dimension. The positional embedding is important as it provides the time signal for each entity in our framework. Layer normalization (LN) [6] is added to ensure the same scale of original and positional embeddings. To enhance the *alignment* among multiple sensors, we propose several Pre-training tasks illustrated in Section 4.3.

Depends on the downstream task, our framework has two variants: one-time classification and real-time classification. The overall settings are similar. For the one-time variant, the downstream classification task will take the pooled output corresponds to the `<CLS>` token as the input feature. We use feed forward neural network (FFNN) and binary cross entropy (BCE) loss to perform the classification task. While the real-time variant will use fused embedding generated by decoder head as the input feature, and perform classification task on each timestamp. Another important difference between these two variants is the masking pattern, which determines the attention-based fusion mechanism in the Multi-Sensor Transformer. The details are illustrated in Section 4.2.3.

4.2.3 Fully-Visible vs. Causal-Prefix Masking Pattern

By comparing the one-time classification and real-time classification variants, the "mask" applied in the self-attention mechanisms is the major distinguishing factor in our framework. As illustrated in [126], the self-attention in our multi-sensor transformer can be described as mapping a query and a set of key-value pairs to an output. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. The entire process takes a sequence as input and outputs a new sequence of the same length. We use our notation introduced in Section 4.2.1 to denote the input entities and z_t to denote the output entities. In addition to our data mask M_{pad} for unified sequence length, the attention mask M_{att} built upon it is used to zero out certain weights assigned on corresponding positions to constraint the interactions with other entities in the input.

Most of the encoder architecture considers Fully-Visible mask, which allows every entity to attend over all positions in the input. While making a one-time prediction, our model and especially the `<CLS>` token appended before the input sequence should have the access to all the information within the time interval. Thus, the Fully-Visible masking pattern is applied in our one-time classification setting.

In terms of the decoder architecture, when producing the entities of the output sequence, the causal masking pattern is adopted in the training process to prevent the information leakage from the future. Specifically, causal masking pattern generates triangle mask $m_{input\_index,output\_index}$ where the mask is set to zero if $input\_index > output\_index$. Recall that our encoder and decoder are concatenated and share the same multi-sensor transformer layers, so we need to further combine the prefix masking pattern proposed in [103] to ensure causality in decoding. To summarize, in our real-time classification setting, we apply the Causal-Prefix masking pattern. As shown in Figure 4.2, we take input entities with two modalities as the example to illustrate the attention mask. For inter- and intra- modality attention, we apply the causal mask on blocks to ensure the outputs on timestamp t cannot attend to any entity in future timestamps. For encoder and decoder separation, we apply the prefix mask to ensure the multimodal signals cannot attend to any decoding steps, and the decoding steps at timestamps t can only attend to previous decoding steps in addition to previous multimodal signals. The visibility in the self-attention layer among input entities is further visualized in Figure 4.3.

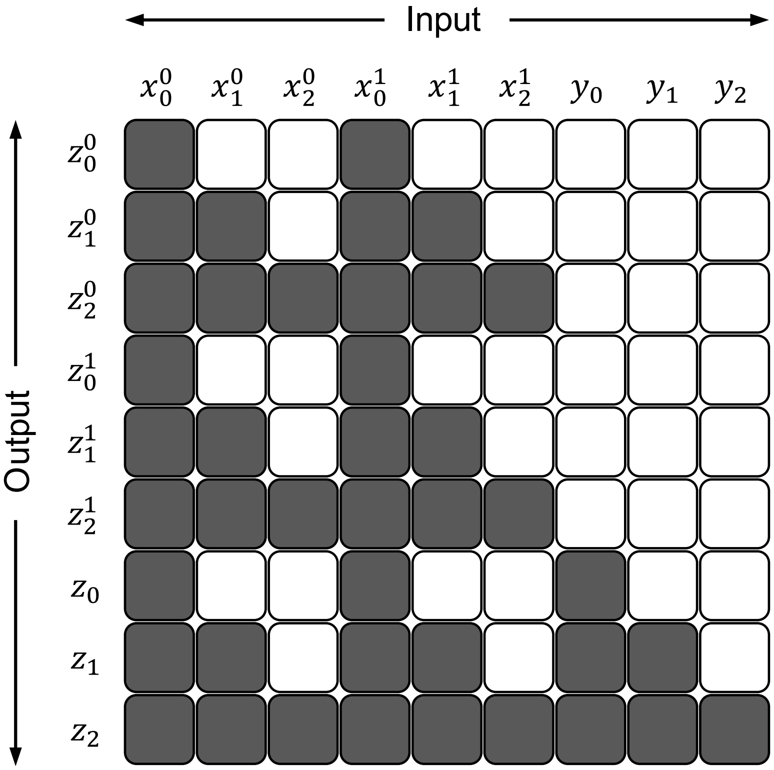


Figure 4.2: Causal-Prefix masking pattern. Black cell represents 1 in the mask, $m_{0,0} = 1$ means the attention mechanism is allowed to attend to the input entity x_0^0 while producing the output entity z_0^0. White cell represents 0 in the mask, $m_{0,1} = 0$ indicates the attention is not allowed to attend to the future input entity x_1^0 for z_0^0.

