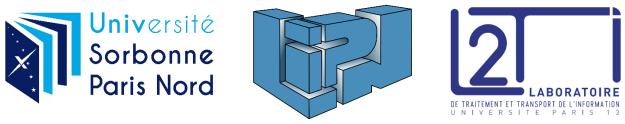


Enhancing Neural Network Efficiency Through Prior Knowledge

Normalization, Multimodal Representation, and Open-Vocabulary Object Detection

Bilal FAYE

This thesis was conducted at Sorbonne Paris Nord University,
within the *LIPN* (Laboratoire d'Informatique de Paris-Nord), UMR CNRS 7030,
and the *L2TI* (Laboratoire de Traitement et de Transmission de l'Information),
at the Institut Galilée in Villetaneuse, France.

June 2025

Thesis Supervisor:	Hanane AZZAG, Professor, Sorbonne Paris Nord University, LIPN
Co-supervisor:	Fanchen FENG, Associate Professor, Sorbonne Paris Nord University, L2TI
Reviewers:	Cédric WEMMERT, Professor, University of Strasbourg
	Céline HUDELOT, Professor, CentraleSupélec
Examiners:	Camille KURTZ, Professor, Paris Cité University
	Mustapha LEBBAH, Professor, Paris-Saclay - UVSQ University
Invited Member:	Djamel BOUCHAFFRA, Director of Research, CDTA, Algiers

Remerciements

Je tiens à exprimer ma gratitude envers toutes les personnes qui m'ont soutenu tout au long de ce parcours.

À mes parents, Insa FAYE et Boudouel DIOP : Votre amour incommensurable, votre sagesse et votre foi en moi ont été les fondations solides sur lesquelles j'ai bâti mon parcours académique. Merci pour votre soutien indéfectible.

À mon épouse, Aminata THIOUNE : Merci pour ta patience, ton soutien sans faille, et ton amour. Tu as été un pilier dans les moments les plus intenses de cette aventure, et sans toi, rien de cela n'aurait été possible.

À mes sœurs, Ndeye Marie FAYE et Lena FAYE : Votre affection et vos encouragements m'ont toujours donné la force d'avancer. Vous avez été des sources constantes d'inspiration.

En mémoire de M. Omar AIDARA : Ta passion pour la technologie et l'IA continue de me guider. Que ton âme repose en paix.

À M. Jean Joseph LAPOLICE : Tu as éveillé en moi la curiosité intellectuelle. Ton influence perdure à travers ce travail.

À Gérard GARDARIN : Comme un père, ton soutien et ta sagesse ont été inestimables. Repose en paix.

À Yola GARDARIN : Merci pour ta générosité et ton soutien constant, tu as été une aide précieuse tout au long de cette aventure.

À Mambodj DIOP et Mbathio SOW : Votre accueil chaleureux en France a été une source de réconfort. Merci de m'avoir fait sentir chez moi loin de chez moi.

Contents

List of Acronyms

AI	Artificial Intelligence
DNN	Deep Neural Network
BN	Batch Normalization
LN	Layer Normalization
IN	Instance Normalization
GN	Group Normalization
DN	Divisive Normalization
UBN	Unsupervised Batch Normalization
SwitchNorm	Switchable Normalization
ModeNorm	Mode Normalization
MixNorm	Mixture Normalization
CN	Context Normalization
CN-X	Context Normalization Extended
ACN	Adaptive Context Normalization
EM	Expectation Maximization
GMM	Gaussian Mixture Model
UP	Universal Projection
OneEncoder	Framework with a single trainable component, the UP
OneEncoder-$\oplus$	OneEncoder with addition fusion
OneEncoder-$\odot$	OneEncoder with cross-attention fusion
MDETR	Modulated Detection for End-to-End Multi-Modal Understanding
LightMDETR	LightWeight MDETR
LightMDETR-Plus	LightWeight MDETR with Cross-Fusion Attention Layer

Introduction

French. L'apprentissage profond a profondément transformé l'intelligence artificielle, en permettant aux modèles d'apprendre automatiquement des représentations complexes à partir de grandes quantités de données. Ces progrès ont conduit à des percées majeures dans des domaines comme la vision par ordinateur, le traitement du langage naturel et les systèmes intelligents. Néanmoins, ces modèles restent fortement dépendants de données annotées massives, sont coûteux à entraîner, et rencontrent des difficultés à généraliser hors des distributions vues pendant l'apprentissage. Cette thèse explore une voie prometteuse pour dépasser ces limites : l'intégration de connaissances a priori dans les architectures d'apprentissage profond. En exploitant des informations externes ou structurées, il devient possible de guider l'apprentissage, de stabiliser l'entraînement, et d'améliorer la robustesse des représentations. Nous étudions cette approche selon trois axes complémentaires : l'optimisation de techniques de normalisation efficaces, l'alignement intermodal pour le traitement de données multimodales, et la détection d'objets à vocabulaire ouvert, un cadre dans lequel les modèles doivent être capables de reconnaître des catégories non vues pendant l'entraînement, en s'appuyant sur des connaissances sémantiques.

Deep learning has revolutionized a wide range of disciplines by enabling models to automatically learn complex patterns from vast amounts of data, outperforming traditional machine learning approaches that rely on hand-crafted features. This capability has driven remarkable progress in fields such as computer vision, natural language processing (NLP), speech recognition, and multimodal learning, powering innovations like autonomous systems, large language models, and medical diagnostics. From image segmentation and object detection to machine translation and generative models, deep learning systems have surpassed human-level performance in many benchmarks, fueling a new era of artificial intelligence research and commercial applications.

The hierarchical nature of neural networks allows them to capture abstract representations, progressively building higher-level features from raw inputs. Advances in architectures — such as transformers, convolutional networks, and graph neural networks — have expanded the scope of tasks that deep learning can tackle, while optimizations in training strategies, regularization techniques, and distributed computing have enabled training on ever-larger datasets. However, these successes come with significant costs: deep models are computationally expensive to train and deploy, often require millions of labeled examples, and struggle to generalize beyond the distribution of their training data.

In light of these challenges, integrating **prior knowledge** into deep learning models has emerged as a promising strategy to enhance efficiency and performance. By embedding domain-specific information, models can learn more robust representations, reduce the reliance on extensive labeled datasets, and improve generalization. For example, normalization techniques that incorporate statistical insights can stabilize training dynamics, while cross-modal alignment methods enable better information fusion across diverse data modalities. Similarly, leveraging structured knowledge can facilitate scalable frameworks for complex tasks like open-vocabulary object detection, where the ability to recognize unseen categories is crucial.

This thesis investigates how prior knowledge can be systematically integrated to address the limitations of deep learning, focusing on three interconnected areas: efficient normalization techniques, effective cross-modal alignment for multimodal data representation, and scalable frameworks for open-vocabulary object detection training.

1.1 Prior Knowledge in Deep Learning

Prior knowledge in deep learning refers to the integration of existing information or assumptions about a problem domain that can guide the learning process [137, 14]. This knowledge can come from a variety of sources, such as expert insights, statistical properties of the data, or predefined models trained on related tasks. By incorporating such knowledge, deep learning models can start with a more informed representation, allowing them to make more efficient use of available data and reduce the need for large labeled datasets.

One of the primary benefits of incorporating prior knowledge is that it enhances generalization. Models are better able to avoid overfitting, especially when dealing with limited data. Additionally, it enables faster convergence during training by guiding the model towards more meaningful feature representations and improving robustness. This becomes particularly important in situations where data is scarce or costly to acquire.

Prior knowledge can be integrated into deep learning models in a variety of ways. For example, statistical knowledge can inform initialization schemes or regularization methods, domain-specific constraints can guide the architecture of the network, and semantic relationships (e.g., between objects in a scene) can influence how the model processes data.

The three next sections of this thesis explore how prior knowledge is specifically integrated into three key areas: normalization techniques, cross-modal alignment, and open-vocabulary object detection.

1.2 Normalizing Deep Learning Models Using Prior Knowledge of Data Distributions

Normalization techniques are critical in deep learning, addressing challenges such as vanishing and exploding gradients that hinder the training of deep networks. Methods like batch normalization [54] and layer normalization [4] stabilize training by rescaling activations, improving both convergence speed and model generalization. However, many existing methods make overly simplistic assumptions about data distribution, which may not hold in real-world, heterogeneous datasets. Additionally, these methods often lack adaptability to domain-specific tasks or low-resource settings, leading

to inefficiencies.

To address these challenges, we introduce Context Normalization, a novel technique that integrates prior knowledge about the data distribution to enhance performance and accelerate convergence. By incorporating domain-specific statistical insights or known data properties, we guide the normalization process to improve scalability and adaptability. **Context Normalization** is presented in three variants: **Context Normalization**, **Context Normalization - Extended**, and **Adaptive Context Normalization**, each designed to improve deep learning models' efficiency. These techniques are validated across various domains, including image classification, image generation, and domain adaptation, demonstrating their effectiveness in improving training performance.

1.3 Improving Multimodal Data Representation Through Cross-Modal Alignment Using Prior Knowledge of Modalities

Multimodal learning involves the integration and alignment of data from various modalities, such as images, text, and audio, to capture meaningful cross-modal relationships. Vision-language models, such as CLIP [94], have shown impressive performance by jointly embedding text and visual features. However, achieving effective cross-modal alignment requires massive datasets and substantial computational resources, presenting challenges for domain-specific applications in low-resource settings. Furthermore, existing approaches often struggle to generalize across diverse modalities, resulting in suboptimal performance.

To address these limitations, we propose **OneEncoder**, a progressive alignment framework that incorporates prior knowledge specific to the modality of the given data. By leveraging semantic relationships and domain-specific knowledge relevant to each modality, our approach alleviates resource constraints and improves generalization across different data types. We demonstrate its application in zero-shot classification, querying, and visual question answering, utilizing modalities such as text, image, audio, and video.

1.4 Enhancing Open-Vocabulary Object Detection with Modality-Specific Prior Knowledge

Traditional object detection methods are limited to recognizing only the categories seen during training, restricting their applicability in dynamic environments where new categories frequently emerge. Open-vocabulary object detection (OVOD) addresses this limitation by enabling models to recognize objects beyond their training categories using textual descriptions. However, existing OVOD methods rely on computationally intensive vision-language models and large-scale datasets, making them difficult to deploy in resource-constrained or domain-specific settings. Balancing generalization to unseen categories with accurate detection of seen categories remains a persistent challenge.

To address these issues, we propose a modular framework that integrates prior knowledge specific to the object categories and the modalities they belong to, reducing training costs while maintaining high accuracy. By incorporating knowledge of object semantics and category relationships, we enhance both the scalability and adaptability of the framework. We demonstrate the approach through **LightMDETR**, an adaptation of the MDETR [58] model, and validate its performance on tasks such as Phrase Grounding, Referring Expression Comprehension, and Referring Expression Segmentation.

1.5 Thesis Objectives

The overarching goal of this thesis is to address critical challenges in deep learning related to efficiency, scalability, and adaptability, with a particular focus on enhancing its applicability to diverse and resource-constrained environments. By leveraging prior knowledge, the thesis aims to propose solutions that bridge gaps in existing methodologies and contribute to the broader adoption and effectiveness of deep learning. The specific objectives of the thesis are as follows:

1. **Enhancing Training Efficiency and Generalization:** Modern deep learning models often

face challenges such as slow convergence and limited generalization when applied to complex, heterogeneous, or domain-specific datasets. This thesis aims to develop strategies that not only address these limitations but also improve the computational efficiency of training, making models more practical for real-world applications. By tackling these challenges, this work aspires to enable deep learning models to perform effectively across a wide range of tasks and data distributions.

2. **Advancing Multimodal Learning:** Integrating and aligning multimodal data, including text, images, audio, and video, is crucial for capturing rich and meaningful representations. Current approaches often require large paired datasets and significant computational resources, which may be infeasible in many scenarios. This thesis seeks to develop frameworks that facilitate efficient and robust cross-modal alignment, improving generalization across modalities while reducing dependence on large-scale datasets. Achieving this will open new avenues for multimodal applications, particularly in domains with limited resources.

3. **Enabling Scalable Open-Vocabulary Object Detection:** The ability to detect and recognize unseen object categories is vital for deploying object detection systems in dynamic environments. However, existing methods are limited by their reliance on fixed training categories and resource-intensive vision-language models. This thesis aims to address these issues by developing scalable and adaptable frameworks for open-vocabulary object detection. This will expand the applicability of object detection models, particularly in low-resource and domain-specific settings.

4. **Promoting Practicality and Accessibility of Deep Learning:** Beyond theoretical advancements, this thesis aims to bridge the gap between research and practical deployment. The solutions developed will prioritize resource efficiency, making them accessible to researchers and practitioners in various domains, including those with limited computational resources. This emphasis on accessibility ensures that the work has a tangible impact on the broader field of artificial intelligence and its real-world applications.

1.6 Contributions of the Thesis

This thesis makes significant contributions to the field of deep learning by addressing critical challenges in efficiency, scalability, and adaptability. The work advances both theoretical understanding and practical implementations, validated across various domains and applications. The key contributions are as follows:

1. **Development of Advanced Normalization Techniques:** This thesis introduces **Context Normalization**, along with its two variants—**Context Normalization-Extended** and **Adaptive Context Normalization**. These methods integrate prior knowledge to address the limitations of existing normalization techniques, leading to enhanced training efficiency, faster convergence, and improved generalization. These techniques are extensively validated on tasks such as image classification, image generation, and domain adaptation, demonstrating their effectiveness and scalability.

2. **Creation of a Progressive Cross-Modal Alignment Learning Framework for Multimodal Data Representation:** The proposed **OneEncoder** framework offers a lightweight and efficient solution for multimodal representation learning. By leveraging prior knowledge, it enables seamless alignment across modalities—such as text, image, audio, and video—while minimizing reliance on large-scale paired datasets. OneEncoder achieves state-of-the-art performance in applications, including zero-shot classification, querying, and visual question answering, highlighting its practical utility.

3. **Proposing a Modular Framework for Open-Vocabulary Object Detection:** Building upon the MDETR model, this thesis presents **LightMDETR**, a modular framework that addresses the challenges of scalability and adaptability in open-vocabulary object detection. By incorporating prior knowledge, LightMDETR achieves efficient generalization to unseen object categories while reducing computational costs. It is validated through tasks such as phrase grounding, referring expression comprehension, and referring expression segmentation, showcasing its robustness and versatility.

4. **Extensive Empirical Validation:** The methods and frameworks proposed in this thesis

are rigorously evaluated on a wide range of benchmark datasets and tasks spanning computer vision and multimodal applications. This comprehensive validation underscores the effectiveness, efficiency, and broad applicability of the proposed approaches.

5. **Theoretical Insights and Practical Guidelines:** This work provides an in-depth exploration of the role of prior knowledge in deep learning, offering valuable theoretical insights into its integration for improving efficiency and generalization. Additionally, practical guidelines for researchers and practitioners are presented, facilitating the adoption of the proposed methods in real-world scenarios.

1.7 Overview

The thesis is structured into four main parts, each addressing critical challenges in deep learning with the aim of improving efficiency, scalability, and adaptability across various domains. The organization of the thesis is as follows:

- **Part I: Enhancing Deep Learning Training through Advanced Normalization Techniques** This section examines the limitations of existing normalization methods in deep learning and introduces Context Normalization along with its two variants: Context Normalization-Extended and Adaptive Context Normalization. These methods leverage prior knowledge to stabilize training, accelerate convergence, and improve generalization across diverse and domain-specific data distributions. Their effectiveness is validated in tasks such as image classification, image generation, and domain adaptation.
- **Part II: Efficient Multimodal Representation Learning** This section addresses the challenges of multimodal representation learning by proposing the OneEncoder framework. This lightweight and progressive alignment approach reduces the dependency on large-scale paired datasets and integrates prior knowledge to achieve efficient cross-modal alignment. Applications of this framework are demonstrated in tasks such as zero-shot classification, querying, and visual question answering across modalities including text, image, audio, and video.

- **Part III: Modular Framework for Open-Vocabulary Object Detection** This section focuses on the challenges of open-vocabulary object detection and introduces the LightMDETR framework, built on the MDETR model. LightMDETR utilizes prior knowledge to enable efficient generalization to unseen object categories while significantly reducing computational costs. The framework is validated through tasks such as phrase grounding, referring expression comprehension, and referring expression segmentation, showcasing its adaptability and robustness in dynamic and low-resource environments.

- **Part IV: Conclusions and Future Directions** The final section summarizes the key contributions of the thesis and their impact on the field of deep learning. It also discusses the broader implications of the proposed methods and outlines promising directions for future research, including potential extensions to other domains and further refinement of the frameworks introduced.

Part I

Normalization in Deep Learning

Efficient and stable training of deep neural networks (DNNs) is a persistent challenge, particularly in scenarios involving complex and heterogeneous data. This part examines the role of normalization techniques in addressing these challenges, emphasizing their importance in enhancing model scalability, adaptability, and convergence.

Chapter 2 outlines the limitations of existing normalization strategies, particularly their inability to adapt to diverse data distributions and domain-specific tasks. Building on this, Chapter 3 introduces our proposed method **Context Normalization**, along with its variants **Context Normalization-Extended** and **Adaptive Context Normalization**. These methods are designed to improve training efficiency and generalization across applications such as image classification, image generation, and domain adaptation.

This part establishes the foundation for a deeper understanding of normalization's impact on deep learning and demonstrates the effectiveness of the proposed methods through detailed experimental evaluation.

Chapter 2

State of the Art in Activation Normalization for DNNs

2.1 Introduction

DNNs are powerful models characterized by stacked layers that apply linear transformations followed by nonlinear activation functions. While their complex architectures enable effective feature learning and strong representational power, they also present significant challenges during training. Issues such as slow convergence, overfitting, and training instability arise due to factors like vanishing gradients and sensitivity to hyperparameters.

The success of DNNs largely depends on advancements in training methodologies that address these challenges. One crucial advancement is *normalization*, which enhances training stability, improves optimization efficiency, and boosts generalization performance [54, 4, 124, 136]. Normalization techniques help mitigate the difficulties associated with training deep networks, allowing them to learn more effectively.

Normalization is commonly applied in data preprocessing, data mining, and various other domains. It refers to a general transformation process that ensures the resulting data exhibits specific statistical characteristics. Given a dataset $\mathbf{x} \in \mathbb{R}^d$, normalization is defined as a function $f : \mathbf{x} \to \hat{\mathbf{x}}$, which guarantees that the transformed data $\hat{\mathbf{x}}$ meets specific statistical properties. Several key normalization techniques exist, including centering, scaling, standardizing, decorrelating, and whitening [51].

Centering defines the transformation as:

$$\hat{\mathbf{x}} = f_c(\mathbf{x}) = \mathbf{x} - \mathbb{E}(\mathbf{x}). \tag{2.1}$$

This operation ensures that the normalized output $\hat{\mathbf{x}}$ has a mean of zero, expressed as: $\mathbb{E}(\hat{\mathbf{x}}) = 0$.

Scaling defines the transformation as:

$$\hat{x} = f_s(\mathbf{x}) = \Lambda^{-\frac{1}{2}}\mathbf{x}. \tag{2.2}$$

Here, $\Lambda = \text{diag}(\sigma_1^2, \ldots, \sigma_d^2)$, where σ_j^2 represents the variance of the data samples for the j-th dimension, calculated as $\sigma_j^2 = \mathbb{E}(\mathbf{x}_j^2) - [\mathbb{E}(\mathbf{x}_j)]^2$. Scaling ensures that the normalized output $\hat{\mathbf{x}}$ has a unit variance, expressed as $\mathbb{E}(\hat{\mathbf{x}}_j^2) - [\mathbb{E}(\hat{\mathbf{x}}_j)]^2 = 1$ for all $j = 1, \ldots, d$.

Standardizing is an operation that integrates both centering and scaling, defined as:

$$\hat{\mathbf{x}} = f_{st}(\mathbf{x}) = \Lambda^{-\frac{1}{2}}(\mathbf{x} - \mathbb{E}(\mathbf{x})). \tag{2.3}$$

This process guarantees that the normalized output $\hat{\mathbf{x}}$ possesses both zero mean and unit variance properties.

Decorrelating defines the transformation as:

$$\hat{\mathbf{x}} = f_d(\mathbf{x}) = \mathbf{D}\mathbf{x} \tag{2.4}$$

where $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_d]$ represents the eigenvectors of the covariance matrix $\Sigma = \mathbb{E}(\mathbf{x}\mathbf{x}^T)$. Decorrelating ensures that the correlation between different dimensions of the normalized output $\hat{\mathbf{x}}$ is zero, meaning that the covariance matrix $\mathbb{E}(\hat{\mathbf{x}}\hat{\mathbf{x}}^T)$ is a diagonal matrix.

Whitening defines the transformation as:

$$\hat{\mathbf{x}} = f_w(\mathbf{x}) = \tilde{\Lambda}^{-\frac{1}{2}}\mathbf{D}\mathbf{x} \tag{2.5}$$

where $\tilde{\Lambda} = \text{diag}(\tilde{\sigma}_1, \ldots, \tilde{\sigma}_d)$ and $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_d]$ represent the eigenvalues and corresponding eigenvectors of the covariance matrix Σ. Whitening ensures that the normalized output $\hat{\mathbf{x}}$ follows a spherical Gaussian distribution, which can be expressed as: $\mathbb{E}(\hat{\mathbf{x}}\hat{\mathbf{x}}^T) = \mathbf{I}$.

In DNNs, applying normalization methods to input data is crucial for training, as they reduce variations in feature magnitudes. While this normalization can accelerate convergence in networks with a single hidden layer [69], its effectiveness in multi-layer networks is less certain. This uncertainty arises because each layer transforms the data, leading to activations that may not retain the characteristics of the normalized inputs. Therefore, normalizing activations during training is essential for maintaining the advantages of input normalization. By ensuring a consistent

statistical distribution of activations across layers, DNNs achieve more stable and efficient training, ultimately enhancing model performance.

Batch Normalization (BN), introduced by Ioffe and Szegedy in their influential work [54], has become the dominant and widely-used technique for normalizing activations in DNNs. BN standardizes activations using batch-level statistics, which enables the use of higher learning rates and improves training efficiency. However, BN has limitations, particularly its dependence on batch size and the assumption of a uniform data distribution. To mitigate the *batch size dependence* issue, various single-mode normalization methods have been proposed [4, 114, 53, 124, 136, 61]. Additionally, to address the *uniform data distribution assumption*, multi-mode normalization methods have been developed [81, 80, 57, 73].

2.2 Single-mode normalization

Single-mode normalization refers to normalization techniques that operate by standardizing activations using statistics computed from a single mode or source, such as a layer or mini-batch of data. These methods were pioneered by Batch Normalization (BN), introduced by Ioffe and Szegedy in their seminal work [54], which became a cornerstone of training deep neural networks.

2.2.1 Batch Normalization Method

BN normalizes activations by using the mean and variance calculated over mini-batches during training. This approach mitigates the problem of *internal covariate shift*—the tendency of layer inputs to change distribution during training—thereby allowing higher learning rates and faster convergence. The normalization is done by centering the activations around zero with a mean of zero and scaling them with unit variance.

Consider a 4-D activation tensor $\mathbf{x} \in \mathbb{R}^{N \times C \times H \times W}$ in a convolutional neural network, where N, C, H, and W represent the batch size, number of channels, height, and width, respectively. BN computes the mini-batch mean (μ_B) and standard deviation (σ_B) over the set $B = \{x_{1:m} : m \in [1, N] \times [1, H] \times [1, W]\}$, where $\mathbf{x}$ is flattened across all dimensions except the channel axis. A small

constant ϵ is included for numerical stability, as shown in Equation 2.6.

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \quad \sigma_B = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 + \epsilon} \tag{2.6}$$

If the samples within the mini-batch come from the same distribution, the transformation $\mathbf{x} \rightarrow \hat{\mathbf{x}}$, as shown in Equation 2.7, produces a normalized distribution with zero mean and unit variance. BN then applies learnable scale (γ) and shift (β) parameters to re-scale the normalized data to a new distribution with mean β and standard deviation γ.

$$\hat{x}_i = \frac{x_i - \mu_B}{\sigma_B} \quad y_i = \gamma\hat{x}_i + \beta \tag{2.7}$$

During inference, rather than using the batch statistics, BN employs a moving average of the mean and variance computed during training. The moving average mean $\bar{\mu}$ and variance $\bar{\sigma}^2$ are calculated as:

$$\bar{\mu} = \alpha\bar{\mu} + (1-\alpha)\mu_B \quad \bar{\sigma}^2 = \alpha\bar{\sigma}^2 + (1-\alpha)\sigma_B^2 \tag{2.8}$$

Here, α is a momentum parameter that controls the update rate of the moving averages. During inference, these moving averages are used to normalize activations as:

$$\hat{x}_i = \frac{x_i - \bar{\mu}}{\sqrt{\bar{\sigma}^2 + \epsilon}} \quad y_i = \gamma\hat{x}_i + \beta \tag{2.9}$$

This ensures consistency across different batch sizes during inference.

Despite its remarkable performance in stabilizing the training of DNNs, BN faces significant limitations related to its dependency on mini-batch size. Specifically, BN's effectiveness diminishes when the size of the mini-batch is small. This occurs because BN relies on accurate estimates of batch statistics (mean and variance) during training, which become less reliable with smaller mini-batches, leading to noisy gradients and unstable updates. This limitation poses a challenge in scenarios where memory constraints or certain applications require smaller mini-batches. To

address this issue, several variants of BN have been proposed, which we will discuss in detail in Section 2.2.2.

2.2.2 Extensions of Batch Normalization to Address Mini-Batch Dependency

To address the mini-batch dependency issue, several extensions of Batch Normalization have been introduced, including Layer Normalization (LN) [4], Instance Normalization (IN) [114], Group Normalization (GN) [124], and Divisive Normalization (DN) [96], Unsupervised Batch Normalization (UBN) [61]. In this section, we adopt the notations from [57] to illustrate that the primary distinction between these methods lies in the specific set over which the sample statistics are computed. Let's consider $i = (i_N, i_C, i_L)$ as a vector indexing the tensor of activations $\mathbf{x} \in \mathbb{R}^{N \times C \times L}$, associated with a convolutional layer where the spatial domain has been flattened. The general normalization, $\mathbf{x} \rightarrow \hat{\mathbf{x}}$, is defined as:

$$v_i = x_i - \mathbb{E}_{B_i}(\mathbf{x}), \quad \hat{x}_i = \frac{v_i}{\sqrt{\mathbb{E}_{B_i}(\mathbf{v}^2) + \epsilon}}, \tag{2.10}$$

where $\mathbb{E}_{B_i}(x)$ denotes the expectation computed over a subset B_i of activations. Similar to BN, the normalized activations can be further adjusted by scaling and shifting using the parameters γ and β. To derive the BN transformation (Equation 2.9) from the general normalization Equation 2.10, it is only necessary to define the appropriate B_i as:

$$B_i = \{j : j_N \in [1, N], j_C \in [i_C], j_L \in [1, L]\}. \tag{2.11}$$

In this case, B_i captures all activations within the same channel i_C across the entire mini-batch and spatial dimensions.

Layer Normalization (LN) [4] adapts BN for architectures like recurrent neural networks (RNNs), where temporal information is critical. Unlike BN, which normalizes across the mini-batch, LN normalizes across features for each training example independently, addressing RNN-specific challenges like varying batch sizes and dependencies on prior time steps. This ensures consistent normalization across all time steps, improving training stability and convergence. LN can be formulated as Equation 2.10 when

$$B_i = \{j : j_N \in [i_N], j_C \in [1, C], j_L \in [1, L]\}. \tag{2.12}$$

LN effectively reduces internal covariate shift in RNNs, enhancing long-range dependency capture and performance in tasks like natural language processing and time-series forecasting. It's also computationally efficient and widely used in modern architectures like transformers [116]. However, LN underperforms in convolutional layers, where local spatial variations are important, as it applies the same normalization across the entire spatial domain, making it less suited for convolutional architectures.

Instance Normalization (IN) [114] extends the ideas of BN and LN, specifically designed for generative models and style transfer. Unlike BN, which normalizes across mini-batches, or LN, which normalizes across all features of a single example, IN normalizes each channel independently for each instance. This helps preserve instance-specific characteristics, making it particularly effective in tasks like image generation and style transfer, where separating content from style is crucial for creative manipulations and high-quality output [52, 140]. IN can be formulated as Equation 2.10 when

$$B_i = \{j : j_N \in [i_N], j_C \in [i_C], j_L \in [1, L]\}. \tag{2.13}$$

However, IN can underperform in tasks like classification or CNN-based image recognition, where capturing correlations between instances is important. Its focus on instance-specific normalization can lead to a loss of shared statistics, limiting its effectiveness in scenarios that benefit from global feature interactions.

Group Normalization (GN) [124] divides channels into smaller groups and computes the mean and variance for each group independently, making it robust to fluctuations in batch size. This is particularly useful in tasks like object detection and segmentation, where small batch sizes are common. GN balances the strengths of LN (G=1) and IN (G=C), providing more stable and effective normalization by ensuring group-specific statistics are representative of the data, leading to improved convergence and generalization. GN can be formulated as Equation 2.10 when

$$B_i = \{j : j_N \in [i_N], j_C \in [i_C], j_L \in [1, L] | \lfloor \frac{j_C}{C/G} \rfloor\} \tag{2.14}$$

However, GN's performance heavily depends on the choice of group size, requiring tuning to optimize results. While it outperforms BN in small-batch scenarios, it may underperform in very deep networks where capturing global batch statistics across all channels is crucial for effective feature learning.

Divisive Normalization (DN) [96] is a biologically inspired technique where each neuron's activity is divided by a weighted combination of its neighbors' activities, offering more dynamic control of activations. Unlike other methods that use simple statistics, DN adjusts activations as follows:

$$v_i = x_i - \mathbb{E}_{A_i}(\mathbf{x}), \quad \hat{x}_i = \frac{v_i}{\sqrt{\mathbb{E}_{B_i}(\mathbf{v}^2) + \rho^2}}, \tag{2.15}$$

where:

$$A_i = \{j \mid d(x_i, x_j) \leq R_A\}, \quad B_i = \{j \mid d(v_i, v_j) \leq R_B\},$$

with d representing the distance between hidden units, ρ the normalizer bias, and R the neighborhood radius. This method enhances decorrelation of neuronal responses, reducing redundancy and improving focus on salient features. DN has shown to improve model robustness and interpretability, particularly in visual tasks. However, DN is computationally intensive, requiring the calculation of weighted sums for neighboring neurons, which can slow down large networks. Additionally, DN may underperform in convolutional networks, where global methods like BN are better at capturing broad data distributions. Its effectiveness also depends on fine-tuning parameters like neighborhood size and weights, adding complexity to model design. Thus, while DN has powerful benefits, its computational cost and complexity limit its broader use.

Unsupervised Batch Normalization (UBN) [61] addresses biased batch statistics in Batch Normalization (BN) when working with small labeled datasets. By incorporating additional unlabeled data from the same distribution to compute batch statistics, UBN reduces the bias introduced by small mini-batches. It is formulated as:

$$B_i = \{j : j_N \in [1, N], j_C \in [i_C], j_L \in [1, L]\} \cup U_i, \tag{2.16}$$

where U_i represents the indices of unlabeled data. This approach enhances the representation of the data distribution, leading to more accurate normalization and stable training without needing changes to the network architecture. However, UBN relies on the assumption that the unlabeled data is from the same distribution as the labeled data; if there is a domain mismatch, the normalization may not generalize effectively.

These techniques represent a significant step forward in overcoming the challenges of mini-batch dependency. Each method offers specific benefits suited to different DNN architectures and

tasks. The choice of technique should be based on the model architecture and the training requirements, with newer methods providing a balance between flexibility and efficiency in training.

2.3 Multi-mode normalization

Multi-mode normalization standardizes activations using statistics from various sources, such as different layers, mini-batches, or feature channels. Several methods have been proposed to enhance this process, including Switchable Normalization (SwitchNorm) [81], Mode Normalization (ModeNorm) [80] and Mixture Normalization (MixNorm) [57]. These techniques address the limitations of BN by overcoming the uniform data distribution assumption, which can hinder performance on diverse datasets. Overall, multi-mode normalization improves the robustness and stability of normalization in DNNs.

Switchable Normalization (SwitchNorm) [81] is an advanced extension of BN that dynamically combines multiple normalization techniques, including BN, LN, and IN, through a set of learnable weights. Unlike BN, which assumes uniform data distribution across mini-batches and can suffer when batch sizes are small or when data distributions are not consistent, SwitchNorm allows the model to adaptively select the most appropriate normalization method for each layer. By leveraging this flexibility, SwitchNorm improves performance across a variety of scenarios, particularly when BN's reliance on mini-batch statistics becomes unreliable, such as in tasks with small batch sizes or non-uniform activations.

For each activation x_i, SwitchNorm alters the normalization process by dynamically adjusting the computation of the batch statistics, as shown in Equation 2.6:

$$\hat{x}_i = \frac{x_i - \sum_{k\in\Omega} w_k \mu_k}{\sqrt{\sum_{k\in\Omega} w'_k \sigma_k^2 + \epsilon}} \quad y_i = \gamma \hat{x}_i + \beta. \tag{2.17}$$

Here, Ω represents a set of statistics estimated using different normalization methods. In the context of SwitchNorm, $\Omega = \{\text{BN}, \text{LN}, \text{IN}\}$, which means that μ_k and σ_k^2 are computed for BN, LN, and IN using the batch B_i as defined in Equations 2.11, 2.12, and 2.13 respectively. The calculations for

these statistics can be expressed as follows:

$$\mu_k = \frac{1}{|B_i|} \sum_{j \in B_i} x_j, \quad \sigma_k^2 = \frac{1}{|B_i|} \sum_{j \in B_i} (x_j - \mu_k)^2. \tag{2.18}$$

Furthermore, w_k and w'_k are importance ratios used to weight the means and variances, respectively. Each w_k and w'_k is a scalar variable constrained to the range $[0, 1]$, satisfying the conditions $\sum_{k \in \Omega} w_k = 1$ and $\sum_{k \in \Omega} w'_k = 1$. The weights w_k can be computed as follows:

$$w_k = \frac{e^{\lambda_k}}{\sum_{z \in \{\text{BN,LN,IN}\}} e^{\lambda_z}}, \quad k \in \{\text{BN, LN, IN}\}, \tag{2.19}$$

where $\lambda_{\text{BN}}, \lambda_{\text{LN}}$, and λ_{IN} are control parameters learned during backpropagation. The weights w'_k are defined similarly, using an additional set of control parameters $\lambda'_{\text{BN}}, \lambda'_{\text{LN}}, \lambda'_{\text{IN}}$.

Let Θ represent the set of network parameters (e.g., filters) and Φ denote the set of control parameters that define the network architecture. In SwitchNorm, the learned parameters are given by $\Phi = \{\lambda_{\text{BN}}, \lambda_{\text{LN}}, \lambda_{\text{IN}}, \lambda'_{\text{BN}}, \lambda'_{\text{LN}}, \lambda'_{\text{IN}}\}$. Training a DNN with SwitchNorm involves minimizing the loss function:

$$\min_{\{\Theta, \Phi\}} \frac{1}{N} \sum_{j=1}^{N} L(y_j, f(x_j; \Theta, \Phi)),$$

where $\{x_j, z_j\}_{j=1}^N$ represents a set of training samples and their corresponding labels. The function $f(x_j; \Theta)$ is the model learned by the DNN to predict z_j. The parameters Θ and Φ are optimized jointly through backpropagation.

SwitchNorm provides a valuable integration of various normalization methods but is limited by its dependence on BN, LN, and IN for parameter estimation. This reliance means it inherits the same limitations as these techniques, particularly in handling non-uniform data distributions, which may undermine its effectiveness in addressing the challenges posed by diverse data conditions.

Mode Normalization (ModeNorm) [80] introduces the concept of "modes" within the data. A mode refers to a dominant pattern or cluster within the data distribution, representing different subpopulations or variations in the input. ModeNorm detects these modes and normalizes the activations based on the statistics of their respective modes, rather than using the entire batch's statistics. This provides a more fine-grained and adaptive normalization process compared to SwitchNorm.

For each activation x_i, ModeNorm adapts the normalization formula as follows:

$$\hat{x}_i = \sum_{k=1}^{K} g_k(x_i) \frac{x_i - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}} \quad y_i = \gamma \hat{x}_i + \beta, \tag{2.20}$$

where $g_k, k \in \{1, ..., K\}$ are gating functions represented by a mixture of experts. For each x_i, $g_k(x_i) \in [0, 1]$ and $\sum_{k=1}^{K} g_k(x_i) = 1$. The estimators for μ_k and σ_k^2 are computed under the weighting from the gating network using the indices B_i:

$$\mu_k = \frac{1}{N_k} \sum_{j \in B_i} g_k(x_j) \cdot x_j \quad \sigma_k^2 = \frac{1}{N_k} \sum_{j \in B_i} g_k(x_j) \cdot (x_j - \mu_k)^2, \tag{2.21}$$

where $N_k = \sum_{j \in B_i} g_k(x_j)$. ModeNorm uses an affine transformation followed by softmax activation to represent the gating networks. When the number of modes $K = 1$, or when the gates collapse to a constant $g_k(x_i) = \text{const}$, ModeNorm reduces to BN. Like BN, during training, ModeNorm normalizes activations using statistics computed from the current batch. During inference, it uses moving averages of mean and variance, as in Equation 2.8, similarly to BN.

ModeNorm helps overcome BN's shortcomings, especially when the data contains multiple modes or clusters that differ significantly. It excels in scenarios with non-uniform data distributions, where BN's global batch statistics may be misleading. However, ModeNorm adds complexity by requiring the identification of modes and calculating separate statistics for each mode, which can increase computational cost and introduce additional hyperparameters. Moreover, its effectiveness depends heavily on the accurate identification of modes, which may be challenging in complex or highly variable datasets, potentially limiting its generalizability in certain tasks.

Mixture Normalization (MixNorm) [57] extends BN by leveraging a probabilistic approach based on Gaussian Mixture Models (GMM). Rather than assuming a single underlying distribution for activations in a mini-batch, MixNorm captures the multimodal nature of data by normalizing each sample based on multiple modes. Each sample is assigned to one of several Gaussian components, enabling a more fine-grained adaptation of normalization to the underlying data distribution. This method improves on the limitations of BN, which can struggle with non-uniform or complex distributions across mini-batches.

In MixNorm, the probability density function p_θ that characterizes the data is modeled as a GMM with K components. The distribution for each sample $\mathbf{x} \in \mathbb{R}^D$ is expressed as:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \lambda_k p(\mathbf{x}|k), \quad \text{s.t. } \forall k : \lambda_k \geq 0, \sum_{k=1}^{K} \lambda_k = 1, \tag{2.22}$$

where λ_k is the mixture coefficient for the k-th component, and $p(\mathbf{x}|k)$ is the Gaussian distribution for the k-th component, given by:

$$p(\mathbf{x}|k) = \frac{1}{(2\pi)^{D/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{(\mathbf{x}-m_k)^T\Sigma_k^{-1}(\mathbf{x}-m_k)}{2}\right), \tag{2.23}$$

with m_k being the mean and Σ_k the covariance matrix of the k-th Gaussian. Considering K components, MixNorm is implemented in two stages:

- Estimation of the mixture model's parameters $\theta = \{\lambda_k, m_k, \Sigma_k : k = 1, \ldots, K\}$ using the Expectation-Maximization (EM) algorithm [22].
- Normalization of each sample based on the estimated parameters and aggregation using posterior probabilities.

For a given activation x_i, the MixNorm transformation is formulated as:

$$\hat{x}_i = \sum_{k=1}^{K} \frac{p(k|x_i)}{\sqrt{\lambda_k}} \cdot \frac{x_i - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}}, \quad y_i = \gamma\hat{x}_i + \beta, \tag{2.24}$$

where $p(k|x_i) = \frac{\lambda_k p(x_i|k)}{\sum_{l=1}^{K} \lambda_l p(x_i|l)}$ represents the probability that x_i belongs to the k-th component. The weighted mean and variance for the k-th component are computed as follows:

$$\mu_k = \sum_{j \in B_i} \frac{p(k|x_j)}{\sum_{l \in B_i} p(k|x_l)} \cdot x_j, \tag{2.25}$$

$$\sigma_k^2 = \sum_{j \in B_i} \frac{p(k|x_j)}{\sum_{l \in B_i} p(k|x_l)} \cdot (x_j - \mu_k)^2, \tag{2.26}$$

MixNorm ensures that each sample is normalized according to the distribution it most likely belongs to, making it highly adaptive to complex, multimodal data distributions. MixNorm extends BN to heterogeneous complex datasets and often yield superior performance in supervised learning tasks. However, they are frequently computationally expensive due to the use EM algorithm.

2.4 Discussion

Activation normalization is a promising approach for addressing slow convergence and training instability in DNNs. BN, a single-mode method, has shown significant success by mitigating the

internal covariate shift issue. However, BN's effectiveness diminishes when mini-batches are small or when the data samples within a batch come from different distributions. To address the small batch size problem, several single-mode alternatives such as LN, IN, GN, DN, and UBN have been introduced.

To handle the challenge of non-uniform data distribution within mini-batches, multi-mode approaches such as SwitchNorm, ModeNorm, and MixNorm have been developed. However, this area is relatively underexplored, and existing methods tend to be computationally expensive, often requiring additional parameters or complex algorithms like EM in MixNorm. In the following Chapter 3, we propose three new multi-mode methods aimed at accelerating DNN training convergence and improving performance by leveraging prior knowledge-driven approaches.

Chapter 3

Context Normalization

3.1 Introduction

In this chapter, we introduce a novel approach to normalization in deep neural networks (DNNs), aimed at improving training efficiency and model performance. The proposed method, Context Normalization, leverages prior knowledge to define "contexts" within the input data—groups of samples with similar characteristics—enabling more efficient normalization and faster convergence compared to traditional methods.

We propose three variants of Context Normalization to address different challenges in training:

- **Context Normalization (CN)**, the base method that identifies and normalizes contexts within the data.

- **Context Normalization - Extended (CN-X)**, which enhances the base method by extending its applicability to more complex data distributions.

- **Adaptive Context Normalization (ACN)**, which further adapts to dynamic variations in data and allows for more flexibility in real-world scenarios.

These methods are validated through extensive experiments in domains such as **image classification**, **image generation**, and **domain adaptation**. In each case, we observe improvements in convergence speed, model stability, and performance, demonstrating the broad applicability and effectiveness of Context Normalization.

The chapter is structured as follows: Section 3.2 introduces the foundational concept of Context Normalization (CN); Section 3.3 describes the extended version, Context Normalization - Extended (CN-X); and Section 3.4 focuses on the adaptive variant, Adaptive Context Normalization (ACN).

3.2 Context Normalization (CN)

CN modifies Equation 2.24 from Mixture Normalization (MN) [57], where the mixture components are treated as modes for normalization. MN employs the Expectation-Maximization (EM) algorithm to estimate the parameters of these mixture components during training. However, EM is computationally expensive and must be applied repeatedly, as the activation distribution shifts with updates to the DNN weights.

Instead of relying on the EM algorithm, we propose normalizing based on "contexts" that are pre-constructed from the input data before DNN training. Each sample in the input data is assigned to a single, unique context, with all samples within the same context sharing similar characteristics. Further details on how these contexts are constructed will be provided in Section 3.5. Each sample belongs to a unique context k. CN ensures that all activations from a sample are associated with the same context k throughout DNN training.

To align with standard representations in the literature 2, let $\mathbf{x} \in \mathbb{R}^{N \times C \times L}$ be an activation tensor, where N is the batch size, C is the number of channels, and $L = H \times W$ represents the flattened spatial dimensions (height H and width W). Each activation is denoted by $\{x_i, k_i\}$, where x_i is the activation and $k_i \in \{1, \ldots, K\}$ is its context identifier, with K being the number of contexts. Each activation x_i is assigned to the context k_i corresponding to the sample that produced it. Since each activation is associated with a unique known context, we have $p(k_i|x_i) = 1$ if x_i belongs to context k_i, and $p(k_i|x_i) = 0$ otherwise. Consequently, Equation 2.24 simplifies to:

$$\hat{x}_i = \frac{1}{\sqrt{\lambda_{k_i}}} . \frac{x_i - \mu_{k_i}}{\sqrt{\sigma^2_{k_i} + \epsilon}} \quad y_i = \gamma_{k_i} \hat{x}_i + \beta_{k_i} \tag{3.1}$$

where λ_{k_i} represents the proportion of samples in the dataset belonging to context k_i. The mean and variance are then defined as follows:

$$\mu_{k_i} = \frac{1}{N_{k_i}} \cdot \sum_{x_i \in \mathbf{x}^{(k_i)}} x_i \tag{3.2}$$

$$\sigma^2_{k_i} = \frac{1}{N_{k_i}} \cdot \sum_{x_i \in \mathbf{x}^{(k_i)}} (x_i - \mu_{k_i})^2 \tag{3.3}$$

where $\mathbf{x}^{(k_i)}$ denotes the subset of $\mathbf{x}$ containing the activations corresponding to context k_i, and N_{k_i} represents the number of elements in $\mathbf{x}^{(k_i)}$. The moving averages of the mean $\bar{\mu}_{k_i}$ and variance

$\bar{\sigma}^2_{k_i}$ are updated with a momentum rate α during training, following the same approach as in BN (see Equation 2.8). These updated statistics are then used to normalize the feature maps during inference:

$$\bar{\mu}_{k_i} = \alpha\bar{\mu}_{k_i} + (1-\alpha)\mu_{k_i} \quad \bar{\sigma}^2_{k_i} = \alpha\bar{\sigma}^2_{k_i} + (1-\alpha)\sigma^2_{k_i} \tag{3.4}$$

In the special case where there is only a single context ($K = 1$), CN reduces to standard BN.

We present the CN transform (Algorithm 1), applied to a set of activations $\mathbf{x}^{(k)}$ of a specific context

Algorithm 1: CN Transform applied to activations of a specific context.

Input : k: context identifier;

$\mathbf{x}^{(k)}$: subset of activations associated with context k;

λ_k: proportion of samples in the dataset assigned to context k;

$\{\gamma_k, \beta_k\}$: learnable parameters;

Output: $\{y_i\} = \mathrm{CN}_{\{\gamma_k,\beta_k\}}(k, \mathbf{x}^{(k)}, \lambda_k)$

$N_k = |\mathbf{x}^{(k)}|$ // *number of elements*

$\mu_k = \frac{1}{N_k} \cdot \sum_{x_i \in \mathbf{x}^{(k)}} x_i$ // *context mean*

$\sigma^2_k = \frac{1}{N_k} \cdot \sum_{x_i \in \mathbf{x}^{(k)}} (x_i - \mu_k)^2$ // *context variance*

for $x_i \in \mathbf{x}^{(k)}$ **do**

  $\hat{x}_i = \frac{1}{\sqrt{\lambda_k}} . \frac{x_i - \mu_k}{\sqrt{\sigma^2_k + \epsilon}}$ // *normalize*

  $y_i = \gamma_k \hat{x}_i + \beta_k$ // *scale and shift*

end

k. CN can be integrated into a neural network to manipulate activations. The scaled and shifted values $y = \{y_i\}$ are passed to other layers, while the normalized activations $\hat{x} = \{\hat{x}_i\}$, internal to CN, have mean 0 and variance 1. Unlike BN, which normalizes across the entire mini-batch, CN normalizes activations within context k. Each $\hat{x}$ is input to a sub-network with $y = \gamma_k\hat{x} + \beta_k$, accelerating training similarly to BN but per context k.

During training, we need to propagate the gradient of loss ℓ through this transformation, as well as compute the gradients with respect to the parameters of CN transform. We use chain rule, as follows (before simplification):

$$
\begin{aligned}
\frac{\partial \ell}{\partial \hat{x}_i} &= \frac{\partial \ell}{\partial y_i} \cdot \gamma_k \\
\frac{\partial \ell}{\partial \sigma_k^2} &= \frac{1}{\sqrt{\lambda_k}} \cdot \sum_{i=1}^{N_k} \frac{\partial \ell}{\partial \hat{x}_i} \cdot (x_i - \mu_k) \cdot \left(-\frac{1}{2}\right) \left(\sigma_k^2 + \epsilon\right)^{-\frac{3}{2}} \\
\frac{\partial \ell}{\partial \mu_k} &= \frac{1}{\sqrt{\lambda_k}} \cdot \left(\sum_{i=1}^{N_k} \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{-1}{\sqrt{\sigma_k^2 + \epsilon}} \right) + \frac{\partial \ell}{\partial \sigma_k^2} \cdot \frac{\sum_{i=1}^{N_k} -2(x_i - \mu_k)}{N_k} \\
\frac{\partial \ell}{\partial x_i} &= \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{1}{\sqrt{\lambda_k}} \cdot \frac{1}{\sqrt{\sigma_k^2 + \epsilon}} + \frac{\partial \ell}{\partial \sigma_k^2} \cdot \frac{2(x_i - \mu_k)}{N_k} + \frac{\partial \ell}{\partial \mu_k} \cdot \frac{1}{N_k} \\
\frac{\partial \ell}{\partial \gamma_k} &= \sum_{i=1}^{N_k} \frac{\partial \ell}{\partial y_i} \cdot \hat{x}_i \\
\frac{\partial \ell}{\partial \beta_k} &= \sum_{i=1}^{N_k} \frac{\partial \ell}{\partial y_i}
\end{aligned}
$$

The CN transform is a differentiable operation that introduces context-normalized activations into the neural network. This reduces internal covariate shift, accelerating training. Additionally, the learned affine transform, like in BN, allows CN to represent the identity transformation, preserving the neural network's capacity.

To Context-Normalize a deep neural network, we define activations with their context identifiers $\{x_i, k_i\}$ and apply the CN transform on each based on its context, as outlined in Algorithm 1. Layers that previously received $\mathbf{x}^{(k)}$ (activations for context k) now take $\mathrm{CN}(k, \mathbf{x}^{(k)}, \lambda_k)$. This context-based normalization in mini-batches supports efficient training but isn't needed during inference; like BN, the output should depend deterministically on the input. After training, activations are normalized using:

$$
\hat{y}_i = \gamma_{k_i} \cdot \frac{1}{\sqrt{\lambda_{k_i}}} \cdot \frac{x_i - \bar{\mu}_{k_i}}{\sqrt{\bar{\sigma}_{k_i}^2 + \epsilon}} + \beta_{k_i} \tag{3.5}
$$

Here, population statistics replace context-specific ones. Since the means and variances are fixed at inference, normalization reduces to a linear transform for each activation. This can be combined with the scaling by γ_k and shift by β_k, resulting in a single linear transform replacing $\mathrm{CN}(k, \mathbf{x}^{(k)}, \lambda_k)$. Algorithm 2 details the training process for context-normalized deep neural networks.

Limitation. CN divides the mini-batch into multiple subgroups based on predefined contexts, estimates the mean and variance for each subgroup, and normalizes the activations using the

Algorithm 2: Training a Context-Normalized Network.