4.3 Multimodal Pre-training Tasks

4.3.1 Masked Imputation on Each Modality (MIM)

Missing and sparse values are very common in the clinical data due to intentional and unintentional reasons. Such missingness could result in biased prediction and quality degradation on the model. To handling missing data, the simple approach sample-and-hold is applied on the data cohort as the common pre-processing technique. We aim to recover the missing information through the first Pre-training task Masked Imputation on Each Modality (MIM). We randomly mask each entity in the input sequence before feed into the multi-sensor transformer with a probability of 15%. Specifically, the masked entities are replaced with pre-defined `<MASK>` tokens, other

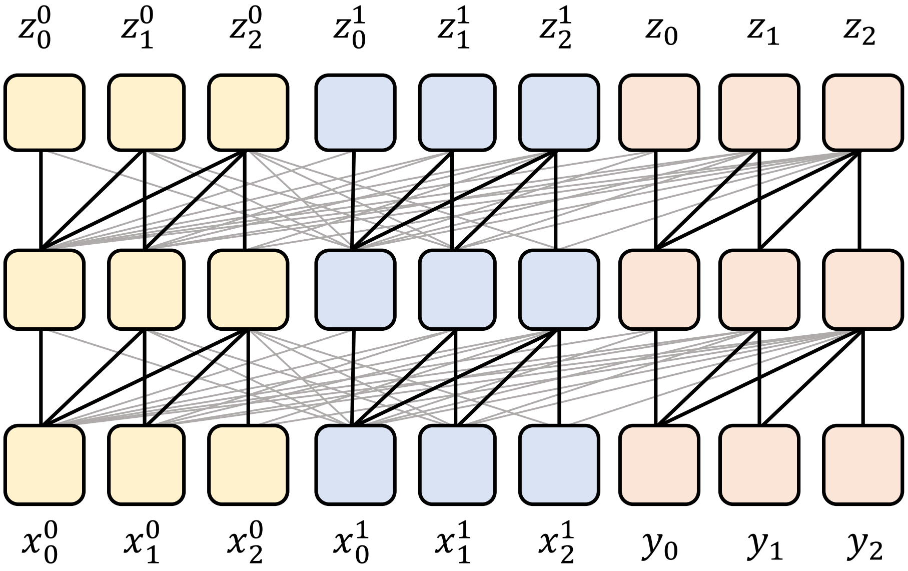


Figure 4.3: Visibility in the self-attention layer among input entities. The cells grouped with same color represents same modality, while the last group represents the decoding part. Dark lines correspond to intra modality visibility and grey lines correspond to inter modality visibility.

random entities and remain unchanged for the probability of 80%, 10% and 10% respectively. The MIM takes the fused feature after multi-sensor transformer at these masked positions as the input, and aims to recover the original value with continuous regression task using two fully-connected layers and mean squared error.

4.3.2 Contrastive Matching through Modality Replacement (MMR)

In order to build connections among multiple modalities in a group-wise manner, we propose the second Pre-training task Contrastive Matching through Modality Replacement (MMR). MMR first randomly select one target modality for each instance, with the probability of 50%, the entire modality is replaced with corresponding modality from a randomly-selected instance in the training dataset. The polluted modality is thus not paired with the rest modalities. The MMR takes the entire output sequence from multi-sensor transformer as the input features and aims to predict if the

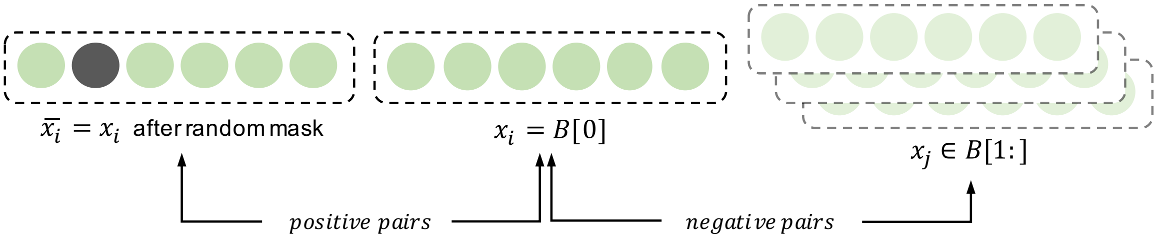


Figure 4.4: Illustration of the data augmentation in Pre-training task MDA

sequence has been polluted or not with binary classification task.

4.3.3 Unsupervised Matching through Data Augmentation (MDA)

Inspired by the contrastive learning framework [16], we design the third Pre-training task Unsupervised Matching through Data Augmentation (MDA) to group similar instances. The basic idea of contrastive learning is to pulling together positive pairs and pushing away negative pairs. For each randomly sampled mini-batch B with batch size N during the training phase. As shown in Figure 4.4, we take the first instance $x_i = B[0]$ as our base sample. The rest instances within this batch $x_j \in B[1:]$ are considered as negative samples compares to base sample. We duplicate the base sample as the extra positive sample $\bar{x}_i$ that can be appended to the batch later. In order to further construct the *positive* and *negative* pairs in an unsupervised manner. We add two kinds of data augmentation as follows.