Input : Net: Deep neural network with trainable parameters Θ;

K: number of contexts;

$\{x_i, k_i\}$, where $k_i \in \{1, \ldots, K\}$: activations and corresponding context;

$\{\lambda_k\}_{k=1}^{K}$: proportion of samples assigned to each context k;

$\{\gamma_k, \beta_k\}_{k=1}^{K}$: learnable parameters;

α: momentum;

Output: Context-normalized network for inference, $\text{Net}_{\text{CN}}^{\text{inf}}$

$\text{Net}_{\text{CN}}^{\text{tr}} \leftarrow$ Net // *Trainig CN deep neural network*

for $k \leftarrow 1$ *to* K **do**

  Construct $\mathbf{x}^{(k)}$ with all activations for context k

  Add transformation $y = \text{CN}_{\{\gamma_k, \beta_k\}}(k, \mathbf{x}^{(k)}, \lambda_k)$ to $\text{Net}_{\text{CN}}^{\text{tr}}$ (Algorithm 1)

  Replace the input $\mathbf{x}^{(k)}$ with $\mathbf{y}^{(k)}$ in each layer of $\text{Net}_{\text{CN}}^{\text{tr}}$

  $\bar{\mu}_k = \alpha\bar{\mu}_k + (1-\alpha)\mu_k \quad \bar{\sigma}_k^2 = \alpha\bar{\sigma}_k^2 + (1-\alpha)\sigma_k^2$

end

Train $\text{Net}_{\text{CN}}^{\text{tr}}$ to optimize the parameters $\Theta \cup \{\gamma_k, \beta_k\}_{k=1}^{K}$

$\text{Net}_{\text{CN}}^{\text{inf}} \leftarrow \text{Net}_{\text{CN}}^{\text{tr}}$ // *Inference CN deep neural network with frozen parameters*

for $k \leftarrow 1$ *to* K **do**

  Construct $\mathbf{x}^{(k)}$ with all activations for context k

  for $x_i \in \mathbf{x}^{(k)}$ **do**

    $\hat{x}_i = \frac{1}{\sqrt{\lambda_k}} \cdot \frac{x_i - \bar{\mu}_k}{\sqrt{\bar{\sigma}_k^2 + \epsilon}}$ // *normalize*

    $y_i = \gamma_k \hat{x}_i + \beta_k$ // *scale and shift*

  end

  Replace the input $\mathbf{x}^{(k)}$ with $\mathbf{y}^{(k)}$ in each layer of $\text{Net}_{\text{CN}}^{\text{inf}}$

end

corresponding parameters. However, if a subgroup contains too few elements, the parameter estimates may become unreliable, causing CN to face the same issues as BN with small mini-batch sizes. To address this limitation, we propose an extension of CN, which we will discuss in Section 3.3.

3.3 Context Normalization - Extended (CN-X)

CN-X is an enhanced version of CN designed for more robust context parameter estimation. While CN estimates the normalization parameters (mean and variance) directly from activations within each context, CN-X instead learns these parameters as trainable weights of the neural network. These parameters are updated during backpropagation, making them more flexible and accurate over time. For each context k, we define the parameter set $\theta_k = \{\mu_k, \sigma_k^2\}$, where μ_k and σ_k^2 are initialized randomly, with the constraint that $\sigma_k^2 \geq 0$. To normalize activations in context k,

Algorithm 3: CN-X Transform applied to activations of a specific context.

Input : k: context identifier;

$\mathbf{x}^{(k)}$: subset of activations associated with context k;

λ_k: proportion of samples in the dataset assigned to context k;

$\phi_k = \{\mu_k, \sigma_k^2\}$: normalization parameters;

$\{\gamma_k, \beta_k\}$: learnable parameters;

Output: $\{y_i\} = \text{CN-X}_{\{\phi_k, \gamma_k, \beta_k\}}(k, \mathbf{x}^{(k)}, \lambda_k)$

for $x_i \in \mathbf{x}^{(k)}$ **do**

  $\hat{x}_i = \frac{1}{\sqrt{\lambda_k}} . \frac{x_i - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}}$ // *normalize*

  $y_i = \gamma_k \hat{x}_i + \beta_k$ // *scale and shift*

end

represented by $\mathbf{x}^{(k)}$, Algorithm 1 is adapted to produce Algorithm 3. In this modified version, the normalization parameters θ_k are provided as inputs, and the normalization operation remains unchanged. However, unlike in CN, where the parameters are estimated directly from $\mathbf{x}^{(k)}$, in CN-X these parameters are learned through the network's training process. Algorithm 4 outlines the process for training a neural network with CN-X. Let Θ represent the neural network parameters, and $\Phi = \{\phi_k\}_{k=1}^K$, where $\phi_k = \{\mu_k, \sigma_k^2\}$, denote the set of learnable normalization parameters. The objective is to minimize the loss function:

$$\min_{\Theta, \Phi} \frac{1}{N} \sum_{j=1}^{N} \ell(z_j, f(x_j; \Theta, \Phi)),$$

Algorithm 4: Training a Context-Normalized Extended Network.

Input : Net: Deep neural network with trainable parameters Θ;

K: number of contexts;

$\{x_i, k_i\}$, where $k_i \in \{1, \ldots, K\}$: activations and corresponding context;

$\{\lambda_k\}_{k=1}^{K}$: proportion of samples assigned to each context k;

$\{\gamma_k, \beta_k\}_{k=1}^{K}$: learnable parameters;

α: momentum;

Output: Context-normalized Extended network for inference, $\text{Net}^{\text{inf}}_{\text{CN-X}}$

Random initialize $\phi_k = \{\mu_k, \sigma_k^2\}$, where $k \in \{1, ..., K\}$ // *initialize normalization parameters*

$\text{Net}^{\text{tr}}_{\text{CN-X}} \leftarrow$ Net // *Trainig CN-X deep neural network*

for $k \leftarrow 1$ *to* K **do**

  Construct $\mathbf{x}^{(k)}$ with all activations for context k

  Add transformation $y = \text{CN-X}_{\{\phi_k, \gamma_k, \beta_k\}}(k, \mathbf{x}^{(k)}, \lambda_k)$ to $\text{Net}^{\text{tr}}_{\text{CN-X}}$ (Algorithm 1)

  Replace the input $\mathbf{x}^{(k)}$ with $\mathbf{y}^{(k)}$ in each layer of $\text{Net}^{\text{tr}}_{\text{CN-X}}$

end

Train $\text{Net}^{\text{tr}}_{\text{CN-X}}$ to optimize the parameters $\Theta \cup \{\phi_k, \gamma_k, \beta_k\}_{k=1}^{K}$

$\text{Net}^{\text{inf}}_{\text{CN-X}} \leftarrow \text{Net}^{\text{tr}}_{\text{CN-X}}$ // *Inference CN-X deep neural network with frozen parameters*

for $k \leftarrow 1$ *to* K **do**

  Construct $\mathbf{x}^{(k)}$ with all activations for context k

  for $x_i \in \mathbf{x}^{(k)}$ **do**

    $\hat{x}_i = \frac{1}{\sqrt{\lambda_k}} \cdot \frac{x_i - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}}$ // *normalize*

    $y_i = \gamma_k \hat{x}_i + \beta_k$ // *scale and shift*

  end

  Replace the input $\mathbf{x}^{(k)}$ with $\mathbf{y}^{(k)}$ in each layer of $\text{Net}^{\text{inf}}_{\text{CN-X}}$

end

where $\{x_j, z_j\}_{j=1}^{N}$ is the set of training samples and labels, with each sample belonging to a single context $k_j \in \{1, \ldots, K\}$. The function $f(x_j; \Theta, \Phi)$ is learned by the network to predict the output y_j. Both Θ and Φ are optimized jointly via backpropagation.

This approach differs from previous methods like BN and CN, where normalization parameters Φ are often treated as separate network modules (e.g., scale and shift) and not essential for normalization. In CN-X, Φ is learned directly during training, contributing to minimizing the loss function. Since the normalization parameters are not estimated from the activations, even small context sizes in a mini-batch do not negatively impact the learned parameters, as they are updated as part of the network's weights.

Similar to CN, in CN-X, we need to propagate the gradient of the loss function ℓ through the transformation during training, while also computing the gradients with respect to the parameters of the CN-X transformation. This is achieved by applying the chain rule, as outlined below (prior to simplification):

$$
\begin{aligned}
\frac{\partial \ell}{\partial \hat{x}_i} &= \frac{\partial \ell}{\partial y_i} \cdot \gamma_k \\
\frac{\partial \ell}{\partial \sigma_k^2} &= \frac{1}{\sqrt{\lambda_k}} \cdot \sum_{i=1}^{N_k} \frac{\partial \ell}{\partial \hat{x}_i} \cdot (x_i - \mu_k) \cdot \left(-\frac{1}{2}\right) \left(\sigma_k^2 + \epsilon\right)^{-\frac{3}{2}} \\
\frac{\partial \ell}{\partial \mu_k} &= \frac{1}{\sqrt{\lambda_k}} \cdot \left(\sum_{i=1}^{N_k} \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{-1}{\sqrt{\sigma_k^2 + \epsilon}} \right) \\
\frac{\partial \ell}{\partial x_i} &= \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{1}{\sqrt{\lambda_k}} \cdot \frac{1}{\sqrt{\sigma_k^2 + \epsilon}} \\
\frac{\partial \ell}{\partial \gamma_k} &= \sum_{i=1}^{N_k} \frac{\partial \ell}{\partial y_i} \cdot \hat{x}_i \\
\frac{\partial \ell}{\partial \beta_k} &= \sum_{i=1}^{N_k} \frac{\partial \ell}{\partial y_i}
\end{aligned}
$$

Limitations. CN-X methods rely on predefined contexts within the input dataset for normalization. In domains where constructing these contexts is challenging, such approaches become difficult to apply effectively. To address this limitation, we propose Adaptive Context Normalization (ACN), a method that retains the strengths of both CN-X and CN without the need for predefined contexts. We will elaborate on ACN in Section 3.4.

3.4 Adaptive Context Normalization (ACN)

In ACN, we shift our focus from predefining contexts within the input dataset to dynamically constructing them during the training of the neural network. Unlike CN-X and CN, where inputs are represented as (x_i, k_i)—indicating predefined contexts—ACN simplifies this representation to just x_i. ACN only requires the specification of the number of contexts, K, to be created during the normalization process, akin to clustering algorithms that use a predefined number of clusters. However, instead of relying on prior knowledge or fixed clusters, ACN allows the neural network to autonomously discover a latent space of activations that adheres to a GMM. During training, ACN incrementally clusters neuron activations without predefined partitions, enabling the model to adapt to task-specific challenges without prior cluster information. This flexibility permits the neural network to explore and adapt to the underlying patterns in the data independently. Since the specific context for each activation is not predetermined, ACN utilizes Equation 2.24 to normalize across all contexts. Unlike traditional methods such as MN, where parameters are often fixed, ACN learns the parameters of these contexts as neural network weights during backpropagation. This approach eliminates the need for computationally intensive algorithms like EM, enhancing efficiency in the training process.

The GMM parameters $\theta = \{\lambda_k, m_k, \Sigma_k : k = 1, \ldots, K\}$ are optimized in alignment with the target task. Algorithm 5 outlines the training procedure of a deep neural network using ACN as the normalization method. Initially, the GMM parameters are randomly initialized, ensuring that $\sum_{k=1}^{K} \lambda_k = 1$ is maintained throughout training. This integration allows the GMM parameter estimation to become a dynamic part of the neural network, offering a more adaptive approach. Unlike methods like MN that rely on the EM algorithm—which cannot efficiently track changes in the activation distribution due to its high computational cost—this approach continuously updates the GMM parameters based on shifts in the activation distribution. As the two approaches (CN and CN-X), in ACN we need to propagate the gradient of the loss function ℓ through the transformation during training. This is achieved by applying the chain rule, as outlined below (prior to

Algorithm 5: Training a Adaptive Context-Normalized Network.

Input : Net: Deep neural network with trainable parameters Θ;

K: number of contexts;

$\{x_i\}$: set of activations;

$\{\gamma_k, \beta_k\}_{k=1}^{K}$: learnable parameters;

α: momentum;

Output: Context-normalized Extended network for inference, $\text{Net}_{\text{ACN}}^{\text{inf}}$

Initialize the parameters for each context as follows:

$\theta_k = \{\lambda_k, \mu_k, \Sigma_k\}$ for $k \in \{1, ..., K\}$, subject to the condition that $\sum_{k=1}^{K} \lambda_k = 1$

for $x_i \in \mathbf{x}$ **do**

  Add transformation $\hat{x}_i$ using Equation 2.24

  Modify each layer in Net_{ACN}^{tr} with input x_i to take $\hat{x}_i$ instead

end

Train $\text{Net}_{\text{ACN}}^{\text{tr}}$ to optimize the parameters $\Theta \cup \{\theta_k, \gamma_k, \beta_k\}_{k=1}^{K}$

$\text{Net}_{\text{ACN}}^{\text{inf}} \leftarrow \text{Net}_{\text{ACN}}^{\text{tr}}$ // *Inference ACN deep neural network with frozen parameters*

for $x_i \in \mathbf{x}$ **do**

  $\hat{x}_i = \sum_{k=1}^{K} \frac{p(k|x_i)}{\sqrt{\lambda_k}} \left(\frac{x_i - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}} \right)$ // *normalize*

  $y_i = \gamma_k \hat{x}_i + \beta_k$ // *scale and shift*

end

Replace the input $\mathbf{x}$ with $\mathbf{y}$ in each layer of $\text{Net}_{\text{ACN}}^{\text{inf}}$

simplification):

$$\frac{\partial \ell}{\partial \hat{x}_i} = \frac{\partial \ell}{\partial y_i} \cdot \gamma_k$$

$$\frac{\partial \ell}{\partial \sigma_k^2} = -\frac{1}{2(\sigma_k^2+\epsilon)^{3/2}} \sum_{i=1}^{N} \frac{\partial \ell}{\partial x_i} \cdot \frac{p(k|x_i)}{\sqrt{\lambda_k}} \cdot (x_i - \mu_k)$$

$$\frac{\partial \ell}{\partial \mu_k} = -\sum_{i=1}^{N} \frac{\partial \ell}{\partial x_i} \cdot \frac{p(k|x_i)}{\sqrt{\lambda_k}} \cdot \frac{1}{\sqrt{\sigma_k^2+\epsilon}} + \frac{\partial \ell}{\partial \sigma_k^2} \cdot \left(-2 \sum_{i=1}^{N} \frac{p(k|x_i)}{\sum_{l=1}^{K} p(l|x_i)} \cdot (x_i - \mu_k) \right)$$

$$\frac{\partial \ell}{\partial p(k|x_i)} = \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{1}{\lambda_k} \cdot \frac{x_i - \mu_k}{\sqrt{\sigma_k^2+\epsilon}} + \frac{\partial \ell}{\partial \sigma_k^2} \cdot \frac{\sum_{l=1}^{K} p(l|x_i) - p(k|x_i)}{(\sum_{l=1}^{K} p(l|x_i))^2} + \frac{\partial \ell}{\partial \mu_k} \cdot \frac{\sum_{l=1}^{K} p(l|x_i) - p(k|x_i)}{(\sum_{l=1}^{K} p(l|x_i))^2} \cdot x_i$$

$$\frac{\partial \ell}{\partial p(x_i|k)} = \frac{\partial \ell}{\partial p(k|x_i)} \cdot \frac{\lambda_k (\sum_{l=1}^{K} p(x_i|l) - p(x_i|k))}{(\sum_{l=1}^{K} p(x_i|l))^2}$$

$$\frac{\partial \ell}{\partial m_k} = \sum_{i=1}^{N} \frac{\partial \ell}{\partial p(x_i|k)} \cdot \frac{1}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \cdot (\Sigma_k^{-1}(x_i - m_k)) \exp\left(\frac{(x_i - m_k)^T \Sigma_k^{-1} (x_i - m_k)}{2}\right)$$

$$\frac{\partial \ell}{\partial \Sigma_k} = \sum_{i=1}^{N} \frac{\partial \ell}{\partial p(x_i|k)} \cdot \frac{1}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \cdot \left(\frac{1}{2} \Sigma_k^{-1} (x_i - m_k)(x_i - m_k)^T \Sigma_k^{-1}\right) \cdot \exp\left(\frac{(x_i - m_k)^T \Sigma_k^{-1} (x_i - m_k)}{2}\right)$$

$$\frac{\partial \ell}{\partial x_i} = \frac{\partial \ell}{\partial \hat{x}_i} \cdot \sum_{k=1}^{K} \frac{p(k|x_i)}{\sqrt{\lambda_k}} \cdot \frac{1}{\sqrt{\sigma_k^2+\epsilon}} + \frac{\partial \ell}{\partial \sigma_k^2} \cdot \frac{p(k|x_i)}{\sum_{l=1}^{K} p(l|x_i)} \cdot 2(x_i - \mu_k) + \frac{\partial \ell}{\partial p(x_i|k)} \cdot$$
$$+ \frac{\partial}{\partial \mu_k} \cdot \frac{p(k|x_i)}{\sum_{l=1}^{K} p(l|x_i)} \frac{1}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \cdot (-\Sigma_k^{-1}(x_i - m_k)) \exp\left(-\frac{1}{2}(x_i - m_k)^T \Sigma_k^{-1} (x_i - m_k)\right)$$
$$+ \frac{\partial \ell}{\partial p(x_i|k)} \cdot \frac{\lambda_k [\frac{\partial p(x_i|k)}{\partial x_i} \sum_{l=1}^{K} \lambda_k p(x_i|l) - p(x_i|k) \sum_{l=1}^{K} \lambda_l \frac{\partial p(x_i|l)}{\partial x_i}]}{(\sum_{l=1}^{K} \lambda_l p(x_i|l))^2}$$

$$\frac{\partial \ell}{\partial \gamma_k} = \sum_{i=1}^{N} \frac{\partial \ell}{\partial y_i} \cdot \hat{x}_i$$

$$\frac{\partial \ell}{\partial \beta_k} = \sum_{i=1}^{N} \frac{\partial \ell}{\partial y_i}$$

The ACN is a differentiable operation that integrates context-sensitive, normalized activations directly into the neural network. This method is particularly advantageous for scenarios involving multi-modal data distributions, as it unifies normalization across multiple modes without requiring complex, separate algorithms for estimating mode-specific parameters. Instead, ACN dynamically adapts its normalization based on context.

In this approach, we use MN as a baseline; however, ACN is not limited to MN and can be gener-

alized to other normalization techniques, such as ModeNorm. The key advantage lies in how ACN enables the model to learn context-relevant parameters, which adapt based on the activation distributions that shift throughout training as the network's weights are updated via backpropagation. By leveraging adaptive context normalization, the method allows for smoother transitions and better performance across different data modes, ensuring more efficient parameterization without the need for additional heavy computations during training. This flexibility makes ACN an appealing approach for tasks where data has varying distributions or requires context-sensitive handling.

Table 3.1 shows that proposed methods (CN, CN-X, ACN), particularly when context construction is well-defined, outperform traditional normalization techniques in various tasks. This demonstrates the potential of context-driven approaches to enhance model performance, handle non-uniform data distributions more effectively, and speed up convergence during training.

3.5 Results

In this section, we present several applications where we compare traditional normalization techniques (see Section 2) with our proposed normalization methods (see Section 3). These comparisons are demonstrated across various tasks, including image classification (Section 3.5.1), domain adaptation (Section 3.5.2), and image generation (Section 3.5.3). We utilize several well-known benchmark datasets that are widely recognized within the classification community:

- **CIFAR-10:** A dataset with 50,000 training images and 10,000 test images, each of size 32×32 pixels, distributed across 10 classes [65].
- **Oxford-IIIT Pet:** A dataset containing images of 37 different breeds of cats and dogs, with approximately 200 images per breed [91].
- **CIFAR-100:** Derived from the Tiny Images dataset, it consists of 50,000 training images and 10,000 test images of size 32×32, divided into 100 classes grouped into 20 superclasses [64].

Normalization Method	Context Learning	Computational Complexity	Flexibility
BN	No context learning; uses batch statistics.	Moderate; relies on batch statistics.	Low; depends on fixed batch-based normalization.
LN	No context learning; normalizes across features.	Low; operates on a per-sample basis.	Moderate; works well with smaller batch sizes.
IN	No context learning; normalizes across instances.	Low; operates on per-instance basis.	High; useful for tasks with small batch sizes.
GN	No context learning; normalizes across groups.	Moderate; handles grouped channels.	Moderate; adapts based on groupings.
SwitchNorm	Dynamically combines multiple normalization techniques (BN, LN, IN) using learned weights.	High; requires combining and selecting normalization methods.	High; adaptive to different types of activations.
ModeNorm	No context learning; normalizes using modes of activations identified during training.	High; requires mode identification and adaptive statistics.	Very High; adaptive to mode-specific distributions.
MixNorm	No context learning; normalizes using Gaussian Mixture Models (GMM).	High; requires Expectation-Maximization (EM) for parameter estimation.	High; adapts to multimodal data distributions.
CN	Predefined contexts based on the input data.	Low; efficient with predefined contexts.	Moderate; fixed context structure.
CN-X	Contexts learned as trainable parameters during training.	Moderate; learns parameters during training.	High; flexible context learning.
ACN	Contexts dynamically learned via GMM during training.	Low; eliminates need for EM.	Very High; learns context adaptively during training.

Table 3.1: Comparison of Normalization Techniques

- **Tiny ImageNet:** A dataset that is a reduced version of the ImageNet dataset [23], containing 200 classes with 500 training images and 50 test images per class [67].
- **MNIST digits:** Contains 70,000 grayscale images of size 28×28 representing the 10 digits, with around 6,000 training images and 1,000 testing images per class [68].
- **SVHN:** A challenging dataset with over 600,000 digit images, focusing on recognizing digits and numbers in natural scene images [104].

For applying CN and CN-X, we will use three approaches to build contexts: (i) applying the k-means algorithm to create clusters and using these clusters as contexts, (ii) utilizing superclasses, which are groups of classes, as contexts, and (iii) treating each domain in domain adaptation as a separate context.

3.5.1 Image Classification

To evaluate our normalization methods (CN, CN-X, and ACN) against traditional normalization techniques (BN, LN, MixNorm, and ModeNorm) in image classification tasks, we employ the DenseNet architecture [50], varying the number of layers to create two distinct models: a shallow model with 40 layers (DenseNet-40) and a deeper model with 100 layers (DenseNet-100).
DenseNet employs BN as the normalization layer. We create a corresponding DenseNet model for each normalization technique (LN, MixNorm, ModeNorm, CN, CN-X, and ACN) by replacing the BN layers with the specific normalization method.
In the first experiment, detailed in the section "Building Custom Contexts", we will employ the k-means algorithm to generate clusters that will act as contexts for CN and CN-X, utilizing the Oxford IIIT Pet, CIFAR-10, CIFAR-100, and Tiny ImageNet datasets. In the second experiment, outlined in the section "Leveraging Predefined Contexts", we will utilize the superclass structure (groups of classes) within the Oxford-IIIT Pet and CIFAR-100 datasets as contexts.

Building Custom Contexts

In this study, we assume that the underlying structure of the dataset is not well understood, and there is no clear prior knowledge regarding the contextual relationships within the data. To address this, we need to establish these contexts before training our neural networks, specifically DenseNet-40 and DenseNet-100, for both CN and CN-X normalization techniques. To define the contexts, we employ the k-means clustering algorithm, treating the resulting clusters as distinct contexts. We conduct multiple experiments by varying the number of contexts K, using values of 2, 3, 4, 6, and 8. For a fair comparison, we maintain consistency in the number of contexts across different methods, ensuring that the same K value corresponds to the number of mixture components in MixNorm and the number of modes in ModeNorm. The models are trained for 200 epochs with a batch size of 64, utilizing Nesterov's accelerated gradient [8]. The learning rate is initially set to 0.1 and is reduced by a factor of 10 at 50% and 75% of the total training epochs. Additionally, weight decay is fixed at 10^{-4} and momentum at 0.9.

Table 3.2 presents the performance comparison of CN, CN-X, and ACN on a shallow neural network (DenseNet-40), while Table 3.3 highlights their effectiveness on a deeper network (DenseNet-100). Across all datasets, which vary in complexity based on the number of classes, our proposed method consistently achieves higher average accuracy. This improvement is particularly noticeable with CN-X. Additionally, when varying the number of contexts (2, 4, 6, and 8), the performance difference remains minimal, suggesting that a large number of clusters is not necessary to achieve optimal performance. Figure 3.3 demonstrates that CN, CN-X, and ACN achieve superior convergence compared to traditional methods such as BN, LN, MixNorm, and ModeNorm. The observed acceleration in convergence, illustrated in Figure 3.3, alongside the improved performance metrics presented in Tables 3.2 and 3.3, indicates that our proposed method can effectively serve as a layer to enhance model performance and accelerate convergence, even when prior knowledge of the datasets is limited. In such cases, the k-means algorithm can be employed to generate clusters, which can then be used as contexts for CN and CN-X.

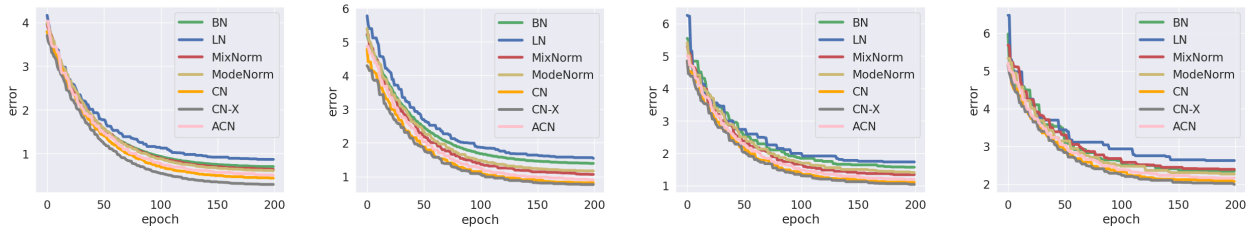
Conversely, when we have a thorough understanding of the dataset and the contexts are well-defined, there is no need to apply k-means clustering; instead, we can directly utilize predefined contexts. This approach will be elaborated upon in the following section.

method	**CIFAR-10**	**Oxford-IIIT Pet**	**CIFAR-100**	**Tiny ImageNet**
BN	92.07	75.63	71.35	52.20
LN	84.65	66.12	58.34	47.20
MixNorm-2	93.10	74.34	73.23	53.20
MixNorm-4	93.60	75.67	73.40	53.24
MixNorm-6	93.60	75.65	73.47	53.18
MixNorm-8	92.62	75.80	73.47	53.67
ModeNorm-2	93.32	75.87	72.90	53.16
ModeNorm-4	93.65	75.84	73.43	54.12
ModeNorm-6	93.68	75.97	73.45	54.18
ModeNorm-8	93.68	76.02	73.27	54.18
CN-2	93.87	75.98	73.88	54.15
CN-4	93.98	76.12	74.10	54.21
CN-6	93.98	76.22	74.10	54.30
CN-8	94.01	76.37	74.12	54.30
CN-X-2	94.06	75.34	73.99	54.23
CN-X-4	94.05	76.23	74.34	55.12
CN-X-6	94.13	76.35	74.23	55.09
CN-X-8	94.54	76.35	74.78	55.26
ACN-2	92.65	75.76	73.77	53.98
ACN-4	93.67	75.87	73.88	54.01
ACN-6	93.89	75.90	74.01	54.23
ACN-8	94.13	75.90	74.01	54.36

Table 3.2: Performance (accuracy %) of DenseNet-40 on CIFAR-10, Oxford-IIIT Pet, CIFAR-100, and Tiny ImageNet. Contexts for CN and CN-X are built using k-means clusters. "2, 3, 4, 8" represent mixture components, modes, and contexts for MixNorm, ModeNorm, and the proposed CN, CN-X, and ACN methods.