Input-level Masking. The model should be robust to the sparse and noisy clinical data. Assuming there are errors, missing values on parts of the temporal data, the extracted underlying temporal patterns should be consistent. We simulate this scenario by applying `<MASK>` tokens on the extracted temporal feature before feeding it into the multi-sensor transformer. With the probability of 10%, we randomly replace entities with `<MASK>` tokens among the input sequence of positive samples $\bar{x}_i$.

Feature-level Dropout. We append the positive sample $\bar{x}_i$ to the current mini-batch during the training process. Through the dropout modules in the multi-sensor transformer, even the identical instance would generate augmented views of input data. The representation would be slightly different due to the default dropout layers. We leverage this property and encourage the positive pairs $f(x_i), f(\bar{x}_i)$ to obtain similar embeddings compares to the negative pairs $f(x_i), f(x_j)$ by applying the following cross-entropy objective [15, 42, 30] for N pairs within this mini-batch.

$$\mathcal{L} = -log \frac{exp(sim(f(x_i), f(\bar{x}_i))/\tau)}{\sum_{j=1}^{N} exp(sim(f(x_i), f(x_j))/\tau)} \tag{4.1}$$

where sim(. , .) denotes the cosine similarity and τ denotes the temperature parameter.

4.4 Applications and Experiments

4.4.1 Clinical Applications and Data Cohorts

We evaluate our framework on two challenging clinical datasets. Both datasets contain multi-source EHR timeseries data and structured data. The pre-processing steps are similar. The datasets are standardized (subtracted by mean and divided by standard deviation, both measured on training set) to specified granularity and represented as numerical timeseries. Structured data are duplicated and appended to each hour of the series. For handling missing data, sample-and-hold approach is used in all the datasets. We consider specific applications for each dataset. Real-time and one-time classification tasks are incorporated accordingly.

Physionet Sepsis Dataset. Sepsis is a potentially life-threatening condition that occurs when the body's response to an infection damages its own tissues. We use the public "Early Prediction of Sepsis from Clinical Data" [107] on PhysioNet [32] as the first dataset. The Physionet Sepsis dataset is obtained from three geographically distinct U.S. hospital systems with three different electronic medical record systems: Beth Israel Deaconess Medical Center, Emory University Hospital and unidentified hospital system. The public dataset contains 40,336 Patients with hourly sepsis label determined by Sepsis-3 clinical criteria [114, 118, 115]. In particular, the data contained 40 clinical variables: 8 vital sign variables, 26 laboratory variables, and 6 demographic variables. Altogether, these data included over 2.5 million hourly time windows and 15 million data points. Specifically, we have the following details related to our framework:

- **Statistics:** Sepsis prevalence is 7.26%. The patients have minimum stay of 8 hours and maximum stay of 336 hours. 85% of patients have length of stay less than 50 hours. Thus, we set the maximum length $T = 50$ and truncate the data with the latest 50 hours.
- **Modalities:** We consider the 8 vital signs as separate modality, the laboratory variables are grouped as one modality, and finally the structured demographic serves as the last modality.
- **Task:** We aim to predict the binary Sepsis label at each timestamp for the patients. The real-time classification variant of our framework is adopted in this dataset.

Girasoles Sensor Dataset. The health effects of environmental heat exposure, the most significant cause of weather-related mortality in the U.S. [91], are critical public health issues. Heat-related illness (HRI) is entirely preventable and is also treatable provided it is detected in a timely manner. Heat exposure invokes multiple modes of physiological response, and thus multiple sensors are necessary to better understand these acute health effects. In addition, a growing body of evidence indicates that repeated instances of these health effects resulting in acute kidney injury (AKI) may lead to longer term effects on kidney function. Emory University has launched pilot study examining the physiological response to heat exposure in agricultural workers using multiple sensors. The Girasoles Sensor dataset captured 4,000 hours of core body temperature (using the CorTemp pill and receiver), heart rate (using the Polar belt sensor), motion activity (using the Actigraph), and occurrence of AKI and dehydration among 254 agricultural workers. The AKI is measured by serum creatinine in blood sample and urine sample using KDIGO guideline [64]. The severe dehydration is measured by the urine sample if urine specific gravity (USG) is greater than 1.030. The related details are listed below:

- **Statistics:** AKI ratio is 9.71%. severe dehydration ratio is 9.84%. Due to the limited number of agriculture workers in the dataset and the high frequency property (per minute) of each worker's data. After splitting workers for train/test purpose, we randomly sampled consecutive windows of 50 minutes duration and generated 21,361 training data and 5,371 testing data. The maximum length T is set to 50.

- **Modalities:** We consider the "core temperature", "heart rate", "vector magnitude" and "steps" as 4 separate modality with timeseries data. The demographics is also considered as separate modality with structured data.

- **Tasks:** We aim to perform the one-time classification to predict the occurrence of AKI and dehydration (denoted as USG). In addition, in order to announce the high temperature warning in a timely manner, we design an extra task to predict whether the individual's core body temperature will exceed $38.0°C$ in the next 10 minutes. The real-time classification is applied in the last task.

4.4.2 Experiment Settings

Baseline Models. We evaluate the performance of our framework by comparing against two baseline models: *(1) early fusion baseline,* multi-sensor data are concatenated together and features are then extracted by the GRU model. *(2) late fusion baseline,* features from each sensor are extracted by exclusive GRU model separately and then concatenated before feed into classifier. These two baseline adopt the FFNN+softmax module used in our framework as classifier. Depends on the type of downstream task, the final hidden state or the packed sequence output features from last layer of the GRU will be used accordingly.