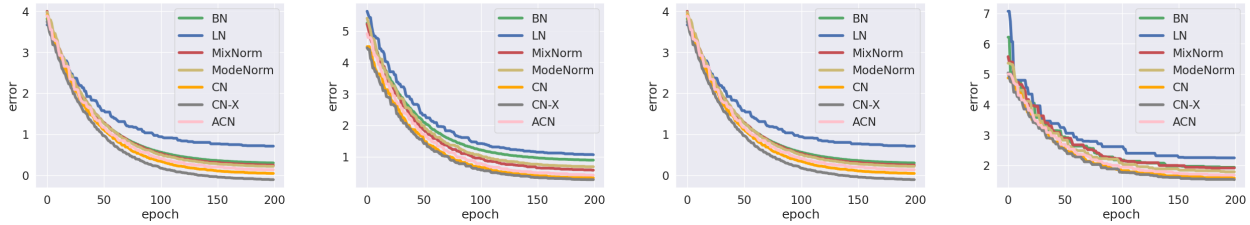
method	**CIFAR-10**	**Oxford-IIIT Pet**	**CIFAR-100**	**Tiny ImageNet**
BN	94.10	76.28	73.32	55.12
LN	85.20	66.34	60.10	47.53
MixNorm-2	94.54	76.67	74.12	55.67
MixNorm-4	94.56	76.73	74.32	55.56
MixNorm-6	94.56	76.75	74.67	55.70
MixNorm-8	95.01	76.87	74.72	55.74
ModeNorm-2	94.65	76.87	74.21	54.76
ModeNorm-4	94.67	76.84	74.34	55.01
ModeNorm-6	94.74	76.89	74.52	55.12
ModeNorm-8	94.74	76.89	74.57	55.12
CN-2	95.10	76.12	74.67	55.26
CN-4	95.76	76.92	74.72	55.17
CN-6	95.76	76.92	74.77	55.78
CN-8	95.67	76.93	74.77	55.98
CN-X-2	95.56	76.67	75.01	55.23
CN-X-4	95.76	76.87	75.10	55.76
CN-X-6	95.87	76.87	75.10	55.78
CN-X-8	96.12	77.01	75.21	55.97
ACN-2	94.76	76.67	74.78	55.22
ACN-4	94.76	76.87	74.88	55.43
ACN-6	94.87	76.89	75.10	55.88
ACN-8	95.10	76.89	75.21	55.89

Table 3.3: Performance (accuracy %) of DenseNet-100 on CIFAR-10, Oxford-IIIT Pet, CIFAR-100, and Tiny ImageNet. Contexts for CN and CN-X are built using k-means clusters. "2, 4, 6, 8" represent mixture components, modes, and contexts for MixNorm, ModeNorm, and the proposed CN, CN-X, and ACN methods.



(a) CIFAR-10 (b) Oxford-IIIT Pet (c) CIFAR-100 (d) Tiny ImageNet

Figure 3.1: DenseNet-40



(a) CIFAR-10 (b) Oxford-IIIT Pet (c) CIFAR-100 (d) Tiny ImageNet

Figure 3.2: DenseNet-100

Figure 3.3: Training Error Trends for DenseNet-40 and DenseNet-100 with Various Normalization Layers. The MixNorm, ModeNorm, CN, CN-X, and ACN methods are implemented using $K = 8$.

Leveraging Predefined Contexts

Some datasets, such as Oxford-IIIT Pet and CIFAR-100, not only have a hierarchical structure of classes but also include superclasses that group similar classes together. For instance, in the Oxford-IIIT Pet dataset, various breeds of dogs and cats can be categorized into two superclasses: "dog" and "cat". Similarly, CIFAR-100 contains 20 distinct superclasses. Rather than applying the k-means algorithm to create clusters for use as contexts, we can leverage these existing superclasses as contextual representations.

In this experiment, we employ the same models as in the previous section, specifically DenseNet-40 and DenseNet-100, to evaluate the evolution of accuracy on the CIFAR-100 and Oxford-IIIT Pet datasets. We utilize the superclasses as contexts and implement normalization layers CN, CN-X, and ACN. The goal is to assess whether a deeper understanding of our dataset, achieved by constructing contexts, yields improved performance compared to relying on predefined contexts (superclasses) present in the datasets. Tables 3.4 and 3.5 illustrate the significant impact that well-defined con-

Oxford-IIIT Pet (K=2)						
model	25 epochs	50 epochs	75 epochs	100 epochs	150 epochs	200 epochs
CN	75.43	76.86	76.88	77.34	78.43	79.26
CN-X	76.12	76.77	77.98	78.66	80.02	80.98
ACN	72.34	72.56	73.10	74.22	74.90	76.13
CIFAR-100 (K=20)						
model	25 epochs	50 epochs	75 epochs	100 epochs	150 epochs	200 epochs
CN	73.88	74.21	74.89	75.10	76.53	77.67
CN-X	74.21	75.10	75.67	77.45	78.54	79.78
ACN	72.34	72.67	74.32	74.32	74.56	74.60

Table 3.4: Evolution of Accuracy with DenseNet-40 Utilizing Superclasses as Contexts on the Oxford-IIIT Pet and CIFAR-100 Datasets.

texts have on the performance of CN and CN-X. Notably, when utilizing superclasses as contexts, we achieve comparable performance in approximately 25 epochs, in contrast to the 200 epochs re-

Oxford-IIIT Pet (K=2)						
model	25 epochs	50 epochs	75 epochs	100 epochs	150 epochs	200 epochs
CN	75.43	75.67	76.98	77.89	79.34	80.23
CN-X	76.54	77.87	79.78	81.23	81.23	82.02
ACN	73.02	74.32	75.43	77.02	77.32	77.85
CIFAR-100 (K=20)						
model	25 epochs	50 epochs	75 epochs	100 epochs	150 epochs	200 epochs
CN	74.21	74.56	76.78	78.22	78.22	79.34
CN-X	73.56	75.43	75.78	79.34	79.89	81.02
ACN	73.21	73.76	75.11	76.21	76.21	76.32

Table 3.5: Evolution of Accuracy with DenseNet-100 Utilizing Superclasses as Contexts on the Oxford-IIIT Pet and CIFAR-100 Datasets.

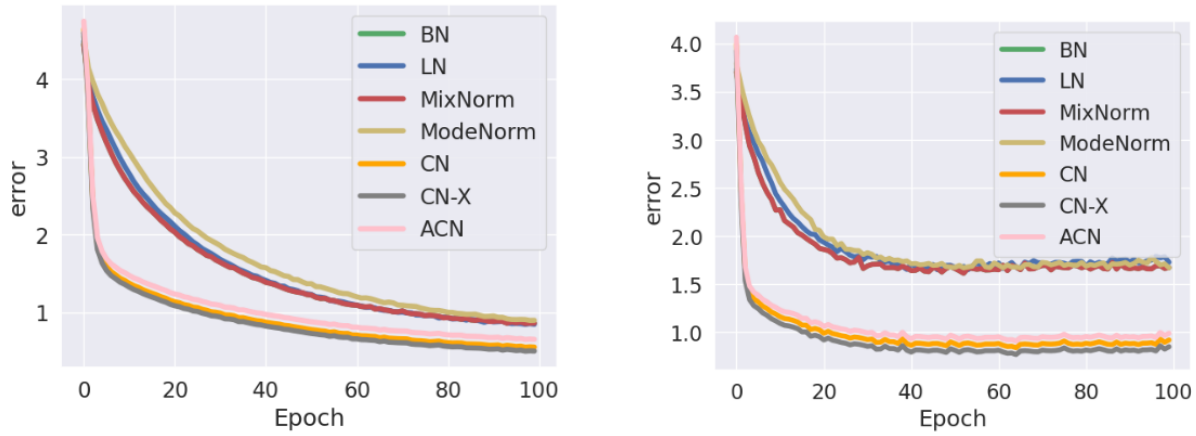
quired when using k-means clusters, as detailed in Tables 3.2 and 3.3. Furthermore, employing $K = 2$ for the Oxford-IIIT Pet dataset and $K = 20$ for CIFAR-100 does not markedly affect ACN performance. This suggests that since contexts are constructed within ACN, merely increasing the number of contexts does not guarantee enhanced model performance.

This experiment highlights the potential advantages of applying CN and CN-X for normalization when we possess a strong understanding of the datasets, allowing us to leverage this knowledge as prior information to construct effective contexts that yield improved performance in both shallow and deep neural networks.

To further evaluate the versatility of CN, CN-X, and ACN, we implement the Vision Transformer (ViT) model [26] and compare its performance against BN, LN, MixNorm, and ModeNorm on the CIFAR-100 dataset. For CN and CN-X, we utilize superclasses as contexts with $K = 20$. In the case of ACN, ModeNorm, and MixNorm, we also set $K = 20$ to ensure a fair comparison across all methods. Table 3.6 demonstrates the versatility of the proposed normalization methods CN, CN-X, and ACN. When applied to the ViT architecture, these methods maintain a performance advantage over BN, LN, MixNorm, and ModeNorm. Similarly to the results obtained with DenseNet, the proposed normalization layers facilitate improved convergence during training and validation,

model	accuracy	precision	recall	f1-score
BN	55.63	8.96	90.09	54.24
LN	54.05	11.82	85.05	53.82
MixNorm	53.2	11.20	87.10	54.23
ModeNorm	54.10	12.12	87.23	54.98
CN	70.76	27.59	98.60	70.70
CN-X	71.28	28.30	98.87	70.98
ACN	60.34	20.21	93.23	60.10

Table 3.6: Performance Rates (%) on the Test Set Using the ViT Architecture with Various Normalization Methods—BN, LN, MixNorm, ModeNorm, CN, CN-X, and ACN—on the CIFAR-100 Dataset, Employing Superclasses as Contexts for CN and CN-X.



(a) Training Error

(b) Validation Error

Figure 3.4: Contrasting Training and Validation Error Curves in CIFAR-100 dataset when using ViT architecture.

as illustrated in Figure 3.4.

In this section, we demonstrate that our proposed normalization methods significantly enhance performance and accelerate convergence in both shallow and deep neural networks. When predefined contexts are not available, we illustrate the feasibility of using k-means clusters as an alternative. Conversely, when contexts are well-defined—such as through superclasses for CN and CN-X—we achieve improved performance. We provide evidence of this through applications with CNN architectures, specifically DenseNet-40 and DenseNet-100, as well as with the Transformer architecture using ViT [26].

To further explore these findings, we propose an additional approach in the following section to effectively construct contexts for CN and CN-X, demonstrating the versatility of these methods and their applicability across various domains.

3.5.2 Domain Adaptation

In this experiment, we introduce an alternative approach to constructing contexts for CN and CN-X in domain adaptation. Domain adaptation [33] is a technique in machine learning, particularly in deep learning, that enables a model trained on data from one domain (source domain) to perform well on data from a different but related domain (target domain). This is useful when labeled data is abundant in the source domain but limited or unavailable in the target domain, which may have different characteristics, like variations in lighting, style, or noise. By aligning feature distributions or representations between domains, domain adaptation allows the model to generalize better across domains, improving performance on tasks where collecting labeled data is challenging.

For CN and CN-X, we will consider two distinct contexts $K = 2$: the source domain and the target domain. Using domains as contexts is motivated by the aim to incorporate domain-specific information into the activation representations. To exemplify this, we employ AdaMatch [99], a domain adaptation algorithm designed to align feature distributions between source and target domains by leveraging labeled source data and a few labeled target samples. AdaMatch uses a dynamically adjusted confidence threshold for pseudo-labeling in the target domain, improving generalization across domains by aligning class distributions while minimizing domain shift. It

combines the tasks of unsupervised domain adaptation (UDA), semi-supervised learning (SSL), and semi-supervised domain adaptation (SSDA). In UDA, we have access to a labeled dataset from the source domain and an unlabeled dataset from the target domain, with the goal of training a model that generalizes effectively to the target data. In this case, we use MNIST as the source dataset and SVHN as the target dataset. These datasets include a range of variations, such as texture, viewpoint, and appearance, and their respective domains, or distributions, are notably distinct.

The baseline model uses BN layers and is trained from scratch using Wide Residual Networks [134]. For comparison, we create additional models by individually replacing the BN layers with LN, MixNorm, ModeNorm, CN, CN-X, and ACN. For MixNorm, ModeNorm, and ACN, we set $K = 2$ to maintain consistency with CN and CN-X. Model training employs the Adam optimizer [60] with a cosine decay schedule, gradually reducing the initial learning rate of 0.03. All models are trained for 100 epochs. The results in Table 3.7 demonstrate that CN, CN-X, and ACN outperform traditional normalization techniques (BN, LN, MixNorm, and ModeNorm) in domain adaptation between MNIST and SVHN. For the MNIST source domain, all methods achieve high performance, with CN-X achieving the best accuracy and F1-score of 99.26%. In contrast, performance differences are more pronounced on the SVHN target domain, where CN-X leads with a significant improvement in accuracy (54.70%), followed closely by CN at 47.63%. These results suggest that CN and CN-X are better suited to handle domain shifts, particularly when there is a substantial difference in data distribution, as seen between MNIST and SVHN. While ACN does not reach the peak accuracy levels of CN-X on SVHN, it still shows a marked improvement over baseline methods like BN and LN, achieving 33.4% accuracy in the target domain. This indicates that ACN contributes to enhanced domain adaptation by capturing some domain-specific features, making it a viable normalization technique for adaptation tasks, though its performance suggests it is less robust to drastic domain shifts compared to CN and CN-X.

These results from CN and CN-X reinforce findings from previous experiments, where contexts are clearly defined. Leveraging well-defined prior knowledge can be highly beneficial, as it allows relevant patterns to be embedded within activation representations. This enhances the overall representation quality and provides normalization benefits that contribute to the stability of the training process. By capturing domain-specific information effectively, CN and CN-X not only improve adaptation to

MNIST (source domain)				
model	accuracy	precision	recall	f1-score
BN	97.36	87.33	79.39	78.09
LN	96.23	88.26	76.20	81.70
MixNorm	98.90	98.45	98.89	98.93
ModeNorm	98.93	98.3	98.36	98.90
CN	99.17	99.17	99.17	99.17
CN-X	99.26	99.20	99.32	99.26
ACN	98.9	98.5	98.90	98.95
SVHN (target domain)				
model	accuracy	precision	recall	f1-score
BN	25.08	31.64	20.46	24.73
LN	24.10	28.67	22.67	23.67
MixNorm	32.14	50.12	37.14	39.26
ModeNorm	32.78	49.87	38.13	40.20
CN	47.63	60.90	47.63	49.50
CN-X	54.70	59.74	54.70	54.55
ACN	33.4	43.83	40.28	42.87

Table 3.7: Test set accuracy (%) of AdaMatch for domain adaptation on MNIST and SVHN datasets using various normalization techniques.

new domains but also support smoother learning by reducing the impact of domain shifts on model performance. This approach highlights the potential of context-driven normalization techniques to boost model robustness in challenging cross-domain tasks, as seen with AdaMatch on the MNIST to SVHN adaptation.

In the next section, we will examine a scenario where the application of ACN is particularly relevant and compare its performance to single-mode normalization (BN) and multi-mode normalization (MixNorm).

3.5.3 Image Generation

Image generation involves creating new, synthetic images by training models to understand and replicate the features and patterns of real images. This process uses a model to learn from a large dataset of images, capturing details like textures, colors, shapes, and spatial relationships. Generated images can range from realistic representations to imaginative interpretations, depending on the training data and model design. An example of method that can generate such images is Generative Adversarial Networks (GANs) [93, 21, 42]. The GAN architecture consists of two neural networks: a generator and a discriminator, which work in tandem through a process called adversarial training. The generator creates synthetic images starting from random noise, while the discriminator evaluates these images, distinguishing between real images (from the training dataset) and those generated by the model. The generator's goal is to create images that can "fool" the discriminator, while the discriminator aims to accurately detect real versus generated images. This adversarial process continues until the generator produces images that are nearly indistinguishable from real ones. GANs have a wide range of applications, including image synthesis, style transfer, super-resolution imaging, and data augmentation. They are also used in fields like healthcare for generating medical images, in entertainment for creating realistic character images, and in autonomous driving for simulating varied road conditions. A common challenge encountered when using GANs is the issue of "mode collapse". This phenomenon occurs when the generator produces only a restricted subset of possible data, leading to a loss of diversity in the generated results. In other words, the generator may focus on producing a specific type of data, neglecting the generation

of other potential variations. This problem can compromise the quality and variety of the generated data, requiring specific techniques and strategies to address and enhance the overall performance of the GAN model. In MixNorm [57], the authors demonstrate that normalizing across multiple modes (mixture components), rather than a single mode as in BN, can help mitigate the issue of "mode collapse". Here, we propose to apply ACN and compare its performance to BN and MixNorm. Notably, CN and ACN are not suited for this scenario, as generated images are produced from random noise vectors, making it difficult to define prior knowledge about vector membership for normalization.

Our baseline model is a Deep Convolutional Generative Adversarial Network (DCGAN) [93], specifically designed for image generation. The generator consists of a linear layer followed by four deconvolutional layers, with the first three layers utilizing Batch Normalization (BN) and a LeakyReLU [83] activation function. The linear layer maps latent space to a higher-dimensional representation, while the deconvolutional layers progressively upsample the input into realistic images. BN stabilizes and accelerates training, and LeakyReLU introduces non-linearity for better learning of complex mappings. We create two additional models by replacing the BN layers with MixNorm and ACN, using $K = 3$ for MixNorm as specified in the paper [57] and matching $K = 3$ for ACN to ensure a fair comparison. All models are trained on CIFAR-100 for 200 epochs using the Adam optimizer [60] with $\alpha = 0.0002$, $\beta_1 = 0$, and $\beta_2 = 0.9$ for both the generator and discriminator. We evaluate GAN quality using the Fréchet Inception Distance (FID) [49], calculated every 10 epochs for efficiency. Figure 3.5 illustrates that the DCGAN incorporating ACN exhibits not only a quicker convergence compared to its batch-normalized (BN) and mixture-normalized (MixNorm) counterparts but also achieves superior (lower) FID scores. Reducing the FID is crucial as it indicates that the generated images are more similar to real images, enhancing the overall quality and diversity of the outputs. A lower FID score suggests that the model is effectively capturing a broader range of features in the training data, which helps mitigate mode collapse—a phenomenon where the generator produces a limited variety of outputs. By improving the distribution of generated images and reducing the gap between real and synthetic data distributions, ACN promotes a more stable training process and encourages the model to explore different modes within the data, leading to richer and more varied image generation. Figure 3.6 showcases examples of images generated by DCGANs utilizing

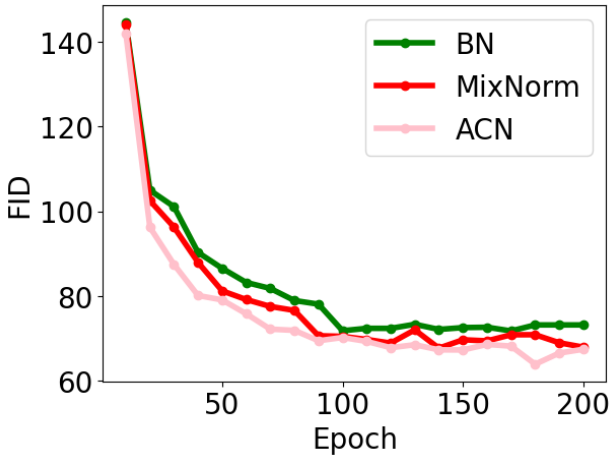


Figure 3.5: ACN integrated as a normalization layer in a DCGAN. Our results show that incorporating ACN into the DCGAN generator leads to improved (lower) Fréchet Inception Distance (FID) scores.

(a) BN (b) MixNorm (c) ACN

Figure 3.6: Examples of generated images at epoch 200 are showcased for BN, MixNorm, and ACN in Figure 3.6a,3.6b, and3.6c, respectively.

BN, MixNorm, and ACN. The results reveal that multi-mode normalization techniques, such as MixNorm and ACN, produce notably clearer object structures in the generated images compared to those using BN. Additionally, both MixNorm and ACN demonstrate greater diversity in their outputs, enhancing the overall richness of the generated content. This improvement in image quality and diversity underscores the effectiveness of these advanced normalization methods, paving the way for more sophisticated and nuanced image generation in future applications.

3.6 Discussion

In this chapter, we proposed three advanced normalization methods: CN, CN-X, and ACN. We demonstrated that single-mode normalization techniques, such as BN and LN, performed less effectively than multi-mode approaches like MixNorm and ModeNorm. Our contributions centered on multi-mode normalization methods that relied on prior knowledge to improve activation normalization during neural network training.

CN and CN-X grouped data into predefined structures, called contexts, before training. CN used these contexts within each mini-batch to estimate and apply normalization parameters specific to each context, while CN-X defined these parameters as trainable weights that updated dynamically through backpropagation. We outlined multiple methods for constructing contexts, including k-means clustering, superclass assignments, and domain-based contexts in domain adaptation tasks. When context construction was less straightforward, ACN provided flexibility by allowing the number of contexts to be set as a hyperparameter.

In tasks spanning classification, domain adaptation, and image generation, our proposed methods consistently delivered superior performance compared to traditional normalization techniques. CN and CN-X exhibited higher robustness than ACN when contexts were well-defined, emphasizing the effectiveness of prior knowledge in enhancing neural network representation, accelerating convergence, and improving performance.

To advance this approach, we further explored how structured prior knowledge in multimodal representations (Part II) reduced parameter tuning costs and minimized the reliance on large labeled datasets, achieving competitive performance with fewer resources.

Conclusion

In this part, the focus is on the importance of activation normalization in deep neural networks (DNNs) and the proposed advancements to address training challenges. Single-mode methods like Batch Normalization (BN) have been successful in mitigating issues like internal covariate shift but struggle with small batch sizes or non-uniform data distributions. To address these limitations, multi-mode approaches like MixNorm and ModeNorm have been developed, though they are often computationally expensive and require complex algorithms.

The proposed multi-mode normalization methods—Context Normalization (CN), Extended Context Normalization (CN-X), and Adaptive Context Normalization (ACN)—leverage prior knowledge to improve training convergence and performance. These methods are designed to handle more complex data distributions and accelerate the training process. CN and CN-X group data into predefined contexts, applying specific normalization parameters to each context within a mini-batch. CN-X further enhances CN by making these context parameters trainable through backpropagation, thus providing additional flexibility. ACN goes a step further by allowing the dynamic adjustment of the number of contexts, making it highly adaptable to different training scenarios.

Part II

Cross-Modal Alignment Learning (CM-AL) for Multimodal Data Representation

This part addresses the challenges associated with **high costs**, **data limitations**, and **scalability** in training multimodal encoders, particularly when integrating data from different modalities such as text, image, audio, and video. To overcome these challenges, we introduce OneEncoder, a novel approach for cross-modal alignment learning that leverages prior knowledge to enhance the encoder representations, reducing the dependency on large-scale paired datasets and making the training process more efficient.

The part is structured as follows: Chapter 4 provides a review of existing methods for cross-modal alignment learning, examining current approaches and their limitations. Chapter 5 details the OneEncoder framework, explaining its design, the integration of prior knowledge, and its ability to efficiently handle various modalities. The proposed framework is validated on three tasks: **zero-shot classification**, **querying**, and **visual question answering**. We compare OneEncoder with state-of-the-art methods, showcasing its improvements in efficiency and scalability.

Chapter 4

State of the Art in Cross-Modal Alignment Learning Techniques

4.1 Introduction

The advancement of large language models (LLMs) [111, 89, 10, 3, 28] has significantly broadened their application across various domains beyond natural language processing, including vision, audio, and even multimodal tasks. These models leverage vast amounts of data and sophisticated architectures, allowing them to capture intricate patterns and relationships within and between modalities. As LLMs have grown in capability, they have become increasingly integral to cross-modal learning tasks, where the goal is to align disparate modalities—such as text, images, and audio—within a shared semantic space. This alignment is crucial as it facilitates improved representation learning, enabling models to leverage contextual information from one modality to enhance the learning process in another. For instance, in applications like visual question answering, models can combine visual data with textual queries, leading to more accurate and contextually aware predictions. Consequently, the integration of LLMs into cross-modal frameworks not only boosts performance but also opens new avenues for research in areas such as automated content generation, multimodal sentiment analysis, and improved user interaction systems.

4.2 Dual Modality Alignment (DMA)

DMA techniques focus on integrating representations from pairs of distinct modalities, such as image-text, text-audio, and image-audio, to create a unified semantic space. This integration is crucial for enabling models to understand and relate information across different types of content, allowing for more complex and contextually rich cross-modal tasks.

Early breakthroughs in DMA include models like Flamingo [1], which introduces cross-attention [116] mechanisms to align visual and textual features directly within LLMs. This architecture allows for intricate interactions between images and text, enhancing the model's performance in vision-language tasks such as visual question answering by enabling each modality to directly inform the other through a shared attention mechanism.

ConVIRT [139] pioneered contrastive learning in the medical domain, aligning medical images and textual descriptions to facilitate cross-modal retrieval in data-limited settings. Building on this approach, CLIP [94] extended contrastive learning to a large scale with extensive paired image-text datasets, creating a shared representation space for open-vocabulary recognition and zero-shot learning across general domains. CLIP's generalization has broadened its applicability, supporting diverse tasks that require robust understanding of visual and textual inputs without additional fine-tuning.

ALIGN [20] builds on these ideas with a focus on robustness to noisy data, which is essential for real-world applicability. ALIGN optimizes contrastive learning techniques to handle large and potentially noisy image-text datasets effectively, ensuring consistent performance in more variable environments where data quality may be less controlled.

Research in DMA has evolved from traditional image-text models to encompass various modality pairings tailored for specific applications. Text-audio alignment [98, 120] utilizes self-supervised and contrastive learning to connect audio signals with transcriptions, enhancing speech recognition and audio retrieval. Similarly, image-audio alignment [19] combines visual and auditory data, improving multimedia applications like audiovisual content analysis, where synchronized data offers deeper insights. Text-video models [56] correlate descriptive text with video sequences, facilitating tasks such as video summarization and action recognition by capturing temporal and semantic relationships.

Emerging alignments are broadening the scope of cross-modal learning. For example, text-3D models [87, 48] link textual descriptions to 3D shapes, which is valuable in virtual reality and robotics for generating accurate renderings. In neurocomputational fields, text-EEG alignment [29, 30] connects language with brain activity data, supporting brain-computer interface research and assistive technologies. Additionally, image-depth alignment [62] is vital for autonomous driving and AR/VR, pairing visual data with depth information for safer, more accurate interpretations. Finally, text-sensor alignment [127] integrates language with diverse sensor data, enhancing health monitoring and smart home applications by enabling more intuitive human-computer interactions. Collectively, these dual-modality pairings are significantly advancing cross-modal learning across various industries.

Despite these advancements, dual-modality alignment approaches face notable challenges. A significant limitation is their reliance on extensive aligned datasets, which are expensive to curate and may not be available for all modality pairs, particularly in niche or specialized fields. Moreover, these approaches are typically designed to handle only two modalities at a time, which constrains their ability to generalize to or incorporate additional modalities. This modality restriction limits the broader applicability of DMA models, especially in scenarios where integrating information from multiple modalities simultaneously is beneficial or required. Consequently, while dual-modality alignment has paved the way for cross-modal alignment learning, there is a clear need for further research to address these resource dependencies and extend current models' capacity to work across multiple modalities in a unified framework.

4.3 Multiple Modalities Alignment (MMA)

MMA advances the concept of dual-modality alignment by synchronizing representations from three or more distinct modalities, creating a shared semantic space that enables more comprehensive multimodal understanding. For example, AudioCLIP [44] extends CLIP's capabilities to incorporate audio alongside text and image data, allowing it to perform tasks that require understanding across

audio, visual, and textual elements. This model enriches applications such as video retrieval and audiovisual content analysis, where all three modalities provide unique yet complementary information.

Similarly, ImageBind [39] takes multimodal alignment even further by synchronizing six modalities—text, images, audio, depth, thermal, and IMU (inertial measurement unit) data. By leveraging the zero-shot capabilities of vision-language models, ImageBind can link diverse sensory data into a unified space without requiring aligned training data for every combination, enabling cross-modal retrieval and understanding tasks across a broader spectrum of sensory inputs. This alignment of heterogeneous modalities is particularly beneficial for applications like robotics and virtual/augmented reality, where multi-sensory input aids in creating a rich, context-aware environment.

Another recent model, NExT-GPT [123], builds on multimodal understanding by enabling any-to-any modality transformations. It allows for flexible input-output combinations across modalities, which is essential for scenarios demanding complex data interaction, such as assistive technology and interactive AI. However, NExT-GPT still depends on large aligned datasets for training, limiting its accessibility and scalability. These high demands on resources underscore the challenges of current MMA models, which rely on computationally heavy architectures and extensive aligned data.

As the field progresses, moving from resource-intensive architectures toward more lightweight models is essential. Lightweight MMA models aim to lower dependency on extensive training datasets and reduce computation costs while still maintaining alignment across multiple modalities. This shift supports practical deployment in diverse applications, such as mobile devices or edge computing, where multimodal understanding is needed but resources are constrained. Developing such efficient MMA models will be pivotal for expanding multimodal AI into everyday applications, allowing high-performance interaction across multiple modalities even in resource-limited settings.

4.4 Transitioning to Lightweight Models for Modalities Alignment

In recent advancements, researchers have optimized multimodal learning by employing frozen pre-trained models and modality-specific tokens to align multiple data types using a single encoder. This technique drastically reduces the need for large aligned datasets and minimizes the parameters that need training, effectively lowering the computational demands of multimodal models [46, 138]. One notable example is Meta-Transformer [138], which leverages a frozen visual encoder alongside modality tokens, achieving strong performance across 12 distinct data modalities without requiring individual encoders for each type. By keeping the core encoder fixed, Meta-Transformer aligns diverse data types through minimal modifications, facilitating efficient processing across modalities like images, text, and audio.

Building on this idea, Han et al. [46] introduced a unified framework using a frozen CLIP model and a Universal Projection (UP) module that dynamically switches between modalities via modality tokens. This approach aligns eight modalities within a single model architecture, using the modality tokens to activate relevant components of the frozen encoder based on input type. These methods represent a significant shift towards modular, parameter-efficient architectures in multimodal AI, sidestepping the need for separate encoders for each modality. However, a current limitation lies in integrating entirely new modalities; adding a new data type to these models often requires extensive adjustments or even retraining to ensure cohesive alignment with existing modalities.

4.5 Discussion

This chapter explored advancements in cross-modal alignment learning (CM-AL), focusing on dual-modality alignment (DMA) and multiple-modalities alignment (MMA). While DMA techniques, such as CLIP and ALIGN, have shown strong performance, they are limited by their reliance on large aligned datasets and the need to pair only two modalities. MMA approaches, like AudioCLIP and ImageBind, extend this by integrating multiple modalities, but they too face challenges related to data requirements and computational costs.

Recent efforts to create lightweight models, such as Meta-Transformer, reduce these issues by leveraging frozen pretrained models. However, integrating new modalities remains difficult. Our proposed approach offers a solution by introducing an open, progressive alignment framework, allowing seamless integration of new modalities without retraining. This improves scalability and adaptability while reducing computational overhead, making our framework suitable for real-world applications with limited resources.

Chapter 5

OneEncoder

5.1 Introduction

To develop a lightweight approach for training multimodal systems, we leverage *prior knowledge* to significantly reduce the number of tunable parameters, thereby minimizing the need for large datasets. This strategy supports the creation of an open, flexible system that can incorporate additional modalities in the future at a low cost.

To achieve it, we introduce OneEncoder, which progressively aligns four modalities (image, text, audio, and video) within a single unified framework.

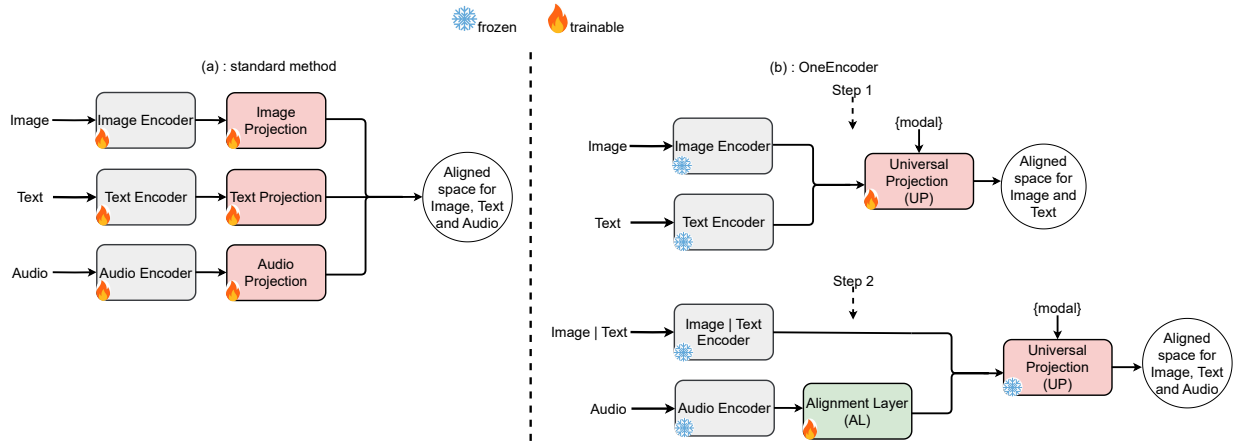


Figure 5.1: Comparison of alignment methods: Standard approaches train large, modality-specific encoders, requiring extensive data and compute. OneEncoder uses frozen encoders, a lightweight Universal Projection (UP) module, and trains a small Alignment Layer (AL) module for new modalities, enabling efficient, flexible alignment.

As shown in Figure 5.1, OneEncoder incorporates frozen, pretrained modality-specific en-

coders alongside a lightweight Universal Projection(UP) module, a compact Alignment Layer (AL) module, and modality tokens (referred to as "modal") to enable seamless switching between modalities with shared parameters. Here, the modality token encodes prior knowledge by embedding the modality type directly into the UP representation, allowing for a unified parameter set across diverse modalities. Unlike conventional methods that require tuning separate encoders for each modality, OneEncoder achieves efficiency by freezing the modality-specific encoders purely for feature extraction, thereby using a single encoder across modalities.

We propose a two-step approach for progressively training our framework across multiple modalities, emphasizing prior knowledge to streamline parameter tuning and reduce reliance on large aligned datasets. Specifically, we introduce a modality token as an element of prior knowledge that embeds modality information directly into the representation. This approach enables us to use the same parameters across modalities and achieve effective alignment with minimal tuning.

Step 1 involves pretraining the UP using image-text data, which is more widely available than other modality data. **Step 2** is consistent for all new modalities: we freeze the pretrained UP and train only the lightweight AL to map new modalities into the shared space established by the UP. For instance, we first align audio with image and text, then align video with image, text, and audio. The purpose of the AL is solely to project new data into the shared space without altering the underlying representation. By focusing on this modular design with a compact UP and AL, OneEncoder achieves a balance between alignment effectiveness and reduced complexity, allowing for scalable integration of new modalities at a low cost. This method ensures robust performance even without extensive aligned datasets.

Our contributions are summarized as follows:

- **Lightweight and efficient architecture:** We propose OneEncoder, which reduces computational costs by using frozen pretrained encoders for feature extraction. Only a small Universal Projection (UP) module is trained, significantly lowering training time and resource requirements compared to methods that train large, modality-specific encoders.
- **Unified representation space with modality tokens:** We introduce modality tokens to guide the UP, enabling a single set of parameters to align features from different encoders

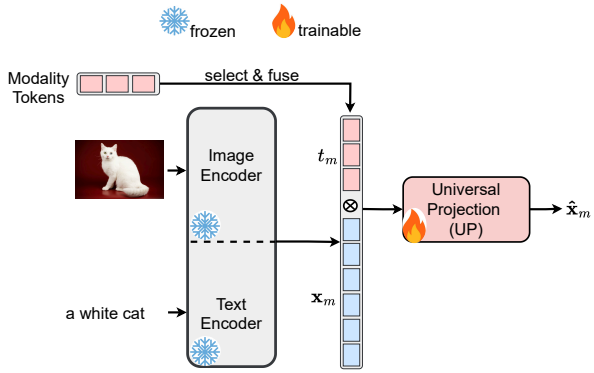
in the same space. This removes the need for multiple modality-specific alignment heads, simplifying the architecture while preserving strong performance.

- **Progressive and flexible modality integration:** Unlike existing closed frameworks, OneEncoder supports progressive expansion. Using a two-step training process, new modalities are integrated via a lightweight Alignment Layer (AL) module, without retraining the UP or existing encoders. This makes the framework adaptable to evolving multimodal needs.
- **Reduced reliance on large paired datasets:** OneEncoder achieves competitive results even with smaller paired datasets, thanks to the efficiency of the UP and the rich features extracted from frozen encoders. This addresses a major limitation of state-of-the-art methods, which often require vast, hard-to-collect paired data.
- **Parameter efficiency and scalability:** By freezing large pretrained encoders and limiting training to the compact UP and AL modules, OneEncoder drastically reduces the number of trainable parameters. This makes the framework more scalable and practical for real-world scenarios with limited computational resources.
- **Broad modality compatibility:** The framework naturally handles diverse modalities (e.g., image, text, audio, video) and facilitates seamless alignment between them, without the architectural complexity seen in many current multimodal systems.

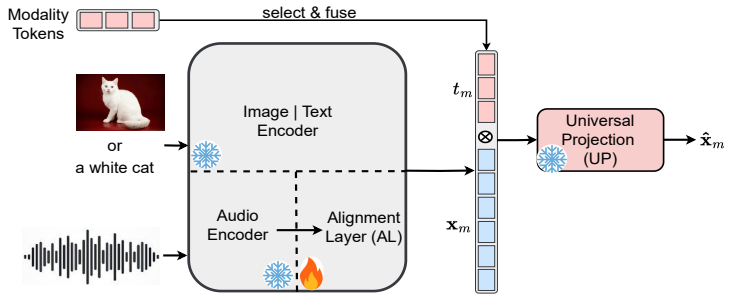
5.2 Model Architecture: OneEncoder

Drawing from research by [138, 46], we capitalize on the robust modality transfer capabilities of pretrained encoders. This approach allows to leverage pretrained modality-specific encoders, who are trained on large modality-specific datasets, which are more readily available than large aligned datasets. Within OneEncoder, we employ ViT [26] for image encoding, BERT [25] for text encoding, Wav2Vec2 [5] for audio encoding and VideoMAE [110] for video encoding. Each model produces an input token $\mathbf{x} \in \mathbb{R}^{L \times D}$ as its output, where L represents the sequence length and D denotes the token dimension. Consistent with previous research [138, 46], we also maintain the parameters of these models frozen during training. Figure 5.2 illustrates the three primary elements comprising

OneEncoder: modality-specific encoders, a Universal Projection (UP) module, and an Alignment Layer (AL) module.



(a) **Step 1:** Training the Lightweight UP and Aligning Image-Text Modalities



(b) **Step 2:** Freeze the Pretrained UP, Train the Compact AL, and progressively Align Audio with the Image-Text Modalities from Step 1. This process can be extended to align additional future modalities, such as video.

Figure 5.2: **OneEncoder architecture:** OneEncoder uses frozen pretrained encoders, a Universal Projection (UP) module, and an Alignment Layer (AL) module. In step 1, the UP (a Transformer encoder) aligns text and image modalities. In step 2, the frozen UP aligns audio through the AL (a small MLP) by pairing audio with either image or text. The UP fuses input features ($\mathbf{x}_\mathrm{m}$) and modality tokens ($\mathbf{t}_\mathrm{m}$) to switch between modalities.

Universal Projection (UP) module. Unlike existing methods that train separate modality-specific encoders, we introduce a single encoder, UP, to align all modalities in a shared space (ref. Figure 5.2).The UP is designed as a lightweight module with four Transformer encoder blocks [116], and each block is composed of the following components:

1. **Multi-head self-attention [116]:** A multi-head attention mechanism with four heads to model cross-token dependencies.
2. **Layer normalization [4]:** Applied after the attention mechanism to stabilize training and accelerate convergence.
3. **Feedforward layers:** Fully connected feedforward layers to refine representations and enhance expressiveness.

The UP is designed to project different modalities into a shared representation space using the same set of parameters. To make this possible, we introduce modality tokens, inspired by Han et al. [46]. These tokens act as learnable parameters that help the UP distinguish and adapt to each modality's characteristics. During training, modality tokens are updated via backpropagation to optimize cross-modal alignment. For a given modality $m \in \mathcal{M}$ (e.g., $\{\text{image}, \text{text}\}$ in step 1), the modality features $\mathbf{x}_{\text{m}} \in \mathbb{R}^{L \times D}$, extracted from the frozen encoder, are fused with the corresponding tokens $\mathbf{t}_{\text{m}} \in \mathbb{R}^{1 \times D}$ before being passed through the UP:

$$\hat{\mathbf{x}}_{\text{m}} = \text{UP}(\mathbf{t}_{\text{m}} \otimes \mathbf{x}_{\text{m}}), \tag{5.1}$$

In Equation 5.1, the fusion operation $\otimes$ can be performed through either element-wise addition, as described in [119], or cross-attention, as in [121], where the modality tokens $\mathbf{t}_{\text{m}}$ act as the query, and the modality features $\mathbf{x}_{\text{m}}$ serve as both the key and value. In the addition operation, modality tokens $\mathbf{t}_{\text{m}}$ are added element-wise to the input tokens $\mathbf{x}_{\text{m}}$, directly injecting modality-specific information into the input representation. This simple mechanism enhances features with minimal computational overhead. In contrast, cross-attention uses $\mathbf{t}_{\text{m}}$ as the query and $\mathbf{x}_{\text{m}}$ as key and value, enabling the model to focus on the most relevant input features for each modality. This allows for more fine-grained interactions, adapting representations to the unique structure of each modality.

Alignment Layer (AL) module. In OneEncoder, the AL makes it easy to integrate new modalities without retraining the entire framework. After training the UP in step 1 (Figure 5.2a), the UP is frozen, and in step 2 (Figure 5.2b), only the AL is trained. The AL's purpose is not to improve the representation but to project the pretrained encoder features into the input space of

the UP. It is a lightweight two-layer MLP, making it much smaller and faster to train than the UP. During the forward pass for a new modality m, the AL transforms the input features $\mathbf{x}_\text{m} \in \mathbb{R}^{L \times D}$, which are then fused with the modality tokens $\mathbf{t}_\text{m}$ and fed into the frozen UP for alignment:

$$\mathbf{x}_\text{m} = \text{AL}(\mathbf{x}_\text{m}) \tag{5.2}$$

$$\hat{\mathbf{x}}_\text{m} = \text{UP}(\mathbf{t}_\text{m} \otimes \mathbf{x}_\text{m}) \tag{5.3}$$

Step 2 can be repeated for each new modality, allowing the framework to expand progressively.

5.3 Training Procedure

The OneEncoder alignment process follows a progressive two-step approach, as shown in Figure 5.2. In step 1 (Figure 5.2a), the UP is trained to initialize the alignment for the initial set of modalities. In step 2 (Figure 5.2b), new modalities can be added by training only the AL, while keeping the UP frozen. This second step can be repeated as needed, allowing the framework to grow and support additional modalities over time.

- **Step 1: Image-Text Alignment.** Using available aligned image-text datasets and advancements in the field [94, 55], we train the UP to align image and text modalities in a shared latent space. The UP's parameters are updated using the adapted InfoNCE loss [88] for contrastive (image, text) representation learning by Zhang et al. [139].

  During training, we sample a minibatch of K input pairs $(\hat{\mathbf{x}}^i_\text{image}, \hat{\mathbf{x}}^i_\text{text})$ from the dataset. The contrastive loss between image and text for each paris $(\hat{\mathbf{x}}^i_\text{image}, \hat{\mathbf{x}}^j_\text{text})$ in the minibatch can be formulated as follow:

  $$\ell_{ij} = -\log\left(\frac{\exp(\langle \hat{\mathbf{x}}^i_\text{image}, \hat{\mathbf{x}}^j_\text{text} \rangle / \tau)}{\sum_{k=1}^{K} \exp(\langle \hat{\mathbf{x}}^i_\text{image}, \hat{\mathbf{x}}^k_\text{text} \rangle / \tau)}\right) \tag{5.4}$$

  The term $\langle \hat{\mathbf{x}}^i_\text{image}, \hat{\mathbf{x}}^j_\text{text} \rangle$ represents cosine similarity, with $\tau \in \mathbb{R}^+$ as a temperature parameter. This loss function preserves mutual information between true pairs through representation functions. To ensure symmetry, we introduce a similar contrastive loss from text to image:

  $$\ell_{ji} = -\log\left(\frac{\exp(\langle \hat{\mathbf{x}}^i_\text{image}, \hat{\mathbf{x}}^j_\text{text} \rangle / \tau)}{\sum_{k=1}^{K} \exp(\langle \hat{\mathbf{x}}^k_\text{image}, \hat{\mathbf{x}}^j_\text{text} \rangle / \tau)}\right) \tag{5.5}$$

The matching pairs are situated along the diagonal of the similarity matrix $(\hat{\mathbf{x}}^i_{\text{image}}, \hat{\mathbf{x}}^i_{\text{text}})$, which serves as the target for the loss function:

$$t_{ij} = \frac{\exp((\langle \hat{\mathbf{x}}^i_{\text{image}}, \hat{\mathbf{x}}^j_{\text{image}} \rangle + \langle \hat{\mathbf{x}}^i_{\text{text}}, \hat{\mathbf{x}}^j_{\text{text}} \rangle)/2 \cdot \tau)}{\sum_{k=1}^{K} \exp((\langle \hat{\mathbf{x}}^i_{\text{image}}, \hat{\mathbf{x}}^k_{\text{image}} \rangle + \langle \hat{\mathbf{x}}^i_{\text{text}}, \hat{\mathbf{x}}^k_{\text{text}} \rangle)/2 \cdot \tau)} \tag{5.6}$$

$$t_{ji} = \frac{\exp((\langle \hat{\mathbf{x}}^i_{\text{image}}, \hat{\mathbf{x}}^j_{\text{image}} \rangle + \langle \hat{\mathbf{x}}^i_{\text{text}}, \hat{\mathbf{x}}^j_{\text{text}} \rangle)/2 \cdot \tau)}{\sum_{k=1}^{K} \exp((\langle \hat{\mathbf{x}}^j_{\text{image}}, \hat{\mathbf{x}}^k_{\text{image}} \rangle + \langle \hat{\mathbf{x}}^j_{\text{text}}, \hat{\mathbf{x}}^k_{\text{text}} \rangle)/2 \cdot \tau)} \tag{5.7}$$

The ultimate training loss $\mathcal{L}$ (5.8) is computed by combining the two losses ℓ_{ij} and ℓ_{ji} and averaging them over all pairs within each minibatch.

$$\mathcal{L} = \frac{1}{2 \cdot K} \sum_{i=1}^{K} \sum_{j=1}^{K} t_{ij} \cdot \ell_{ij} + t_{ji} \cdot \ell_{ji} \tag{5.8}$$

- **Step 2: Alignment of Future Modalities.** Once the UP is trained in Step 1, it is frozen for Step 2. In this step, a new modality m_i is aligned with the already aligned image and text modalities by selecting one (either image or text) for alignment, as illustrated in Figure 5.2b using the audio modality. The alignment of the selected modality ensures transitive alignment across all three modalities (image, text, and m_i). During this step, only the AL is trained, using the same loss function as in Step 1 (Equation 5.8) to update its parameters for consistent input to the UP. This process is repeated whenever a new modality m_j is introduced (e.g., video).

Algorithm 6 provides a detailed procedure for training the UP on text-image modalities. Once trained, the UP is utilized in Algorithm 7 to align a new modality, denoted as m_2, with the set of already aligned modalities, $\mathcal{M}$, using an intermediary modality m_1, where m_1 must be part of $\mathcal{M}$. This alignment process is achieved by training the AL to project the new modality, m_2, into a coherent space compatible with the UP representation. After this process, the expanded set of aligned modalities becomes $\mathcal{M} \cup \{\mathrm{m}_2\}$. This alignment can be repeated indefinitely, allowing additional modalities to be aligned with those already in $\mathcal{M}$.

In Algorithm 8, the OneEncoder framework is used to represent any modality in $\mathcal{M}$. For text and image modalities, only the UP is required, while for other modalities, both the UP and AL are necessary.

Algorithm 6: Step 1: Training the Universal Projection (UP) module on the image-text modality

Input : image_encoder; text_encoder; $\mathbf{I}$: minibatch of aligned images; $\mathbf{T}$: minibatch of aligned texts; UP: transformer; $\mathcal{M} = \{\text{image, text}\}$; $\{\mathbf{t}_{\text{m}}\}_{\text{m}\in\mathcal{M}} \in \mathbb{R}^{N\times D}$; τ: learned temperature parameter; $\otimes$: fusion operator

Output: Trained UP; List of aligned modalities; modality tokens

// *Freeze the pretrained encoders*

- Freeze(image_encoder)
- Freeze(text_encoder)

// *Extract feature representations of each modality*

- $\mathbf{X}_{\text{image}}$ = image_encoder($\mathbf{I}$)
- $\mathbf{X}_{\text{text}}$ = text_encoder($\mathbf{T}$)

// *Encode each modality after selection and fusion*

- $\hat{\mathbf{X}}_{\text{image}} = \text{UP}(\mathbf{t}_{\text{image}} \otimes \mathbf{X}_{\text{image}})$
- $\hat{\mathbf{X}}_{\text{text}} = \text{UP}(\mathbf{t}_{\text{text}} \otimes \mathbf{X}_{\text{text}})$

// *Compute Loss and Update UP Parameters,* $\mathbf{t}_{image}$*, and* $\mathbf{t}_{text}$

- Compute Loss using Equation 5.8: $\mathcal{L}(\hat{\mathbf{X}}_{\text{image}}, \hat{\mathbf{X}}_{\text{text}}, \tau)$
- Update the UP parameters, $\mathbf{t}_{\text{image}}$, and $\mathbf{t}_{\text{text}}$ using an optimizer algorithm based on the computed loss.

// *Return the trained UP, list of aligned modalities* $\mathcal{M}$*, modality tokens*

- return UP, $\mathcal{M}$, $\{\mathbf{t}_{\text{m}}\}_{\text{m}\in\{\text{image,text}\}}$

Algorithm 7: Step 2: Align a new modality with the previously aligned modalities

Input : m_1_encoder; m_2_encoder; $\mathbf{M}_1$: minibatch of aligned m_1 modality; $\mathbf{M}_2$: minibatch of aligned m_2 modality; UP: pretrained transformer in algorithm 6; $\mathcal{M}$: aligned modalities; $\{\mathbf{t}_m\}_{m\in\{m_1,m_2\}} \in \mathbb{R}^{N\times D}$: modality tokens ; τ: learned temperature parameter; $\otimes$: fusion operator; AL: Multi-layer Perceptron

Output: Trained AL; List of aligned modalities; $\mathbf{t}_{m_2}$

// *Freeze the pretrained encoders, UP and m_1 modality token*

- Freeze(m_1_encoder)
- Freeze(m_2_encoder)
- Freeze(UP)
- Freeze($\mathbf{t}_{m_1}$)

// *Extract feature representations of each modality*

- $\mathbf{X}_{m_1} = m_1$_encoder($\mathbf{M}_1$)
- $\mathbf{X}_{m_2} = m_2$_encoder($\mathbf{M}_2$)

Project feature representations with the AL

- $\mathbf{X}_{m_1} = \text{AL}(\mathbf{X}_{m_1})$
- $\mathbf{X}_{m_2} = \text{AL}(\mathbf{X}_{m_2})$

// *Encode each modality after selection and fusion*

- $\hat{\mathbf{X}}_{m_1} = \text{UP}(\mathbf{t}_{m_1} \otimes \mathbf{X}_{m_1})$
- $\hat{\mathbf{X}}_{m_2} = \text{UP}(\mathbf{t}_{m_2} \otimes \mathbf{X}_{m_2})$

// *Compute Loss and Update AL Parameters and $\mathbf{t}_{m_2}$*

- Compute Loss using Equation 5.8: $\mathcal{L}(\hat{\mathbf{X}}_{m_1}, \hat{\mathbf{X}}_{m_2}, \tau)$
- Update the AL parameters and $\mathbf{t}_{m_2}$ using an optimizer algorithm based on the computed loss.