Model Architecture. We use unidirectional GRU to avoid leaking information from future timestamps. The hidden size of GRU and the linear layer in the feature extraction phase is determined by the embedding space of Multi-Sensor Transformer. We use $L = 4$ stacks of transformer layers. We explore the BERT miniatures of Mini, Small, Medium and Base. Since the gains on larger model is very limited, we decide to follow the transformer settings as in BERT-Mini. Specifically, we set $d = 256$ as the dimensionality of the joint embedding space, the transformer has 4 attention heads.

Configurations. We use batch size of 64 during training. The model is trained using the Adam optimizer with initial learning rate $1e-4$. Since we have class imbalance issues in our dataset. We use accuracy, macro AUROC and micro AUROC as our evaluation metric.

4.4.3 Experiment Results

In this section, we show the experiment results of our model on different tasks in Table 4.1-4.4. Note that we add a setting of partial data to simulate the common pre-training scenario, where the pre-training task have access to large data corpus. Specifically, the model is pre-trained on full size dataset and fine-tuned on downstream task with partial dataset.

Compares to baseline. Our framework perform significantly better than the baseline models in terms of the most important metric macro AUROC on all the down-

Dataset Size	Models	*Accuracy*	*macro AUC*	*micro AUC*
Full Size	early fusion	0.9579	0.8368	0.9938
	late fusion	0.9575	0.8229	0.9933
	Train from Scratch	0.9631	0.8582	0.9945
	MIM + fine tune	0.9654	0.8666	0.9949
	MMR + fine tune	0.9663	0.8519	0.9943
	MDA + fine tune	0.9651	0.8558	0.9943
10% Size	early fusion	0.9489	0.7570	0.9905
	late fusion	0.9543	0.7403	0.9904
	Train from Scratch	0.9512	0.7645	0.9912
	MIM + fine tune	0.9542	0.8059	0.9927
	MMR + fine tune	0.9484	0.7649	0.9910
	MDA + fine tune	0.9551	0.7745	0.9915
1% Size	early fusion	0.9537	0.6524	0.9827
	late fusion	0.9494	0.6627	0.9875
	Train from Scratch	0.9531	0.7340	0.9901
	MIM + fine tune	0.9435	0.7823	0.9917
	MMR + fine tune	0.9549	0.7367	0.9902
	MDA + fine tune	0.9540	0.7494	0.9907

Table 4.1: Results on the PhysioNet Sepsis prediction task.

stream tasks. Especially for the 1% partial data setting in the PhysioNet Sepsis prediction task, by applying our pre-training task MIM, the framework achieves 0.13 improvement on the AUROC compares to early fusion baseline. More interestingly, our method could outperform baseline models even if the training data is only 10% of baseline's training data.

Pre-training vs. Train from Scratch on full dataset. We observe that the pre-training task provides relatively low improvement on macro AUROC on most tasks with full size. Moreover, the MMR sometimes will even hurt the performance. This result suggests that our pre-training task is more suitable for the large pre-training corpus and limited fine-tuning data scenario. When the MMR is pre-trained and fine-tuned on same data size, the pre-training might be a distraction to the downstream task. This observation coincides with [93].

Dataset Size	Models	***Accuracy***	***macro AUC***	***micro AUC***
Full Size	early fusion	0.9205	0.6177	0.9360
	late fusion	0.9169	0.6230	0.9349
	Train from Scratch	0.8304	0.7073	0.8940
	MIM + fine tune	0.8354	0.7296	0.8954
	MMR + fine tune	0.8270	0.6926	0.9226
	MDA + fine tune	0.8458	0.7122	0.9209
10% Size	early fusion	0.9177	0.5745	0.9275
	late fusion	0.9207	0.5956	0.9313
	Train from Scratch	0.7497	0.6489	0.8422
	MIM + fine tune	0.9035	0.6989	0.9407
	MMR + fine tune	0.8095	0.6532	0.8854
	MDA + fine tune	0.8704	0.6634	0.9206
1% Size	early fusion	0.8601	0.5711	0.9222
	late fusion	0.8931	0.5710	0.9188
	Train from Scratch	0.8588	0.6083	0.9251
	MIM + fine tune	0.7940	0.6624	0.8973
	MMR + fine tune	0.8892	0.6294	0.9357
	MDA + fine tune	0.8637	0.6573	0.8840

Table 4.2: Results on the Girasoles Sensor AKI prediction task.

4.4.4 Conclusions

Unlike most existing works on multi-sensor representation learning for the clinical temporal data. We propose a novel multi-sensor framework that could fuse information in an attention-based halfway manner. We further provide pre-training task specifically designed for healthcare applications. Extensive experiment results validate the effectiveness of our framework and the necessity of the pre-training tasks. We observe the potential of the pre-training tasks on large data corpus, it is valuable to extend our work to public MIMIC-III [59] and eICU [100], as well as the validation on new fine-tuning tasks from these two datasets.

Dataset Size	**Models**	***Accuracy***	***macro AUC***	***micro AUC***
Full Size	early fusion	0.9035	0.6737	0.9336
	late fusion	0.9032	0.6614	0.9314
	Train from Scratch	0.8823	0.7444	0.941
	MIM + fine tune	0.8987	0.7540	0.9472
	MMR + fine tune	0.8901	0.7228	0.9408
	MDA + fine tune	0.8857	0.7260	0.9398
10% Size	early fusion	0.9032	0.6486	0.9257
	late fusion	0.9032	0.6516	0.9297
	Train from Scratch	0.9032	0.6631	0.9317
	MIM + fine tune	0.8927	0.6875	0.9338
	MMR + fine tune	0.8866	0.6683	0.9297
	MDA + fine tune	0.8790	0.6828	0.9319
1% Size	early fusion	0.9020	0.5817	0.9175
	late fusion	0.9032	0.5740	0.9161
	Train from Scratch	0.9002	0.5908	0.9190
	MIM + fine tune	0.8929	0.6125	0.9224
	MMR + fine tune	0.8663	0.5997	0.9134
	MDA + fine tune	0.9026	0.6046	0.9215

Table 4.3: Results on the Girasoles Sensor USG prediction task.

Dataset Size	**Models**	***Accuracy***	***macro AUC***	***micro AUC***
Full Size	early fusion	0.9258	0.9740	0.9909
	late fusion	0.9444	0.9774	0.9922
	Train from Scratch	0.9489	0.9812	0.9936
	MIM + fine tune	0.9503	0.9822	0.9938
	MMR + fine tune	0.9453	0.9772	0.9922
	MDA + fine tune	0.9431	0.9804	0.9931
10% Size	early fusion	0.9082	0.9466	0.9788
	late fusion	0.9246	0.9609	0.9864
	Train from Scratch	0.9198	0.9617	0.9863
	MIM + fine tune	0.9329	0.9752	0.9905
	MMR + fine tune	0.9292	0.9693	0.9890
	MDA + fine tune	0.9321	0.9725	0.9897
1% Size	early fusion	0.8147	0.8548	0.9502
	late fusion	0.8160	0.8636	0.9522
	Train from Scratch	0.8689	0.9070	0.9678
	MIM + fine tune	0.9147	0.9509	0.9830
	MMR + fine tune	0.8374	0.9109	0.9648
	MDA + fine tune	0.8716	0.9296	0.9732

Table 4.4: Results on the Girasoles Sensor Core Temperature prediction task.

Chapter 5

Conclusions and Future Works

The concept of representation learning ties many domains of deep learning, including convolutional neural network, recurrent network, encoder-decoder network and transformer based network, all exploring and exploiting the learned representations. To summarize, in this thesis we explore the deep representation learning in both single and multiple modalities.

For the single-modal representation learning, we draw inspiration from a well-known problem in physics – Thomson problem, where one seeks to find a state that distributes N electrons on a unit sphere as evenly as possible with minimum potential energy. In light of this intuition, we reduce the redundancy regularization problem to generic energy minimization, and propose a minimum hyperspherical energy (MHE) objective as generic regularization for neural networks. We also propose a few novel variants of MHE, and provide some insights from a theoretical point of view. Finally, we apply neural networks with MHE regularization to several challenging tasks. Extensive experiments demonstrate the effectiveness of our intuition, by showing the superior performance with MHE regularization. In addition, how to effectively train a neural network is of great importance. We further propose a novel orthogonal over-parameterized training (OPT) framework that can provably minimize the hyperspherical energy which characterizes the diversity of neurons on a hypersphere. By maintaining the minimum hyperspherical energy during training, OPT can greatly improve the empirical generalization. Interestingly, OPT reveals that learning a appropriate coordinate system for neurons is crucial to generalization. Extensive experiments validate the superiority of OPT over the standard training.

For the multi-modal representation learning, we propose a new use of attention over different modalities. We explore the multimodal learning framework that involves visual and textual data, structured data, as well as the timeseries data. We conduct comprehensive experiments to demonstrate the effectiveness of our framework. Learning the best possible representation from multi-modal data remains challenging and requires a lot of future works. We propose the Multi-Sensor Fusion Framework in Chapter 4, which processes and fuses multi-sensor multi-source directly. In addition, we design three pre-training task for clinical data and subsequent tasks. The pre-training method is the key component for many successful machine learning models. It would also be particularly useful for many healthcare applications where task-specific data is limited. However, the gain we observed on the pre-training task is limited in some settings of our experiments. In particular, while the model has the access to full size dataset, the pre-training task would even hurt the performance. One limitation we suspect is the size of data corpus we used, large-scale corpus is extremely helpful for the pre-training task. We plan to leverage larger public dataset for pre-training and incorporate more challenging tasks in our experiments.

Bibliography

[1] Alireza Aghasi, Afshin Abdi, Nam Nguyen, and Justin Romberg. Net-trim: A layer-wise convex pruning of deep neural networks. In *NIPS*, 2017.

[2] Selim Aksoy and Robert M Haralick. Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern recognition letters*, 22(5): 563–582, 2001.

[3] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018.

[4] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[5] Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary evolution recurrent neural networks. In *ICML*, 2016.

[6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[7] Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. Can we gain more from orthogonality regularizations in training deep networks? In *NeurIPS*, 2018.

[8] J Batle, Armen Bagdasaryan, M Abdel-Aty, and S Abdalla. Generalized thomson problem in arbitrary dimensions and non-euclidean geometries. *Physica A: Statistical Mechanics and its Applications*, 451:237–250, 2016.