// *Return the trained AL, list of aligned modalities and m_2* modality token

- Update list of aligned modalities: $\mathcal{M} = \mathcal{M} \cup \{m_2\}$
- return AL, $\mathcal{M}$, $\mathbf{t}_{m_2}$

Algorithm 8: Inference: Encoding a Given Modality Using Pretrained UP and AL

Input : m: modality of the data to be encoded; $\mathbf{M}$: minibatch of data from modality m; UP: Universal Projection module; AL_m: Alignment Layer module for modality m; $\mathbf{t}_\text{m}$: token representing modality m; m_encoder: encoder for modality m

Output: Encoded representation data

// *Extract feature representations*

$\mathbf{X} = \text{m\_encoder}(\mathbf{M})$

if $m \notin \{image, text\}$ **then**

  // *Use AL for feature projection*

  $\mathbf{X} = \text{AL}_\text{m}(\mathbf{X})$

end

// *Encode with the Universal Projection*

$\hat{\mathbf{X}} = \text{UP}(\mathbf{t}_\text{m} \otimes \mathbf{X})$

// *Return encoded representation of input data*

return $\hat{\mathbf{X}}$

5.4 Results

In this section, we aim to use OneEncoder to align four different modalities: image, text, audio, and video. Given the greater availability of datasets paired with text, we propose leveraging text as the central modality for transitive alignment. The alignment process can be summarized as follows:

1. **Align Image with Text:** Train the UP using Algorithm 6 on the image-text modality pair.
2. **Align Audio with Image and Text:** Train AL_audio using Algorithm 7 on the text-audio modality pairs.
3. **Align Video with Image, Text, and Audio:** Train AL_video using Algorithm 7 on the

text-video modality pairs.

The order of alignment steps can be adjusted based on the availability of aligned data and the specific modalities to be aligned.

5.4.1 Datasets

Training Datasets. Our goal is to achieve robust performance on downstream tasks using a lightweight framework trained on a modest dataset. Following the approach of virTex [24], we train the UP on a combined dataset, which includes COCO Captions [17], Flickr30K [129], and TextCaps [106].

To train the AL_{audio}, we utilize the LibriSpeech Speech Recognition Alignment (SRA) [90] Dataset, a corpus containing approximately 1,000 hours of 16kHz recorded English speech.

For the AL_{video}, we employ the Microsoft Research Video to Text (MSR-VTT) [126] dataset, a large-scale resource designed for open-domain video captioning.

A detailed description of all datasets used in training the OneEncoder framework is provided in Table 5.1.

Validation Datasets. For validating OneEncoder, we use various datasets, tailored either for

Dataset	Type	Training Size	Validation Size
COCO Captions [17]	text-image pairs	413,915	202,520
Flickr30K [129]	text-image pairs	158,915	_
TextCaps [106]	text-image pairs	109,765	15,830
SRA [90]	text-audio pairs	281,241	5,559
MSR-VTT [126]	text-video pairs	6,513	497
DAQUAR [84]	text-image pairs	6,794	5,673

Table 5.1: Training datasets

specific modality-based validation (e.g., classification tasks) or cross-modal validation (e.g., zero-shot tasks). A comprehensive description of the datasets used for validation is provided in Table 5.2.

Dataset	Dataset Type	Training Size	Validation Size
CIFAR-10 [65]	image	50,000	10,000
Oxford-IIIT Pets [91]	image	3,680	3,669
CIFAR-100 [64]	image	50,000	10,000
Caltech 101 [35]	image	7,659	3,060
Tiny ImageNet [67]	image	100,000	10,000
SST-2 [107]	text	67,349	872
TREC [118]	text	5,452	500
Emotion [103]	text	16,000	2,000
GTZAN [113]	audio	1,000	_
UrbanSound8K [101]	audio	7,980	1,022
ESC-50 [92]	audio	1,600	400
MSVD [12]	text-video	48,779	4,291
LSMDC [74]	text-video	118,081	_

Table 5.2: Validation datasets

5.4.2 Implementation Details

Architecture. The pretrained encoders for each modality are as follows: ViT-base [26] with 86M parameters for images, BERT-base [25] with 110M parameters for text, Wav2Vec [5] with 317M parameters for audio and VideoMAE-base [110] with 94.2M for video. Additionally, the UP consists of four Transformer encoder blocks with 4M parameters, while the AL comprises a multi-layer perceptron with 65,792 parameters. The size of modality tokens for each modality is $\mathbb{R}^{1\times768}$.

Training Details. We use the AdamW optimizer [77] with a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.95$, and a weight decay of 0.001. For step 1, we train to align image-text pairs, updating only the UP parameters, on a single A100 GPU for 500 epochs with a batch size of 512. For step

2, to align other modalities (audio and video) , we freeze the pretrained UP from step 1, and train only the $\text{AL}_\text{m}, \text{m} \in \{\text{audio, video}\}$ for 100 epochs, using the same parameters as in step 1 with a batch size of 64.

We trained two OneEncoder variants, each utilizing a different fusion operation: addition and scaled dot product attention [116]. For simplicity, we refer to the model using addition as **OneEncoder-$\oplus$**, and the model using scaled dot product attention as **OneEncoder-$\odot$**.

Our objective is not to achieve state-of-the-art results, which typically demand resource-intensive architectures and extensive hyperparameter tuning. Instead, we aim to explore the behavior of frozen versus non-frozen modality-specific encoders. Specifically, we seek to demonstrate that using frozen encoders within our OneEncoder framework can notably enhance performance and, in many cases, yield better representations for downstream tasks. For a fair comparison, we refer to the baseline approach, which involves training modality-specific encoders, as the **Base** framework.

5.4.3 Quantitative Evaluation

UP Validation Following Image-Text Modalities Training

After training the UP on a combined dataset of COCO Captions, Flickr30K, and TextCaps, we validate the OneEncoder framework by benchmarking it against the baseline CLIP [94]. In our method, the pretrained ViT and BERT models remain frozen during training, with only the UP's 4M parameters being updated. In contrast, the baseline requires training all 196M parameters of the ViT and BERT models. For specific tasks, we employ pretrained models: ResNet-18 [47], EfficientNet-B0 [109], and Swin Transformer [76] for image processing, and RoBERTa [75], DistilBERT [102], and XLNet [128] for text processing.

We encode each modality using Algorithm 8 within the OneEncoder framework and evaluate the performance on various classification tasks.

Zero-shot Classification is a task where a model, trained on labeled images, can classify new images from previously unseen classes. It validates the model's generalization capability and assesses semantic understanding and transfer learning. Using the CLIP approach, we transform labels into text descriptions (*"A photo of a {label}."*), encode them with a pretrained model, compute cosine similarity with image embeddings, and use softmax to determine class probabilities.

Zero-shot image classification obviates the need for retraining pretrained models on target datasets,

model	**CIFAR-10**	**Oxford-IIIT Pets**	**CIFAR-100**	**Caltech-101**	**Tiny ImageNet**
CLIP [94]	62.12	58.27	53.06	52.17	47.15
OneEncoder-⊕	**78.15**	**69.23**	**58.18**	**56.20**	**52.27**
OneEncoder-⊙	74.70	68.98	57.15	54.12	51.12

Table 5.3: Image-Text Alignment Validation: Zero-shot image classification is used to assess the alignment accuracy (%) across five benchmark datasets with varying class counts, providing a measure of the relevance and effectiveness of the image-text alignment.

evaluating their ability to generalize to unseen classes. It underscores the importance of the aligned latent space. Results in Table 5.3 highlight superior performance of OneEncoder (OneEncoder-⊕, OneEncoder-⊙) over the baseline (CLIP) across all datasets, suggesting that training large modality-specific encoders may not always be optimal, as demonstrated by the effectiveness of the lightweight OneEncoder framework. We observe that additive fusion with OneEncoder-⊕ yields better results than scaled dot product fusion with OneEncoder-⊙. This phenomenon appears consistently across most experiments, highlighting the impact of the fusion method on OneEncoder representations. A detailed analysis is provided in Section 5.6.

Linear Classification and Fine-Tuning involve adding a linear classifier to a pretrained model, freezing the pretrained weights and training only the linear classifier for linear classification, while training both the pretrained model and the linear classifier for fine-tuning. Linear classification allows for the assessment of the quality of the extracted features from the pretrained model, while fine-tuning simulates the practical use of pretrained weights. In OneEncoder, we always freeze the

modality-specific encoders; in the fine-tuning task, we train only the UP for image and text datasets. In each case (Linear Classification and Fine-Tuning), we train models for 100 epochs without using any data augmentation strategy.

The results presented in Table 5.4 demonstrate the performance of various models on image and

Linear Classification								
Model	**Image Classification**					**Text Classification**		
	CIFAR-10	Oxford-IIIT Pets	CIFAR-100	Caltech-101	Tiny ImageNet	SST-2	TREC	Emotion
ResNet-18 [47]	89.15	84.98	68.10	63.45	59.11	_	_	_
EfficientNet-B0 [109]	89.87	85.12	70.15	64.87	60.27	_	_	_
Swin Transformer [76]	90.17	86.05	**71.12**	65.10	**62.30**	_	_	_
RoBERTa [75]	_	_	_	_	_	76.04	77.34	59.06
DistilBERT [102]	_	_	_	_	_	77.15	76.14	68.11
XLNet [128]	_	_	_	_	_	79.27	78.11	60.10
CLIP [94]	81.21	78.16	60.12	60.14	58.14	80.15	78.24	60.23
OneEncoder-⊕	**90.16**	86.23	70.10	**68.23**	62.12	**82.12**	**79.10**	**63.09**
OneEncoder-⊙	89.18	**86.78**	68.27	65.05	60.10	80.87	78.06	61.89
Fine-Tuning								
Model	**Image Classification**					**Text Classification**		
	CIFAR-10	Oxford-IIIT Pets	CIFAR-100	Caltech-101	Tiny ImageNet	SST-2	TREC	Emotion
ResNet-18 [47]	93.23	90.19	82.37	78.12	67.89	_	_	_
EfficientNet-B0 [109]	94.56	92.23	80.11	79.98	68.10	_	_	_
Swin Transformer [76]	95.27	92.11	**82.02**	79.15	69.09	_	_	_
RoberTa [75]	_	_	_	_	_	83.24	85.45	66.13
DistilBERT [102]	_	_	_	_	_	82.56	83.27	63.15
XLNet [128]	_	_	_	_	_	84.72	85.67	64.11
CLIP [94]	86.76	81.90	70.87	69.67	60.15	85.15	84.24	64.56
OneEncoder-⊕	**96.01**	92.32	81.10	**80.11**	69.12	**86.11**	**86.12**	**67.12**
OneEncoder-⊙	95.98	**93.12**	80.21	78.23	**69.15**	85.12	86.00	66.78

Table 5.4: Linear classification and fine-tuning accuracy (%) on image and text benchmarks. Linear classification trains only a linear classifier with frozen pretrained models, while fine-tuning updates both the classifier and pretrained models. For OneEncoder, only the UP component is trained during fine-tuning, with modality-specific encoders frozen. In contrast, baseline models are fully retrained during fine-tuning.

text classification tasks using two training strategies: linear classification and fine-tuning. These

approaches allow us to evaluate the models' ability to generalize to new data, providing a comprehensive comparison between OneEncoder, CLIP, and other baselines.

In image classification, OneEncoder consistently outperforms the CLIP model, which uses CLIP-ViT on image datasets. For linear classification, OneEncoder-$\oplus$ achieves the highest accuracy on CIFAR-10 (90.16%), Oxford-IIIT Pets (86.23%), and Caltech-101 (68.23%), closely rivaling Swin Transformer, which leads in CIFAR-100 (71.12%) and Tiny ImageNet (62.30%). This highlights the efficiency of OneEncoder, especially considering that it only updates the 4M parameters of the UP, unlike CLIP, which retrains its larger 196M parameters.

In text classification tasks, where CLIP-BERT is used as the baseline for CLIP, OneEncoder again demonstrates superior performance. OneEncoder-$\oplus$ achieves the best results across all datasets: SST-2 (82.12%), TREC (79.10%), and Emotion (63.09%) in the linear classification setup. This shows its robust ability to handle diverse text modalities, outperforming specialized models like RoBERTa, DistilBERT, and XLNet.

The fine-tuning results further emphasize the effectiveness of OneEncoder. For image classification, OneEncoder-$\oplus$ delivers the highest accuracy on CIFAR-10 (96.01%), Oxford-IIIT Pets (92.32%), and Caltech-101 (80.11%), while also performing competitively on Tiny ImageNet (69.12%), narrowly surpassed by Swin Transformer. In text classification, OneEncoder-$\oplus$ achieves the best performance on SST-2 (86.11%), TREC (86.12%), and Emotion (67.12%), surpassing the fine-tuned CLIP-BERT and other text-specific models.

Overall, the results illustrate that OneEncoder, with its efficient training approach and minimal parameter updates, outperforms CLIP and other models in both image and text tasks, demonstrating its superior generalization and adaptability across multiple modalities.

$\mathbf{AL_{audio}}$ Validation Following Text-Audio Modalities Training

After training the UP on image-text modalities, it is frozen and then used for aligning other modalities. Specifically, for audio alignment, only the AL_{audio} with 65,792 parameters is trained within the OneEncoder framework. This process uses a text-audio modality dataset and follows Algorithm 7 on the SRA dataset. For comparison, we also train AudioCLIP [44], an extended version of CLIP that aligns image, text, and audio using ViT for images, BERT for text, and Wav2Vec for audio, with a total of 513M parameters to tune.

Table 5.5 compares the performance of AudioCLIP and OneEncoder (OneEncoder-$\oplus$ and OneEncoder-$\odot$) in text-audio retrieval. This task validates the alignment between text and audio. Evaluated using Top-1 Precision/Recall (P@1, R@1) and mean Average Precision (mAP), OneEncoder consistently outperforms AudioCLIP across all datasets. This highlights OneEncoder's efficient latent space and its ability to handle cross-modal retrieval effectively. Unlike AudioCLIP, which requires extensive encoder training, OneEncoder achieves superior results with a lightweight framework, demonstrating its robustness with minimal dataset-specific training.

Model	**AudioSet**			**UrbanSound8K**			**ESC-50**		
	P@1	R@1	mAP	P@1	R@1	mAP	P@1	R@1	mAP
AudioCLIP [44]	4.27	75.37	27.12	40.10	45.11	78.27	**48.90**	78.21	75.12
OneEncoder-$\oplus$	**5.37**	**76.10**	**28.37**	**41.11**	**46.12**	**79.65**	47.98	**80.12**	**75.57**
OneEncoder-$\odot$	5.10	76.06	28.10	40.89	45.78	79.23	47.87	78.12	74.98

Table 5.5: Performance metrics for text-audio retrieval tasks on the AudioSet, UrbanSound8K, and ESC-50 datasets. The evaluation includes Top-1 Precision (P@1), Top-1 Recall (R@1), and mean Average Precision (mAP) for the models: AudioCLIP, OneEncoder-$\oplus$, and OneEncoder-$\odot$.

To validate transitive alignment between audio and image, we apply the zero-shot classification method as described in Section 5.4.3, replacing text descriptions ("A photo of a {label}.") with corresponding audio. Comparing Table 5.6 with Table 5.3, which uses text descriptions, demonstrates that the OneEncoder framework maintains strong alignment between image and audio, even without direct image-audio alignment. This approach is more efficient and powerful than the

model	CIFAR-10	Oxford-IIIT Pets	CIFAR-100	Caltech-101	Tiny ImageNet
AudioCLIP [44]	61.28	58.15	52.27	51.10	46.04
OneEncoder-⊕	**77.01**	**69.02**	**56.07**	**55.37**	**50.18**
OneEncoder-⊙	74.07	66.56	55.18	53.11	50.06

Table 5.6: Image-Audio Alignment Validation: Zero-shot image classification is used to assess the alignment accuracy (%) across five benchmark datasets with varying class counts, providing a measure of the relevance and effectiveness of the image-audio alignment.

resource-intensive AudioCLIP, offering a cost-effective solution with superior performance.

model	UrbanSound8K	ESC-50
ESResNet [45]	85.42	91.50
AST [41]	_	95.60
ERANN [117]	_	96.10
AudioCLIP [44]	88.32	96.12
OneEncoder-⊕	**89.23**	96.87
OneEncoder-⊙	88.86	**97.02**

Table 5.7: Fine-tuning accuracy (%) on UrbanSound8K and ESC-50 datasets. The table compares baseline models with the proposed OneEncoder variants.

For representation learning model validation, we fine-tune the models on the UrbanSound8K and ESC-50 datasets. Unlike AudioCLIP, which requires retraining all Wav2Vec parameters, OneEncoder only fine-tunes the UP and the (AL_{audio}) for 100 epochs. Table 5.7 shows that OneEncoder-⊕ and OneEncoder-⊙ outperform AudioCLIP on both datasets, with OneEncoder-⊙ achieving the highest accuracy on ESC-50 (97.02%) and OneEncoder-⊕ leading on UrbanSound8K (89.23%). This demonstrates the efficiency of the OneEncoder framework, achieving superior performance with fewer retrained parameters compared to the more resource-intensive AudioCLIP. These results underscore the robustness of OneEncoder for fine-tuned representation learning across diverse audio classification tasks.

$\mathbf{AL_{video}}$ Validation Following Text-Video Modalities Training

After aligning the audio with both image and text modalities (Section 5.4.3), we further integrate the video modality and align it with image, text, and audio. This alignment is performed using Algorithm 7, following a similar approach as in audio alignment, where only the AL_{video} is trained while keeping the UP frozen. The OneEncoder variants are trained for 100 epochs on the MSR-VTT dataset, using the text modality to align with the video modality. This alignment indirectly links the audio and image modalities to the video through transitive alignment.
For evaluating OneEncoder in the context of text-video alignment, we benchmark its performance against X-CLIP [82], an extended version of CLIP designed for text-video alignment.

Results on Table 5.8 demonstrate the superior performance of OneEncoder in aligning text and video across both MSVD and LSMDC datasets. On MSVD, OneEncoder-$\oplus$ outperforms all models with a Recall at rank 5 (R@5) of 80.76 and Mean Rank (MnR) of 7.98 in text-to-video retrieval. Similarly, in video-to-text retrieval, it achieves the best R@5 score (91.62) and the lowest MnR (3.98), surpassing strong baselines like CLIP4Clip and X-CLIP. These results are particularly remarkable given that OneEncoder is based on a lightweight framework and trained on smaller datasets, whereas baselines like X-CLIP are large models trained on extensive datasets. Despite this, OneEncoder achieves comparable performance, which underscores its strong results.

To validate the transitive alignment between audio and video, we convert each text description into audio and perform audio-video retrieval to assess alignment. Table 5.9 compares these results with those in Table 5.8, demonstrating successful audio-video alignment. This confirms the effectiveness of the progressive alignment process, which requires minimal computational resources while maintaining the strong performance of the OneEncoder framework.

OneEncoder outperforms baseline models due to its efficient design. Unlike baselines that train all parameters and require large datasets, OneEncoder only trains the parameters of the UP and AL, reducing the model's complexity and enabling strong performance even with smaller datasets. Its modality-specific alignment allows dynamic adjustment for each modality, capturing

Retrieval performance comparison on MSVD						
Model	**Text-to-Video**			**Video-to-Text**		
	R@1↑	R@5↑	MnR↓	R@1↑	R@5↑	MnR↓
CE [15]	19.8	49.0	-	-	-	-
SSB [16]	28.4	60.0	-	-	-	-
NoiseE [2]	20.3	49.0	-	-	-	-
CLIP-straight [94]	37.0	64.1	-	59.9	85.2	-
Frozen [6]	33.7	64.7	-	-	-	-
TT-CE+ [37]	25.4	56.9	-	27.1	55.3	-
CLIP4Clip-MeanP (ViT-B/32) [79]	46.2	76.1	10.0	56.6	79.7	7.6
CLIP4Clip-seqTransf (ViT-B/32) [79]	45.2	75.5	10.3	62.0	87.3	4.3
CLIP4Clip-MeanP (ViT-B/16) [79]	47.3	77.7	9.1	62.9	87.2	4.2
CLIP4Clip-seqTransf (ViT-B/16) [79]	47.2	77.7	9.1	63.2	87.2	4.2
X-CLIP (ViT-B/32) [82]	47.1	77.8	9.5	60.9	87.8	4.7
X-CLIP (ViT-B/16) [82]	**50.4**	80.6	8.4	**66.8**	90.4	4.2
OneEncoder-⊕	49.21	**80.76**	**7.98**	65.89	**91.62**	**3.98**
OneEncoder-⊙	47.02	79.27	8.88	65.23	89.78	4.65

Retrieval performance comparison on LSMDC						
Model	**Text-to-Video**			**Video-to-Text**		
	R@1↑	R@5↑	MnR↓	R@1↑	R@5↑	MnR↓
CE [15]	11.2	26.9	96.8	-	-	-
MMT [37]	12.9	29.9	75.0	-	-	-
NoiseE [2]	6.4	19.8	-	-	-	-
CLIP-straight [94]	11.3	22.7	-	6.8	16.4	-
MDMMT [31]	18.8	38.5	58.0	-	-	-
Frozen [6]	15.0	30.8	-	-	-	-
HiT [9]	14.0	31.2	-	-	-	-
TT-CE+ [37]	17.2	36.5	-	17.5	36.0	-
CLIP4Clip-MeanP (ViT-B/32) [79]	20.7	38.9	65.3	20.6	39.4	56.7
CLIP4Clip-seqTransf (ViT-B/32) [79]	22.6	41.0	61.0	20.8	39.0	54.2
CLIP4Clip-MeanP (ViT-B/16) [79]	23.5	43.2	54.8	22.6	50.5	50.3
CLIP4Clip-seqTransf (ViT-B/16) [79]	23.5	45.2	51.6	23.2	42.4	47.4
X-CLIP (ViT-B/32) [82]	23.3	43.0	56.0	22.5	42.2	50.7
X-CLIP (ViT-B/16) [82]	26.1	**48.4**	46.7	26.9	46.2	**41.9**
OneEncoder-⊕	**26.12**	48.23	**46.11**	**27.01**	**46.67**	42.3
OneEncoder-⊙	25.32	46.76	50.19	25.67	44.15	42.10

Table 5.8: Retrieval performance comparison on MSVD and LSMDC datasets. Models are evaluated using Recall at Rank 1 (R@1) and Rank 5 (R@5) — higher is better — and Mean Rank (MnR), where lower is better. Results are reported for both text-to-video and video-to-text retrieval tasks.

Retrieval performance comparison on MSVD						
Model	**Audio-to-Video**			**Video-to-Audio**		
	R@1↑	R@5↑	MnR↓	R@1↑	R@5↑	MnR↓
OneEncoder-⊕	46.34	**78.45**	**8.13**	63.43	**89.12**	**4.15**
OneEncoder-⊙	**46.78**	77.32	8.97	**63.78**	87.78	4.98
Retrieval performance comparison on LSMDC						
Model	**Audio-to-Video**			**Video-to-Audio**		
	R@1↑	R@5↑	MnR↓	R@1↑	R@5↑	MnR↓
OneEncoder-⊕	**24.37**	**46.32**	**47.32**	**25.23**	**44.56**	**44.20**
OneEncoder-⊙	23.32	43.13	49.32	23.21	42.12	46.57

Table 5.9: Comparison of audio-to-video and video-to-audio retrieval performance on the MSVD and LSMDC datasets. Performance is evaluated using Recall at Rank 1 (R@1) and Rank 5 (R@5), where higher values are better, and Mean Rank (MnR), where lower values are preferred.

inter-modal relationships more effectively. The two-step training process—aligning image-text pairs (Step 1) and integrating other modalities (Step 2)—improves scalability and adaptability without retraining the entire model. This approach makes OneEncoder computationally efficient, less prone to overfitting, and highly effective in data-constrained environments, while maintaining strong performance across various tasks.

5.4.4 Qualitative Analysis

Figure 5.3 presents qualitative results of OneEncoder across image, text, audio, and video modalities. In Step 1 (see Algorithm 6), we demonstrate that OneEncoder effectively retrieves images using text queries and vice versa, highlighting the UP's ability to understand both visual and textual content, leading to relevant retrievals through well-aligned latent space. In Step 2 (see Algorithm 7), we show that image retrieval via audio inputs generates coherent results, with the frozen UP maintaining alignment across modalities. This phenomenon extends to video retrieval as well, where transitive

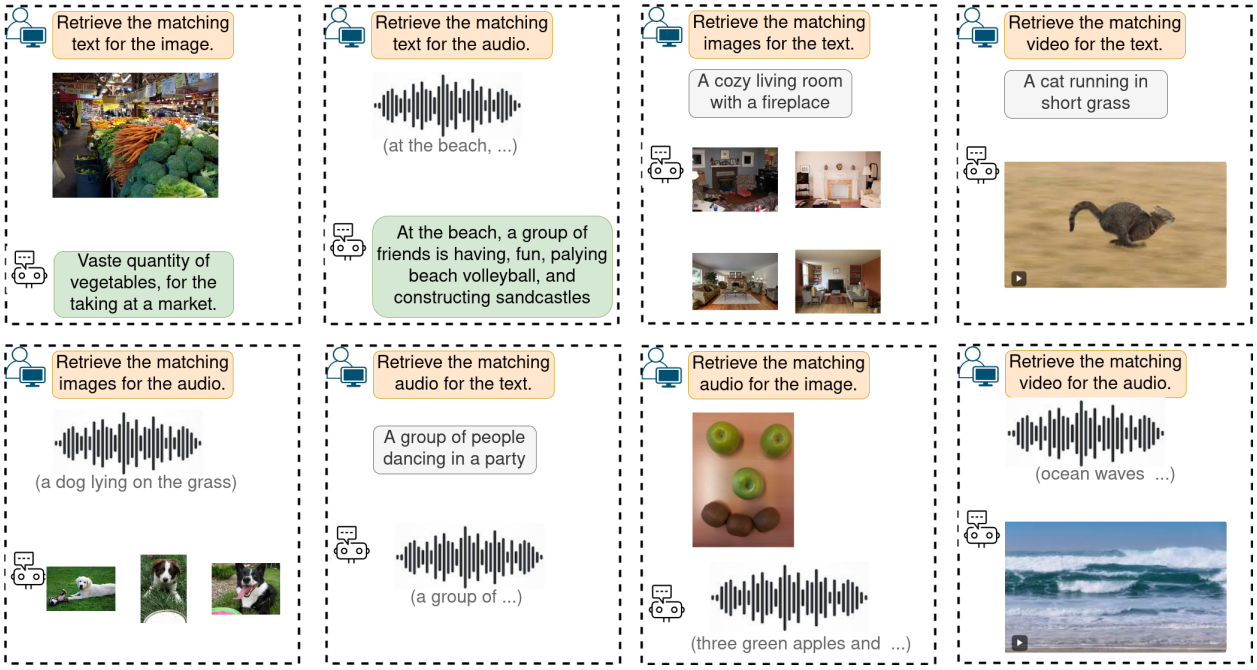


Figure 5.3: Qualitative results showcasing cross-modal retrieval across text, image, audio, and video modalities. For each query, OneEncoder retrieves the most relevant data, highlighting the effectiveness of its cross-modal alignment.