[9] Jeffrey P Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, et al. Vizwiz: nearly real-time answers to visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 333–342, 2010.

[10] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[11] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Neural photo editing with introspective adversarial networks. *arXiv preprint arXiv:1609.07093*, 2016.

[12] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Neural photo editing with introspective adversarial networks. In *ICLR*, 2017.

[13] Matthew Calef, Whitney Griffiths, and Alexia Schulz. Estimating the number of stable configurations for the generalized thomson problem. *Journal of Statistical Physics*, 160(1):239–253, 2015.

[14] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. In *ICLR*, 2017.

[15] Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. On sampling strategies for neural network-based collaborative filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 767–776, 2017.

[16] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[17] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[18] Krzysztof Choromanski, Carlton Downey, and Byron Boots. Initialization matters: Orthogonal predictive state recurrent neural networks. In *ICLR*, 2018.

[19] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[20] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Gated feedback recurrent neural networks. In *International conference on machine learning*, pages 2067–2075. PMLR, 2015.

[21] Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, and Dhruv Batra. Reducing overfitting in deep networks by decorrelating representations. *arXiv preprint arXiv:1511.06068*, 2015.

[22] Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, and Dhruv Batra. Reducing overfitting in deep networks by decorrelating representations. In *ICLR*, 2016.

[23] Harald Cramér. *Mathematical methods of statistics (PMS-9)*, volume 9. Princeton university press, 2016.

[24] Bo Dai, Hanjun Dai, Niao He, Weiyang Liu, Zhen Liu, Jianshu Chen, Lin Xiao, and Le Song. Coupled variational bayes via optimization embedding. In *NeurIPS*, 2018.

[25] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *arXiv preprint arXiv:1801.07698*, 2018.

[26] Xiaohan Ding, Yuchen Guo, Guiguang Ding, and Jungong Han. Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks. In *ICCV*, 2019.

[27] Xin Luna Dong, Xiang He, Andrey Kan, Xian Li, Yan Liang, Jun Ma, Yifan Ethan Xu, Chenwei Zhang, Tong Zhao, Gabriel Blanco Saldana, et al. Autoknow: Self-driving knowledge collection for products of thousands of types. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2724–2734, 2020.

[28] Dmitrii Emelianenko, Elena Voita, and Pavel Serdyukov. Sequence modeling with unconstrained generation order. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 7700–7711. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/1558417b096b5d8e7cbe0183ea9cbf26-Paper.pdf`.

[29] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

[30] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

[31] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

[32] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

[33] Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. In *NeurIPS*, 2017.

[34] Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. In *NeurIPS*, 2018.

[35] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016.

[36] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *ICLR*, 2016.

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

[38] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[40] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[41] Mikael Henaff, Arthur Szlam, and Yann LeCun. Recurrent orthogonal networks and long-memory tasks. *arXiv preprint arXiv:1602.06662*, 2016.

[42] Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*, 2017.

[43] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 1997.

[44] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[45] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899, 2013.

[46] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2019.

[47] Chiori Hori, Takaaki Hori, Teng-Yok Lee, Ziming Zhang, Bret Harsham, John R Hershey, Tim K Marks, and Kazuhiko Sumi. Attention-based multimodal fusion for video description. In *Proceedings of the IEEE international conference on computer vision*, pages 4193–4202, 2017.

[48] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[49] Ronghang Hu, Amanpreet Singh, Trevor Darrell, and Marcus Rohrbach. Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9992–10002, 2020.

[50] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report, 2007.

[51] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. *Technical Report*, 2008.

[52] Lei Huang, Xianglong Liu, Bo Lang, Adams Wei Yu, Yongliang Wang, and Bo Li. Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks. In *AAAI*, 2018.

[53] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[54] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[55] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[56] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *UAI*, 2018.

[57] Lu Jiang, Deyu Meng, Shoou-I Yu, Zhenzhong Lan, Shiguang Shan, and Alexander Hauptmann. Self-paced learning with diversity. In *NIPS*, 2014.

[58] Li Jing, Yichen Shen, Tena Dubcek, John Peurifoy, Scott Skirlo, Yann LeCun, Max Tegmark, and Marin Soljačić. Tunable efficient unitary neural networks (eunn) and their application to rnns. In *ICML*, 2017.

[59] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

[60] Giannis Karamanolakis, Jun Ma, and Xin Luna Dong. Txtract: Taxonomy-aware knowledge extraction for thousands of product categories. *arXiv preprint arXiv:2004.13852*, 2020.

[61] Kenji Kawaguchi. Deep learning without poor local minima. In *NeurIPS*, 2016.

[62] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *CVPR*, 2016.

[63] Nitish Shirish Keskar and Richard Socher. Improving generalization performance by switching from adam to sgd. *arXiv preprint arXiv:1712.07628*, 2017.

[64] Arif Khwaja. Kdigo clinical practice guidelines for acute kidney injury. *Nephron Clinical Practice*, 120(4):c179–c184, 2012.

[65] Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 1571–1581, 2018.

[66] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[67] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[68] Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207, 2003.

[69] Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2020.

[70] Mario Lezcano-Casado and David Martínez-Rubio. Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group. In *ICML*, 2019.