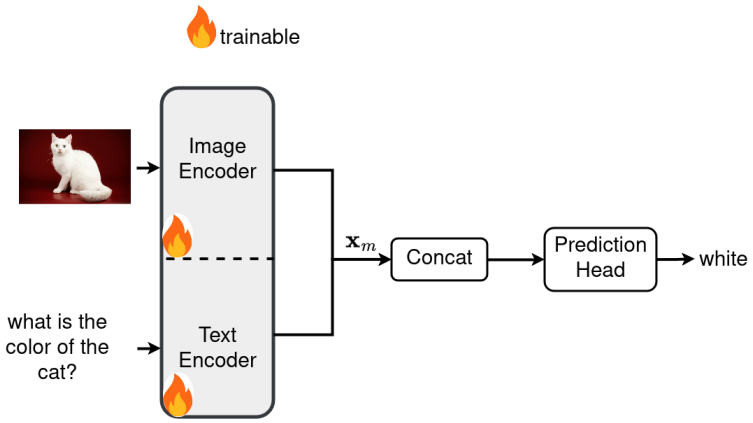
alignment between audio and video also yields accurate and meaningful retrievals. These qualitative results, together with quantitative analysis, underscore OneEncoder's strong performance in progressively aligning modalities. Its lightweight framework efficiently achieves these results even with small aligned datasets, thanks to the use of frozen, pretrained modality-specific encoders.

5.5 OneEncoder on Visual Question Answering

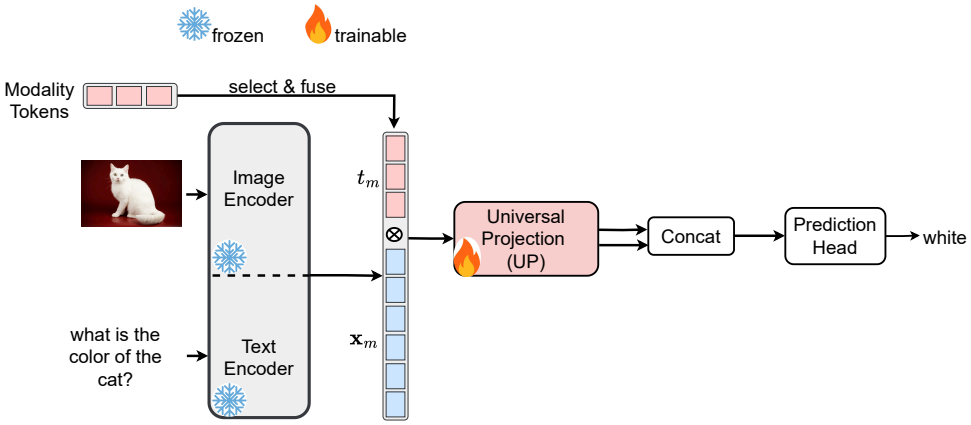
In Section 5.4, we demonstrated that OneEncoder can be efficiently trained using a contrastive learning approach to align multiple modalities at a low computational cost. In this section, we introduce an alternative alignment method tailored for Visual Question Answering (VQA) tasks to further train OneEncoder. The goal is to illustrate the versatility of our proposed framework, showing its ability to be applied across various domains while utilizing different alignment strategies during training.

VQA is a complex task that involves understanding both visual content and textual questions, requiring the model to align and reason across these modalities to generate accurate answers. By employing a specialized alignment mechanism for VQA, we aim to demonstrate OneEncoder's ability to handle cross-modal reasoning tasks beyond retrieval, further highlighting its adaptability across different types of multimodal learning challenges. Figure 5.4 presents a comparison between the classical VQA approach 5.4a and the OneEncoder framework 5.4b. As discussed in 5.4, OneEncoder trains only the UP to align the textual answer with the image and question inputs, significantly reducing the number of parameters compared to the Baseline method 5.4a, which requires training both the image and text encoders. Both methods utilize a "Prediction Head" module to generate the textual answer.

To train both the Baseline and OneEncoder frameworks, we utilize the DAQUAR (Dataset for Question Answering on Real-world images) [84]. For modality-specific encoders, we employ BEiT-base [7], DEiT-base [112], and ViT-base [26] models as image encoders, each with 86M parameters,



(a) **Baseline:** The parameters of both the Image Encoder and Text Encoder are trained.



(b) **OneEncoder:** The parameters of both the Image Encoder and Text Encoder are frozen, with only the parameters of the Universal Projection (UP) module being trained.

Figure 5.4: **OneEncoder architecture for the Visual Question Answering (VQA) task.** The OneEncoder framework in 5.4b trains only the UP to align the textual answer with both the image and the textual question, unlike the baseline method in 5.4a, which trains all specific encoders (image encoder and text encoder), making it more computationally expensive. Both approaches use a "Prediction Head" to generate textual answers.

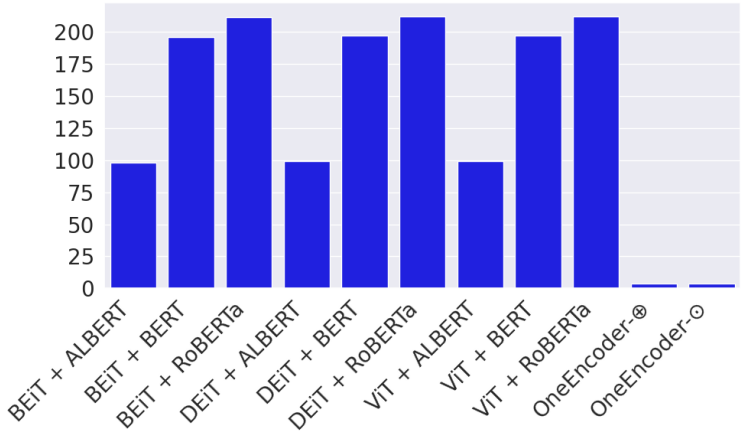
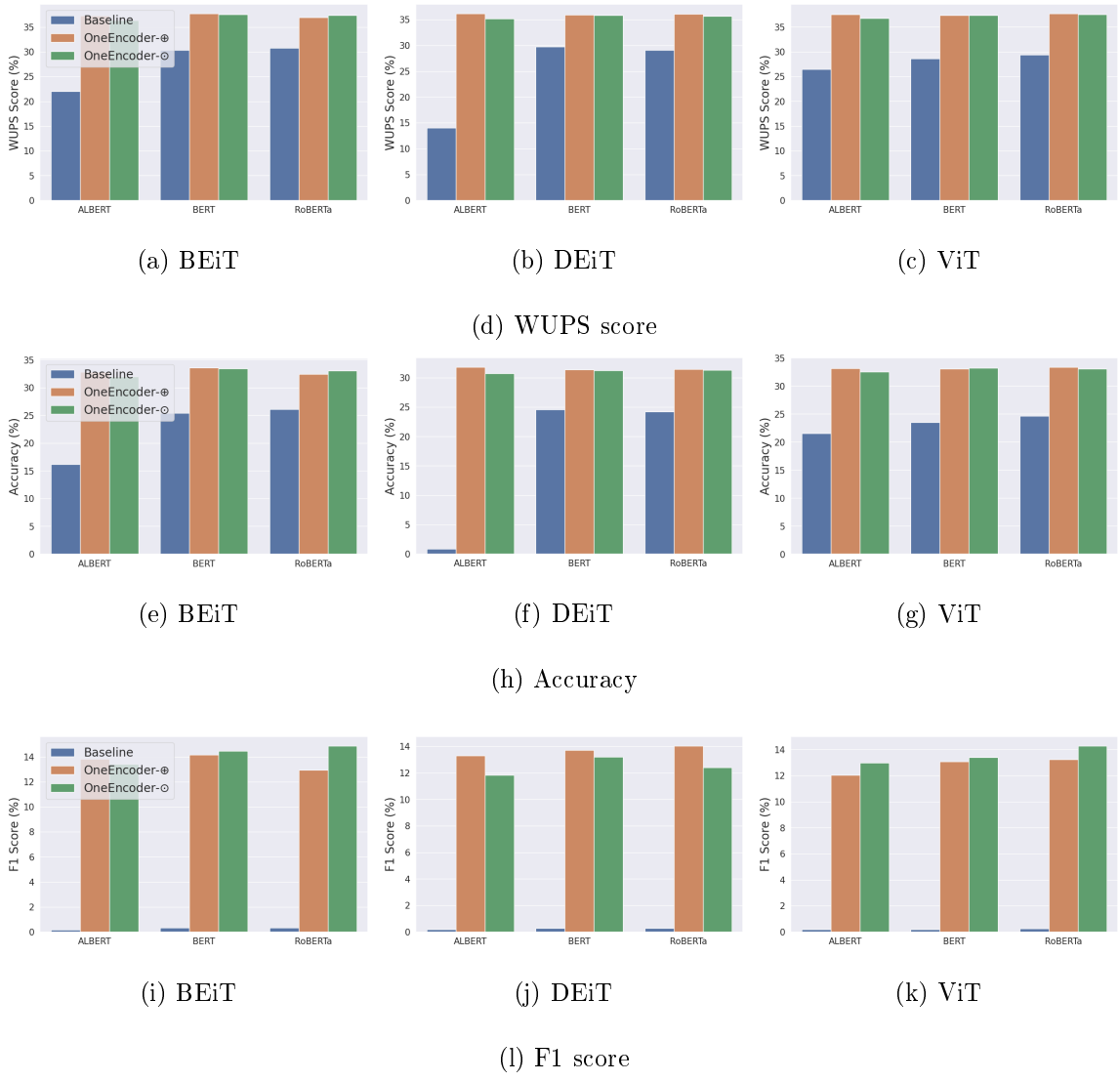


Figure 5.5: Comparison of Trainable Parameters (in millions) between Baseline Models and OneEncoder Variants (OneEncoder-$\oplus$ and OneEncoder-$\odot$).

while ALBERT [66] (60M parameters), BERT-base [25] (110M parameters), and RoBERTa-base [75] (125M parameters) serve as text encoders. We construct 9 VQA models for each method (Baseline and OneEncoder) by combining these encoder pairs: (BEiT, ALBERT), (BEiT, BERT), (BEiT, RoBERTa), (DEiT, ALBERT), (DEiT, BERT), (DEiT, RoBERTa), (ViT, ALBERT), (ViT, BERT), and (ViT, RoBERTa).

Since the DAQUAR dataset features simple vocabulary tokens as answers, we reformulate the task as a classification problem, using a linear layer as the "Prediction Head," where the output dimension matches the vocabulary size, and applying cross-entropy loss. Unlike the Baseline, which fine-tunes the entire pretrained modality-specific encoders, OneEncoder freezes these encoders and focuses solely on training the UP. The goal of this application, using the smaller DAQUAR dataset, is to demonstrate that our framework can achieve strong performance with limited paired data, significantly reducing the number of parameters to optimize and shortening the training time required for convergence. We use four Transformer blocks with a total of 4M parameters for the UP, and modality tokens of size $\mathbb{R}^{1\times768}$. All models are trained for 100 epochs without any data augmentation techniques.



(a) BEiT (b) DEiT (c) ViT

(d) WUPS score

(e) BEiT (f) DEiT (g) ViT

(h) Accuracy

(i) BEiT (j) DEiT (k) ViT

(l) F1 score

Figure 5.6: Validation Performance of Baseline Models and OneEncoder Variants (OneEncoder-⊕, OneEncoder-⊙) on the DAQUAR dataset, evaluated using Wu-Palmer Similarity (WUPS), Accuracy, and F1 Score.

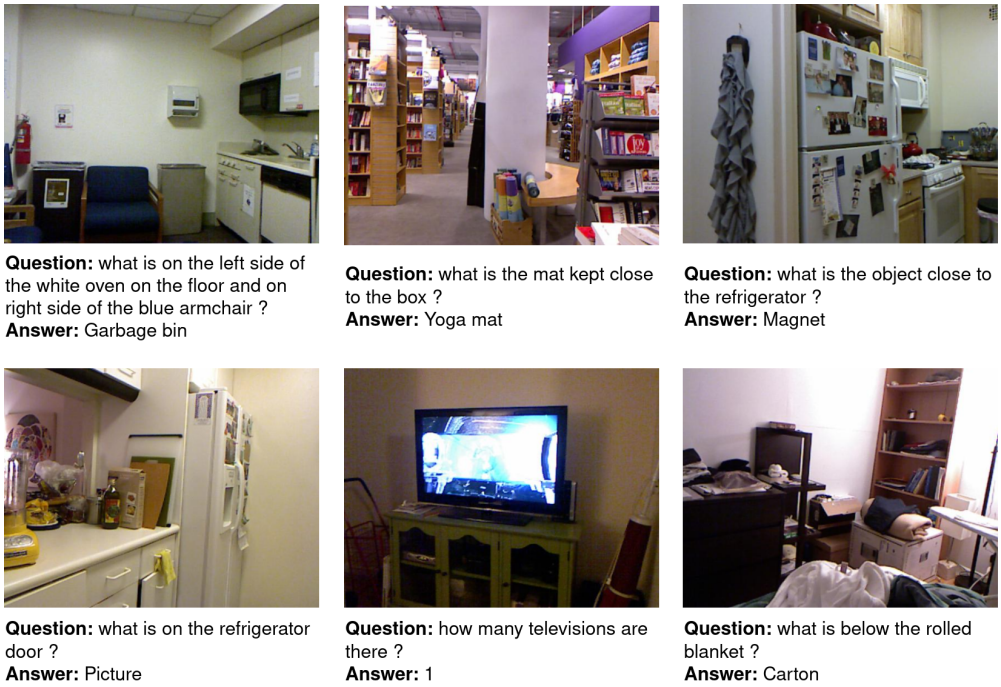


Figure 5.7: Example VQA Results Using the OneEncoder-$\oplus$ Model.

Figure 5.5 provides a detailed comparison of the number of trainable parameters between Baseline models and OneEncoder variants. Specifically, OneEncoder-$\oplus$ utilizes addition-based fusion, while OneEncoder-$\odot$ employs an attention-based fusion mechanism. Unlike the Baseline models, which train all parameters, the OneEncoder versions use Baseline models for feature extraction but keep them frozen during training.

Figure 5.6 demonstrates that the OneEncoder architecture (OneEncoder-$\oplus$, OneEncoder-$\odot$) consistently outperforms baseline models across the three key metrics: Wu-Palmer Similarity (WUPS) [125], Accuracy, and F1 Score. These results indicate that retraining specialized encoders may not be essential for achieving strong performance. By freezing the encoders and only training the UP on a small paired dataset, we can significantly reduce the number of parameters to optimize, minimize the need for large datasets, and shorten training times—all while yielding superior outcomes as illustrated in Figure 5.7.

The VQA experiment further validates the findings in Section 5.4, focused on contrastive learning. OneEncoder, with its efficient and lightweight design, can be effectively integrated into any alignment-based approach, reducing parameter complexity, data requirements, and surpassing traditional methods that rely on retraining modality-specific encoders.

5.6 Discussion: Addition vs. Cross-Attention Fusion in OneEncoder

In our experiments, we evaluated two distinct fusion strategies for integrating modality features: the simple addition approach used by OneEncoder-$\oplus$ and the cross-attention mechanism implemented in OneEncoder-$\odot$. The results revealed a consistent trend where OneEncoder-$\oplus$ outperformed OneEncoder-$\odot$ across a range of tasks, providing insights into how the different fusion methods influence model behavior and performance.

5.6.1 OneEncoder-$\oplus$: Simple Addition for Modality Integration

OneEncoder-$\oplus$ uses a parameter-free addition operation to integrate modality features and tokens. This approach is simple, direct, and efficient, as it combines modality features with modality tokens through a straightforward summation process. The lack of additional learnable parameters allows OneEncoder-$\oplus$ to preserve the integrity of the feature representations, enabling the model to retain more information from each modality. This fusion strategy appears to be more stable across various tasks, possibly because it avoids the complexities of training additional parameters that could introduce variability or overfitting. Moreover, the simplicity of the addition mechanism helps the model focus on the core information from each modality without being distracted by complex inter-modality relationships.

5.6.2 OneEncoder-⊙: Cross-Attention for Dynamic Modality Interaction

In contrast, OneEncoder-⊙ utilizes a cross-attention mechanism, which is more flexible and powerful in its ability to model complex interactions between modality features. By using modality tokens as queries and modality features as keys and values, OneEncoder-⊙ enables the model to dynamically adjust its focus between the two modalities, potentially learning intricate inter-modal relationships. However, this flexibility comes at the cost of introducing learnable parameters within the attention mechanism. These parameters, while allowing the model to better capture interactions, can also introduce instability in the learning process. The query-key-value structure requires careful optimization to ensure that the interactions between modalities are meaningfully learned. This may be particularly challenging with limited data, where the model might struggle to fully realize the potential of cross-attention

5.6.3 Key Insights from Experimentation

Our experimental findings suggest that the additional complexity introduced by OneEncoder-⊙'s attention mechanism may not always translate into better performance. While cross-attention offers greater expressiveness, the potential for instability and the need for more extensive training data can hinder its effectiveness, particularly when compared to OneEncoder-⊕'s straightforward addition strategy. OneEncoder-⊕'s direct integration of modality features allows the model to focus on the most salient aspects of each modality without the added burden of learning complex inter-modal relationships.

5.7 Discussion

We introduce OneEncoder, a novel approach to multimodal representation learning that leverages prior knowledge about modality-specific characteristics to streamline the learning process. By incorporating this information into data representation, OneEncoder reduces both the reliance on large paired datasets and the number of parameters to tune, addressing two critical challenges in

multimodal system design: scalability and training efficiency.

The core idea of OneEncoder lies in utilizing pretrained, modality-specific encoders as fixed feature extractors, thus retaining the inherent strengths of each modality's prior representations. A lightweight Universal Projection (UP) module, shared across all modalities, facilitates the alignment of these diverse representations within a unified space. Importantly, OneEncoder incorporates a modality token—a learned embedding indicating the origin of each representation—before the projection step. This modality token encodes prior modality knowledge, ensuring that the UP can consistently and effectively map diverse inputs to a common space without retraining the entire architecture for each new task.

For contrastive learning, OneEncoder achieves effective alignment of text and image embeddings within the same projection space. Notably, this is accomplished without training separate modality-specific modules, as required by more resource-intensive models like CLIP. This design proves particularly advantageous on smaller datasets, where OneEncoder demonstrates superior performance while requiring fewer computational resources. Furthermore, OneEncoder's framework is naturally extensible to additional modalities such as audio and video through a progressive alignment strategy. In this strategy, the UP remains fixed, and a compact Alignment Layer module is introduced to adapt the output of pretrained feature extractors into the UP-compatible space. This approach offers a highly scalable and flexible solution for multimodal learning, further reducing the need for large-scale retraining while retaining compatibility across diverse modalities.

The versatility of OneEncoder is further exemplified through its application to tasks like visual question answering (VQA). Here, OneEncoder not only achieves improved performance over baseline models but does so with significantly lower training costs. This underscores the effectiveness of leveraging prior modality-specific knowledge and compact, shared representation spaces in reducing computational overhead.

In summary, OneEncoder represents a paradigm shift in multimodal representation learning by directly integrating prior knowledge about modality into the data representation process. This enables it to significantly reduce the need for extensive paired datasets and large parameter sets while maintaining strong performance across tasks. The insights gained from this approach are a step forward in addressing the scalability challenges of multimodal systems.

The next part, Part III, will explore the application of OneEncoder to open-vocabulary object detection. This involves identifying relevant concepts from rich semantic prompts before performing object detection, showcasing the potential of OneEncoder to extend its lightweight, scalable framework to complex downstream tasks.

Conclusion

In this part, we explored the challenges and solutions in multimodal representation learning, focusing on the limitations of large paired datasets and the complexity of training systems that align multiple modalities. In the related work, we reviewed key techniques in cross-modal alignment, highlighting the success of models like CLIP and ALIGN, which integrate dual-modalities but struggle with scalability and efficiency due to their reliance on extensive datasets and separate modality-specific modules.

To address these challenges, we introduced OneEncoder, a novel approach that integrates prior knowledge of modality-specific characteristics to streamline multimodal learning. By utilizing pre-trained, fixed feature extractors and a lightweight Universal Projection (UP) module, OneEncoder reduces the computational burden and eliminates the need for training separate encoders for each modality. The inclusion of a modality token ensures that diverse inputs are aligned in a unified space without retraining the entire model for new tasks. OneEncoder's flexibility allows for seamless integration of new modalities, such as audio and video, through a progressive alignment strategy.

In contrast to traditional models, OneEncoder excels with fewer resources, delivering strong performance on smaller datasets, such as visual question answering (VQA), while minimizing training costs. This makes OneEncoder a scalable, efficient, and adaptable solution to the growing demand for multimodal systems.

Part III

Open-Vocabulary Object Detection

This part builds upon the work presented in Part II, focusing on the challenges of **high costs**, **data limitations**, and **scalability** in training open-vocabulary object detection systems. To address these issues, we introduce LightMDETR, a novel method designed to significantly reduce training costs while maintaining high performance. This method leverages prior knowledge, as seen in OneEncoder (discussed in Part II), through the use of the Universal Projection module, which allows for efficient adaptation to unseen object categories.

The part is organized as follows: Chapter 6 presents a detailed review of object detection techniques, encompassing both classical methods and open-vocabulary approaches. Chapter 7 introduces LightMDETR, emphasizing its improvements over existing frameworks by integrating prior knowledge to reduce computational costs while maintaining robust detection across diverse object categories. The proposed method is validated on three tasks: **phrase grounding**, **referring expression comprehension**, and **referring expression segmentation**.

Chapter 6

State of the Art in Object Detection

6.1 Introduction

Object detection is a crucial task in computer vision, focused on identifying and localizing objects within images. Leading methods like Faster R-CNN [97], YOLO [95], and SSD [72] have shown great success in this domain. However, these approaches are constrained by a fixed set of object categories (e.g., 20 categories in the PASCAL VOC [32] dataset). Once trained, these detectors can only recognize the predefined categories, limiting their flexibility and applicability in more open and dynamic scenarios.

Recent works [135, 105, 43, 27] have leveraged popular vision-language models for open-vocabulary detection by distilling vocabulary knowledge from language encoders. However, these distillation-based approaches face significant limitations due to the scarcity of diverse training data.

Inspired by the success of methods [94, 55, 133] that learn image-level visual representations from large-scale raw image-text pairs, achieving semantically rich projection spaces for easy transfer to downstream tasks (such as zero-shot image classification and text-image retrieval), several approaches [58, 70, 43, 141, 86] have extended this to open-vocabulary object detection, aiming for fine-grained image understanding with object-level visual representations.

6.2 Traditional Object Detection

Traditional object detection methods have undergone a significant evolution, starting with frameworks that utilized separate networks for classification and localization, progressing to unified archi-

tectures optimized for both computational efficiency and accuracy. Early methods like R-CNN [40] relied on a two-stage process, where the first stage generated region proposals using algorithms like Selective Search, and the second stage performed feature extraction using a Convolutionnal Neural Network (CNN). Bounding box regression ($\mathbf{t}$) and classification (c) were treated as separate tasks. The localization task optimized bounding box coordinates (x, y, w, h) using a regression loss, often defined as Smooth L1 loss:

$$\mathcal{L}_{\text{loc}} = \sum_{i \in \{x,y,w,h\}} \text{smooth}_{L1}(t_i - \hat{t}_i), \tag{6.1}$$

where t_i and $\hat{t}_i$ represent the ground truth and predicted box parameters, respectively. Classification is optimized using a cross-entropy loss ($\mathcal{L}_{\text{cls}}$), leading to the combined loss:

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda \mathcal{L}_{\text{loc}}, \tag{6.2}$$

where λ is a hyperparameter balancing the two terms. Despite its accuracy, R-CNN is computationally expensive due to redundant feature extraction. Fast R-CNN [**girshick2015fast**] improved efficiency by sharing feature maps across proposals using RoI pooling, while Faster R-CNN [97] further streamlined the process by introducing a Region Proposal Network (RPN). The RPN generates object proposals by sliding a small network over the feature map, predicting objectness scores (p) and bounding box deltas $(\Delta x, \Delta y, \Delta w, \Delta h)$ for predefined anchor boxes. The RPN loss function combines objectness classification and regression:

$$\mathcal{L}_{\text{RPN}} = \frac{1}{N_{\text{cls}}} \sum_i \mathcal{L}_{\text{cls}}(p_i, p_i^*) + \frac{\lambda}{N_{\text{reg}}} \sum_i p_i^* \mathcal{L}_{\text{loc}}(\mathbf{t}_i, \mathbf{t}_i^*), \tag{6.3}$$

where p_i^* is the ground truth label indicating whether the anchor is positive, and N_{cls} and N_{reg} are normalization factors.