[71] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, 2018.

[72] Jianxin Li, Haoyi Zhou, Pengtao Xie, and Yingchun Zhang. Improving the generalization performance of multi-class svm via angular regularization. In *IJCAI*, 2017.

[73] Jun Li, Fuxin Li, and Sinisa Todorovic. Efficient riemannian optimization on the stiefel manifold via the cayley transform. In *ICLR*, 2020.

[74] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50(6):1–45, 2017.

[75] Nan Li, Yang Yu, and Zhi-Hua Zhou. Diversity regularized ensemble pruning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2012.

[76] Yuanzhi Li, Tengyu Ma, and Hongyang Zhang. Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations. In *COLT*, 2018.

[77] Minghui Liao, Guan Pang, Jing Huang, Tal Hassner, and Xiang Bai. Mask textspotter v3: Segmentation proposal network for robust scene text spotting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[78] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[79] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, 2016.

[80] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, 2016.

[81] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017.

[82] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017.

[83] Weiyang Liu, Yan-Ming Zhang, Xingguo Li, Zhiding Yu, Bo Dai, Tuo Zhao, and Le Song. Deep hyperspherical learning. In *NeurIPS*, 2017.

[84] Weiyang Liu, Yan-Ming Zhang, Xingguo Li, Zhiding Yu, Bo Dai, Tuo Zhao, and Le Song. Deep hyperspherical learning. In *NIPS*, 2017.

[85] Weiyang Liu, Rongmei Lin, Zhen Liu, Lixin Liu, Zhiding Yu, Bo Dai, and Le Song. Learning towards minimum hyperspherical energy. *NeurIPS*, 2018.

[86] Weiyang Liu, Rongmei Lin, Zhen Liu, Lixin Liu, Zhiding Yu, Bo Dai, and Le Song. Learning towards minimum hyperspherical energy. In *NeurIPS*, 2018.

[87] Weiyang Liu, Zhen Liu, Zhiding Yu, Bo Dai, Rongmei Lin, Yisen Wang, James M Rehg, and Le Song. Decoupled networks. *CVPR*, 2018.

[88] Weiyang Liu, Zhen Liu, James M Rehg, and Le Song. Neural similarity learning. In *NeurIPS*, 2019.

[89] Yu Liu, Hongyang Li, and Xiaogang Wang. Rethinking feature discrimination and polymerization for large-scale recognition. *arXiv preprint arXiv:1710.00870*, 2017.

[90] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*, 2019.

[91] George Luber and Michael McGeehin. Climate change and extreme heat events. *American journal of preventive medicine*, 35(5):429–435, 2008.

[92] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *ICML*, 2009.

[93] Matthew McDermott, Bret Nestor, Evan Kim, Wancong Zhang, Anna Goldenberg, Peter Szolovits, and Marzyeh Ghassemi. A comprehensive ehr timeseries pre-training benchmark. In *Proceedings of the Conference on Health, Inference, and Learning*, pages 257–278, 2021.

[94] Dmytro Mishkin and Jiri Matas. All you need is a good init. In *ICLR*, 2016.

[95] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

[96] Louis-Philippe Morency, Ariadna Quattoni, and Trevor Darrell. Latent-dynamic discriminative models for continuous gesture recognition. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

[97] Louis-Philippe Morency, Rada Mihalcea, and Payal Doshi. Towards multimodal sentiment analysis: Harvesting opinions from the web. In *Proceedings of the 13th international conference on multimodal interfaces*, pages 169–176, 2011.

[98] Athma Narayanan, Avinash Siravuru, and Behzad Dariush. Sensor fusion: Gated recurrent fusion to learn driving behavior from temporal multimodal data. *arXiv preprint arXiv:1910.00628*, 2019.

[99] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[100] Tom J Pollard, Alistair EW Johnson, Jesse D Raffa, Leo A Celi, Roger G Mark, and Omar Badawi. The eicu collaborative research database, a freely available multi-center database for critical care research. *Scientific data*, 5(1):1–13, 2018.

[101] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017.

[102] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep

hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[103] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[104] Ignacio Ramirez, Pablo Sprechmann, and Guillermo Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *CVPR*, 2010.

[105] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

[106] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[107] Matthew A Reyna, Chris Josef, Salman Seyedi, Russell Jeter, Supreeth P Shashikumar, M Brandon Westover, Ashish Sharma, Shamim Nemati, and Gari D Clifford. Early prediction of sepsis from clinical data: the physionet/computing in cardiology challenge 2019. In *2019 Computing in Cardiology (CinC)*, pages Page–1. IEEE, 2019.

[108] Pau Rodríguez, Jordi Gonzalez, Guillem Cucurull, Josep M Gonfaus, and Xavier Roca. Regularizing cnns with locally constrained decorrelations. *arXiv preprint arXiv:1611.01967*, 2016.

[109] Pau Rodríguez, Jordi Gonzalez, Guillem Cucurull, Josep M Gonfaus, and Xavier Roca. Regularizing cnns with locally constrained decorrelations. In *ICLR*, 2017.

[110] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, pages 1–42, 2014.

[111] Dymitr Ruta and Bogdan Gabrys. Classifier selection for majority voting. *Information fusion*, 6(1):63–81, 2005.

[112] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

[113] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 2008.

[114] Christopher W Seymour, Vincent X Liu, Theodore J Iwashyna, Frank M Brunkhorst, Thomas D Rea, André Scherag, Gordon Rubenfeld, Jeremy M Kahn, Manu Shankar-Hari, Mervyn Singer, et al. Assessment of clinical criteria for sepsis: for the third international consensus definitions for sepsis and septic shock (sepsis-3). *Jama*, 315(8):762–774, 2016.

[115] Manu Shankar-Hari, Gary S Phillips, Mitchell L Levy, Christopher W Seymour, Vincent X Liu, Clifford S Deutschman, Derek C Angus, Gordon D Rubenfeld, Mervyn Singer, et al. Developing a new definition and assessing new clinical criteria for septic shock: for the third international consensus definitions for sepsis and septic shock (sepsis-3). *Jama*, 315(8):775–787, 2016.

[116] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[117] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[118] Mervyn Singer, Clifford S Deutschman, Christopher Warren Seymour, Manu Shankar-Hari, Djillali Annane, Michael Bauer, Rinaldo Bellomo, Gordon R Bernard, Jean-Daniel Chiche, Craig M Coopersmith, et al. The third international consensus definitions for sepsis and septic shock (sepsis-3). *Jama*, 315 (8):801–810, 2016.

[119] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8317–8326, 2019.

[120] Steve Smale. Mathematical problems for the next century. *The mathematical intelligencer*, 20(2):7–15, 1998.

[121] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.

[122] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *CVPR*, 2014.

[123] Cheng Tai, Tong Xiao, Yi Zhang, Xiaogang Wang, et al. Convolutional neural networks with low-rank regularization. *arXiv preprint arXiv:1511.06067*, 2015.

[124] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019.

[125] Joseph John Thomson. Xxiv. on the structure of the atom: an investigation of the stability and periods of oscillation of a number of corpuscles arranged

at equal intervals around the circumference of a circle; with application of the results to the theory of atomic structure. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 7(39):237–265, 1904.

[126] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[127] Oriol Vinyals, Igor Babuschkin, Junyoung Chung, Michael Mathieu, Max Jaderberg, Wojciech M Czarnecki, Andrew Dudzik, Aja Huang, Petko Georgiev, Richard Powell, et al. Alphastar: Mastering the real-time strategy game starcraft ii. *DeepMind blog*, page 2, 2019.

[128] Feng Wang, Weiyang Liu, Haijun Liu, and Jian Cheng. Additive margin softmax for face verification. *arXiv preprint arXiv:1801.05599*, 2018.

[129] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Zhifeng Li, Dihong Gong, Jingchao Zhou, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. *arXiv preprint arXiv:1801.09414*, 2018.

[130] Qifan Wang, Li Yang, Bhargav Kanagal, Sumit Sanghai, D Sivakumar, Bin Shu, Zac Yu, and Jon Elsas. Learning to extract attribute value from product via question answering: A multi-task approach. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 47–55, 2020.

[131] Sy Bor Wang, Ariadna Quattoni, L-P Morency, David Demirdjian, and Trevor Darrell. Hidden conditional random fields for gesture recognition. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1521–1527. IEEE, 2006.

[132] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016.

[133] Zaiwen Wen and Wotao Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 2013.

[134] Scott Wisdom, Thomas Powers, John Hershey, Jonathan Le Roux, and Les Atlas. Full-capacity unitary recurrent neural networks. In *NIPS*, 2016.

[135] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015.

[136] Di Xie, Jiang Xiong, and Shiliang Pu. All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation. *arXiv:1703.01827*, 2017.

[137] Pengtao Xie, Jun Zhu, and Eric Xing. Diversity-promoting bayesian learning of latent variable models. In *ICML*, 2016.

[138] Pengtao Xie, Yuntian Deng, Yi Zhou, Abhimanu Kumar, Yaoliang Yu, James Zou, and Eric P Xing. Learning latent space models with angular constraints. In *ICML*, 2017.

[139] Pengtao Xie, Aarti Singh, and Eric P Xing. Uncorrelation and evenness: a new diversity-promoting regularizer. In *ICML*, 2017.

[140] Pengtao Xie, Wei Wu, Yichen Zhu, and Eric P Xing. Orthogonality-promoting distance metric learning: convex relaxation and theoretical analysis. In *ICML*, 2018.

[141] Huimin Xu, Wenting Wang, Xinnian Mao, Xinyu Jiang, and Man Lan. Scaling up open tagging from tens to thousands: Comprehension empowered attribute

value extraction from product title. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5214–5223, 2019.

[142] Zhengyuan Yang, Yijuan Lu, Jianfeng Wang, Xi Yin, Dinei Florencio, Lijuan Wang, Cha Zhang, Lei Zhang, and Jiebo Luo. Tap: Text-aware pre-training for text-vqa and text-caption, 2020.

[143] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv:1411.7923*, 2014.

[144] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*, 2017.

[145] Amir Zadeh, Paul Pu Liang, Navonil Mazumder, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. Memory fusion network for multi-view sequential learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[146] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Making visual representations matter in vision-language models, 2021.

[147] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *arXiv preprint arXiv:1707.01083*, 2017.

[148] Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. Opentag: Open attribute value extraction from product profiles. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1049–1058, 2018.