Single-stage detectors like YOLO [95] and SSD [72] aimed to simplify the pipeline by predicting object classes and bounding boxes directly from the feature map, removing the need for region proposals. YOLO divides the image into a grid of size $S \times S$, where each grid cell predicts B bounding boxes and C class probabilities. The YOLO loss function combines classification, localization, and object confidence terms:

$$\mathcal{L}_{\text{YOLO}} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \Big[\mathcal{L}_{\text{coord}} + \mathcal{L}_{\text{conf}} + \mathcal{L}_{\text{cls}} \Big], \tag{6.4}$$

where $\mathbb{1}_{ij}^{\text{obj}}$ is an indicator for the presence of an object in cell i and box j. SSD enhanced this by introducing anchor boxes at multiple scales and aspect ratios, predicting class probabilities and bounding box offsets for each anchor.

To handle challenges like class imbalance, RetinaNet introduced *Focal Loss*, which modifies cross-entropy to focus learning on hard-to-classify examples:

$$\mathcal{L}_{\text{Focal}} = -\alpha_t (1 - p_t)^{\gamma} \log(p_t), \tag{6.5}$$

where p_t is the predicted probability for the target class, γ adjusts the focus on hard examples, and α_t balances positive and negative samples.

More recently, transformer-based architectures like DETR [11] have redefined object detection. DETR replaces anchor-based mechanisms with *learnable object queries*, using a transformer encoder-decoder [116] to match queries with objects in the image. DETR optimizes a combination of classification and bounding box regression, using a Hungarian matching cost for alignment:

$$\mathcal{L}_{\text{DETR}} = \sum_{i=1}^{N} \left[\mathcal{L}_{\text{cls}}(c_i, \hat{c}_i) + \mathbb{1}_i^{\text{obj}} \mathcal{L}_{\text{box}}(\mathbf{b}_i, \hat{\mathbf{b}}_i) \right], \tag{6.6}$$

where N is the number of object queries, c_i is the true class, and $\mathbf{b}_i$ represents bounding box coordinates. Deformable DETR [142] refined this by focusing attention on sparse, key regions, improving convergence speed and computational efficiency.

Through these advancements, object detection has transitioned from labor-intensive multi-stage pipelines to efficient, unified systems leveraging innovative loss functions, multi-scale feature learning, and attention mechanisms, continually improving scalability, speed, and accuracy.

While these models are highly effective within their defined scope, their limitation lies in their inability to generalize beyond the fixed set of categories, making them less adaptable in dynamic environments.

This limitation has paved the way for open-vocabulary object detection methods, driven by advances in models like CLIP [94].

6.3 Open-vocabulary Object Detection (OVD)

OVD builds on traditional object detection techniques by enabling the identification and localization of objects from a large set of categories, including those that were not present during training. Model trained separately on the concepts "cat" and "white," can infer and detect a "white cat" during inference by dynamically combining learned embeddings of "cat" and "white" in a shared vision-language space. This capability is achieved by leveraging external knowledge, such as pre-trained language models, to generalize the detection task to unseen classes. In contrast to classical methods, which require each object class to be defined during training, OVD systems aim to predict new classes by associating visual features with textual descriptions or embeddings of unseen objects.

Early Approaches and Region-Based OVD

Traditional object detection pipelines, such as Faster R-CNN, perform detection by generating region proposals through a Region Proposal Network (RPN), followed by bounding box regression and class prediction through fully connected layers. The output is typically a fixed set of categories that the model was trained on. To extend this to open-vocabulary detection, methods like Vision-Language Detectors (ViLD) [43] replace the fixed classification layer with a mechanism that uses text embeddings from pre-trained language models such as CLIP.

In ViLD, the model first generates region proposals via RPN and computes visual feature representations for each region. These visual features $\mathbf{f}_i$ are then compared to class embeddings $\mathbf{t}_j$ obtained from a pre-trained vision-language model, typically CLIP. The similarity between the visual feature and the text embedding for each class is computed using a similarity function such as cosine similarity:

$$s_{ij} = \text{sim}(\mathbf{f}_i, \mathbf{t}_j) \cdot p_{\text{obj}}(\mathbf{f}_i), \tag{6.7}$$

where $\text{sim}(\mathbf{f}_i, \mathbf{t}_j)$ denotes the cosine similarity between the visual feature $\mathbf{f}_i$ and the text embedding $\mathbf{t}_j$, and $p_{\text{obj}}(\mathbf{f}_i)$ is the objectness score (i.e., the likelihood that the region contains an object). The model is trained using a combination of a bounding box regression loss $\mathcal{L}_{\text{box}}$ and a contrastive loss $\mathcal{L}_{\text{sim}}$, which encourages the visual features to be closer to the correct textual embedding:

$$\mathcal{L} = \mathcal{L}_{\text{box}} + \lambda \mathcal{L}_{\text{sim}}, \tag{6.8}$$

where λ is a hyperparameter balancing the two losses. This formulation allows the model to recognize new object categories as long as there exists a textual embedding for those categories.

Grounded Language-Image Pre-training (GLIP)

GLIP [70] is another method that improves open-vocabulary object detection by pre-training a model with a joint objective of grounding language (associating textual descriptions with visual features) and detecting objects. Unlike ViLD, which primarily relies on CLIP embeddings, GLIP uses a grounding loss that explicitly aligns regions in an image with their corresponding text descriptions. During training, the model is provided with image-caption pairs and learns to align image regions with corresponding words from the caption. The grounding loss used in GLIP is based on contrastive learning, where the model learns to associate each image region with the correct textual description.

The GLIP model uses a Vision Transformer (ViT) [26] backbone to extract features from an image, and each image region is matched with a textual embedding via cosine similarity. The total loss function consists of a bounding box regression loss $\mathcal{L}_{\text{box}}$, a classification loss $\mathcal{L}_{\text{cls}}$, and the grounding contrastive loss $\mathcal{L}_{\text{grounding}}$, which is computed as follows:

$$\mathcal{L}_{\text{grounding}} = -\sum_{j} \log \frac{\exp(\text{sim}(\mathbf{f}_i, \mathbf{t}_j))}{\sum_k \exp(\text{sim}(\mathbf{f}_i, \mathbf{t}_k))}, \tag{6.9}$$

where $\mathbf{f}_i$ is the visual feature vector for region i, and $\mathbf{t}_j$ is the embedding for word j. The grounding loss ensures that the correct textual description is closer in the embedding space to the visual features of the corresponding region. This approach allows the model to generalize well to unseen categories by relying on textual descriptions, even for categories that were not included in the training set.

Modulated Detection Transformer (MDETR)

MDETR [58] is a transformer-based architecture designed for open-vocabulary object detection, where the input consists of both image patches and textual descriptions. The key innovation of MDETR is its use of a transformer encoder-decoder architecture that processes both modalities simultaneously, allowing the model to reason about the relationships between the visual and textual information. MDETR uses a tokenized representation of the input text, where each word or phrase is transformed into a fixed-length embedding. The image is split into patches, which are processed by the transformer encoder along with the text tokens. The resulting embeddings are used to pre-

dict both bounding boxes and class labels for detected objects.

MDETR extends traditional object detection by using a *modulated attention mechanism*, where textual embeddings are used to modulate the attention weights in the visual feature extraction process. This allows the model to focus more on image regions that are relevant to the provided textual descriptions. The output of the model is a set of bounding boxes and class labels, where the class labels are predicted based on the similarity between visual features and textual embeddings. The overall loss function combines bounding box regression $\mathcal{L}_{\text{box}}$, classification $\mathcal{L}_{\text{cls}}$, and a contrastive loss $\mathcal{L}_{\text{contrastive}}$, which ensures alignment between the visual and textual modalities.

Open-World Learning Vision Transformer (OWL-ViT)

OWL-ViT further advances OVD by using a vision transformer architecture pre-trained on large-scale image-text datasets like CLIP. This model is designed to recognize both seen and unseen object categories by associating image patches with text descriptions in a shared embedding space. OWL-ViT adapts the traditional object detection pipeline by incorporating a large number of potential object categories, not limited to those seen during training. The transformer architecture is capable of handling varying levels of semantic ambiguity, and the model's contrastive objective helps it generalize to novel categories by learning better alignment between image features and textual descriptions.

The loss function in OWL-ViT combines the standard object detection losses (bounding box regression and classification) with a contrastive loss that forces the visual features to be close to the correct textual embeddings for both seen and unseen categories. The model's ability to process large-scale text-image data enables it to detect objects from classes that were not part of the training set, making it a highly effective solution for open-vocabulary object detection.

The evolution of open-vocabulary object detection has been a progressive integration of vision-language models into traditional object detection pipelines. From methods like ViLD, which use CLIP embeddings to generalize to new categories, to transformer-based models like MDETR and OWL-ViT, which jointly process visual and textual modalities, each step has made it possible to detect a wider array of objects without requiring explicit retraining for every new class. The key mathematical principles across these methods involve aligning visual features with text embeddings via contrastive losses, making OVD a highly flexible and scalable approach for real-world object

detection tasks.

Despite their superior generalization capabilities, these open-vocabulary methods are resource-intensive, requiring substantial computational power and large-scale datasets for training, primarily due to the reliance on extensive pre-trained models for text and image encoding. Nonetheless, they represent a significant advancement in object detection, offering the ability to detect a vast range of objects, including those unseen during training.

To tackle the challenges associated with the extensive training required for open-vocabulary object detection methods, we propose a new method based on *prior knowledge*, that significantly reduces training demands while maintaining performance. Our approach can be seamlessly integrated into any existing open-vocabulary object detection system, ensuring more efficient training without compromising the model's effectiveness.

6.4 Discussion

Open-vocabulary object detection (OVD) represents a significant leap forward in the field of computer vision, offering the ability to detect a wide variety of objects, including those not present during training. By leveraging powerful vision-language models like CLIP, OVD systems can generalize detection tasks to unseen categories by associating visual features with textual descriptions. The integration of transformer-based architectures, such as MDETR and OWL-ViT, enhances this capability by allowing the model to process both visual and textual information simultaneously, thereby improving detection accuracy. However, challenges remain, particularly regarding computational efficiency, as these models require extensive resources for training and inference. Additionally, the alignment between visual features and textual embeddings remains a complex task, and scalability becomes an issue as the number of potential object categories increases. Despite these challenges, OVD offers immense potential for real-world applications, and future research could focus on optimizing model architectures, incorporating prior knowledge, and reducing the need for large-scale retraining to make these systems more efficient and adaptable.

Chapter 7

LightMDETR

7.1 Introduction

Open-vocabulary object detection methods face limitations primarily due to the extensive training requirements needed to align visual and textual embeddings effectively. Training often involves large-scale datasets and sophisticated vision-language models like CLIP, requiring significant computational resources and time. This process can be particularly challenging when attempting to balance generalization across unseen categories while maintaining accuracy on seen categories. Additionally, the reliance on massive pre-training makes it difficult to adapt these methods to domain-specific tasks or smaller datasets without considerable fine-tuning efforts.

To address this challenge, we propose a *Lightweight Modular Framework for Low-Cost Open-Vocabulary Object Detection Training*. Similar to the approaches introduced in Parts I and II, our method leverages prior knowledge to unify the parameter representation of distinct modalities, such as image and text. This unified representation minimizes the number of parameters that need to be fine-tuned during training, significantly reducing computational overhead. To validate the efficacy of the proposed framework, we integrate it into MDETR, a state-of-the-art model for multimodal detection.

In addition to enhancing general object detection, our proposed framework excels in tasks like **Phrase Grounding**, **Referring Expression Comprehension**, and **Referring Expression Segmentation**. These applications involve identifying and localizing objects in images based on

textual descriptions, ranging from simple phrases to complex referring expressions. Our method's ability to leverage a unified representation and lightweight architecture reduces computational complexity while maintaining high accuracy across these tasks:

- **Phrase Grounding:** The framework enables efficient grounding of textual phrases to corresponding image regions, allowing for accurate mapping even in challenging open-vocabulary scenarios.
- **Referring Expression Comprehension:** By aligning visual and textual modalities, the system improves comprehension of textual descriptions and enhances localization performance, especially for unseen or ambiguous expressions.
- **Referring Expression Segmentation:** Our lightweight architecture extends its capabilities to pixel-level segmentation tasks, enabling precise identification and segmentation of objects described in text with minimal additional computational costs.

To validate its performance, we demonstrate the integration of our framework into MDETR, showcasing its ability to lower training costs while maintaining or improving performance on tasks such as phrase grounding and referring expression tasks. The chapter is organized as follows: Section 7.2 provides a detailed overview of MDETR, while Section 7.3 presents our proposed method, showcasing its integration into MDETR to lower training costs and improve performance.

7.2 MDETR

MDETR is built on the traditional object detection system DETR [11]. DETR is an end-to-end object detection model built with a convolutional residual network backbone and a Transformer Encoder-Decoder [116] architecture. The encoder processes flattened 2D image features from the backbone, while the decoder uses learned object queries, which serve as slots to detect objects in the image. Through cross-attention, the decoder predicts embeddings for each object query, which are then decoded into bounding boxes and class labels. DETR is trained using Hungarian matching to align the predicted objects with ground-truth, utilizing a combination of L1 loss and Generalized

IoU [115] for bounding box supervision.

MDETR extends DETR by integrating both visual and textual information into a unified framework. Unlike DETR, which classifies objects into fixed categories, MDETR associates detected objects with spans of text. It encodes images using a ResNet [47] backbone and text via a pre-trained language model (RoBERTa [75]), projecting both into a shared embedding space (ref. Fig. 7.1). These features are concatenated and processed through a joint transformer encoder. The transformer decoder then cross-attends to this combined representation, predicting object bounding boxes linked to the text.

For training, MDETR employs two additional key loss functions to align image and text data the **soft token prediction loss** and the **contrastive alignment loss**. The soft token prediction loss ($\mathcal{L}_{soft\_token}$) guides the model to predict a uniform distribution over the tokens in the text that correspond to each detected object, rather than predicting discrete class labels. Given a maximum

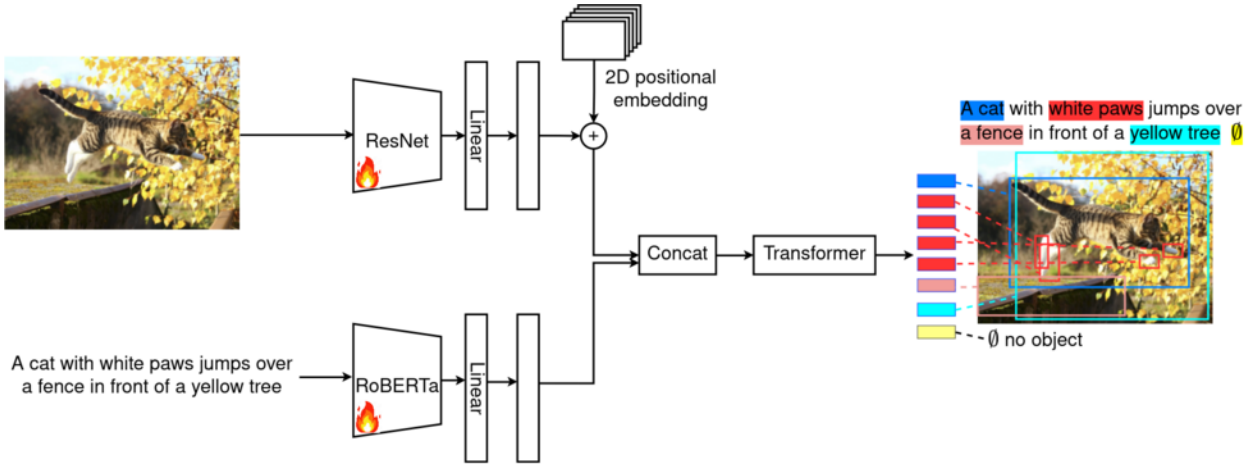


Figure 7.1: MDETR Architecture: Visual features are extracted via ResNet and textual features through RoBERTa. Both are projected into a shared embedding space, concatenated, and processed by a transformer encoder-decoder, which predicts object bounding boxes and their alignment with the text.

token length L and a set of predicted bounding boxes, the loss for each object is computed by predicting the probability distribution over possible token positions. Specifically, if o_i represents the embedding of the i-th object and t_j denotes the j-th token, the soft token prediction loss is designed to minimize the discrepancy between predicted token spans and the true token spans in the

text. The **contrastive alignment loss** enforces that the embeddings of visual objects and their corresponding text tokens are closely aligned in the feature space. This loss is calculated using:

$$\mathcal{L}_o = \frac{1}{N} \sum_{i=0}^{N-1} \frac{1}{|T_i^+|} \sum_{j \in T_i^+} -\log \left(\frac{\exp(o_i^\top t_j / \tau)}{\sum_{k=0}^{L-1} \exp(o_i^\top t_k / \tau)} \right) \tag{7.1}$$

$$\mathcal{L}_t = \frac{1}{L} \sum_{i=0}^{L-1} \frac{1}{|O_i^+|} \sum_{j \in O_i^+} -\log \left(\frac{\exp(t_i^\top o_j / \tau)}{\sum_{k=0}^{N-1} \exp(t_i^\top o_k / \tau)} \right) \tag{7.2}$$

where τ is a temperature parameter set to 0.07, T_i^+ is the set of tokens aligned with the i-th object, and O_i^+ is the set of objects aligned with the i-th token. The total loss is the average of these two components:

$$\mathcal{L}_{contrast} = \frac{1}{2}(\mathcal{L}_o + \mathcal{L}_t) \tag{7.3}$$

The overall training loss for MDETR combines the bounding box losses (L1 and GIoU), soft token prediction loss, and contrastive alignment loss:

$$\mathcal{L}_{total} = \mathcal{L}_{bbox} + \mathcal{L}_{soft\_token} + \mathcal{L}_{contrast} \tag{7.4}$$

with

$$\mathcal{L}_{bbox} = \mathcal{L}_{L1} + \mathcal{L}_{GIoU} \tag{7.5}$$

where $\mathcal{L}_{L1}$ is the L1 loss calculated as:

$$\mathcal{L}_{L1} = \frac{1}{N} \sum_{i=1}^{N} \|\hat{b}_i - b_i\|_1 \tag{7.6}$$

and $\mathcal{L}_{GIoU}$ is the Generalized Intersection over Union loss:

$$\mathcal{L}_{GIoU} = 1 - \text{IoU} + \frac{\text{area}(C - (A \cup B))}{\text{area}(C)} \tag{7.7}$$

where $\hat{b}_i$ and b_i are the predicted and ground truth bounding boxes, respectively, and C is the smallest enclosing box covering both A and B.

Training the pretrained feature extractors ResNet and RoBERTa, as depicted in Figure 7.1, is both unnecessary and costly. To address this challenge, a lightweight modular framework is proposed, designed to be seamlessly integrated into any open-vocabulary object detection system. This

framework reduces training costs by minimizing the number of tunable parameters while maintaining or enhancing the performance of the baseline object detector. The core innovations of this approach include freezing the backbone of pretrained models and introducing a "Universal Projection" (UP) module that shares parameters to represent both visual and language data. To ensure the UP effectively processes data from different distributions (visual and language) using the same parameters, a learnable "modality token" is incorporated, enabling efficient switching between the two modalities. This framework is applied to MDETR, resulting in Lightweight MDETR (LightMDETR), whose efficacy is validated on tasks such as phrase grounding, referring expression comprehension, and segmentation.

The main contributions of this work are as follows:

- A lightweight approach for open-vocabulary object detection systems is introduced, significantly reducing the number of parameters to tune, thereby improving training efficiency.
- This approach is applied to the MDETR architecture, resulting in two variants: LightMDETR, which trains only the UP module, and LightMDETR-Plus, which extends LightMDETR with a cross-fusion layer between text and image modalities to enhance representation capabilities.
- The framework achieves its efficiency by training only the UP module while freezing all pretrained specialized backbone models for images and text. The inclusion of the "modality token" within the UP module enables effective switching between image and text modalities.

7.3 LightMDETR

We depict the LightMDETR architecture in Fig. 7.8. The image is encoded by a frozen ResNet backbone, producing feature vectors O, while the text is encoded by a frozen RoBERTa model, yielding feature vectors T. Both image and text features are projected into a shared embedding space, with the "Universal Projection" (UP) module being the only trained component in the backbone. The UP acts as a lightweight encoder, adapting the frozen feature representations for the target task. To handle both modalities, an early fusion method combines the features with a

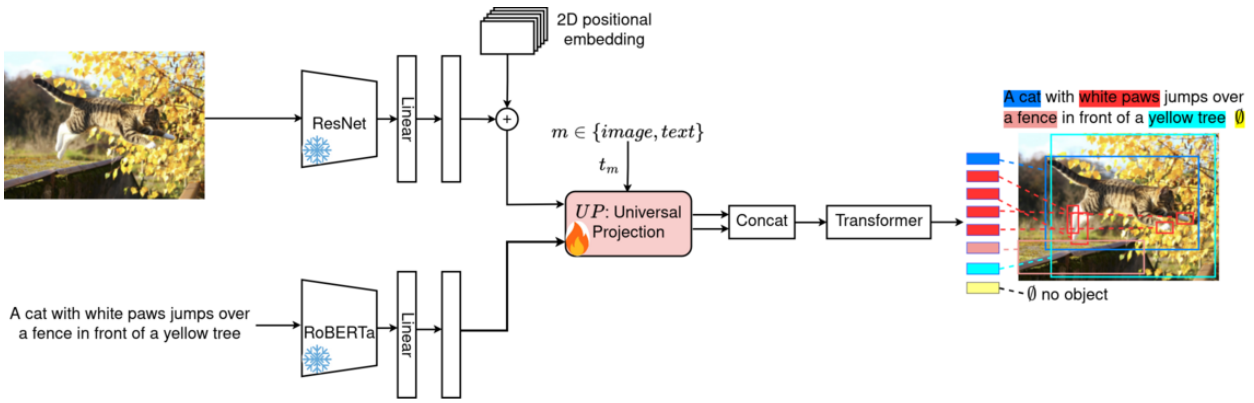


Figure 7.2: LightMDETR Architecture: Visual and textual features are extracted via frozen ResNet and RoBERTa, then projected into a shared embedding space. A lightweight Universal Projection (UP) module, the only trainable component, processes early fused modality features using a learnable "modality token" t_m. The UP outputs are concatenated and fed into a Transformer encoder-decoder (DETR) to predict object bounding boxes.

learnable "modality token" t_m, specific to each modality (image or text). This approach allows the UP to encode both types of features as follows:

$$\begin{aligned} O_{UP} &= UP(O \otimes t_{\text{image}}) \\ T_{UP} &= UP(T \otimes t_{\text{text}}) \end{aligned} \tag{7.8}$$

where $\otimes$ denotes the fusion operation (e.g., addition, multiplication, concatenation, or cross-attention). The outputs O_{UP} and T_{UP} are then concatenated, similar to MDETR, and used as input for the transformer encoder-decoder (DETR) to predict object bounding boxes.

In MDETR, image and text features are encoded separately and only concatenated before being passed into DETR. However, as shown in [70], early fusion of image and text features can make visual features language-aware, allowing predictions to be conditioned on the text prompt. Building on this idea, we introduce an enhanced version of LightMDETR, called LightMDETR-Plus, shown in Fig. 7.3. LightMDETR-Plus adds three key components: a cross-fusion layer with Multi-Head Attention (MHA) [116], and two projection layers that refine MHA outputs before they are processed by the UP.

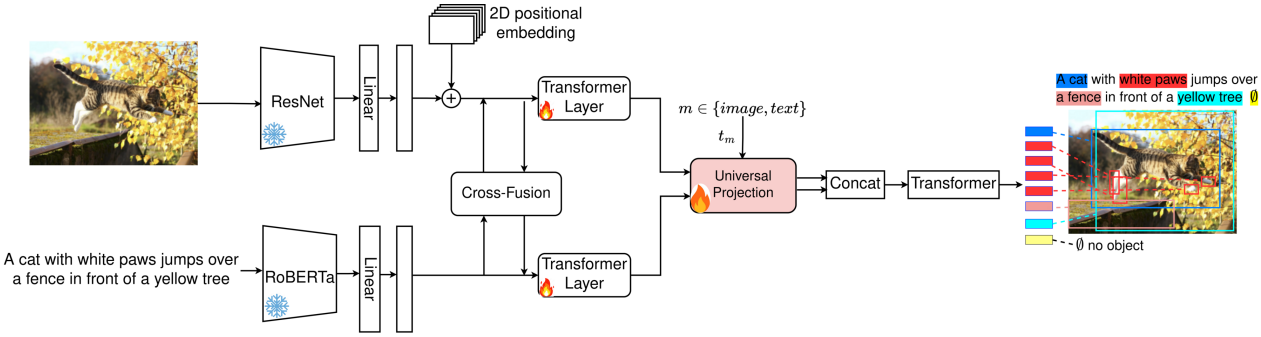


Figure 7.3: Architecture of LightMDETR-Plus: LightMDETR-Plus extends LightMDETR (ref. Figure 7.2) by introducing a cross-fusion layer prior to the UP thereby enhancing the model's representation capabilities.

The MHA takes as input the ResNet and RoBERTa encoder outputs, denoted as O and T, respectively. The transformations are expressed as:

$$\begin{aligned} O^{(q)} &= OW^{(q,O)}, \quad T^{(q)} = TW^{(q,T)}, \quad \text{Attn} = \frac{O^{(q)} \cdot (T^{(q)})^\top}{\sqrt{d}}, \\ T^{(v)} &= TW^{(v,T)}, \quad O_F = \text{SoftMax}(\text{Attn}) \cdot T^{(v)} \cdot W^{(\text{out},O)}, \\ O^{(v)} &= OW^{(v,O)}, \quad T_F = \text{SoftMax}(\text{Attn}^\top) \cdot O^{(v)} \cdot W^{(\text{out},T)}, \end{aligned} \tag{7.9}$$

where $\{W^{(\text{symbol},O)}, W^{(\text{symbol},T)} : \text{symbol} \in \{q, v, \text{out}\}\}$ are trainable parameters that play similar roles to those of query, value, and output linear layers in MHA [116], respectively, and d corresponds the output dimension.

After applying the cross-fusion mechanism with the Multi-Head Attention approach, a projection is performed using P_1 and P_2:

$$\begin{aligned} O_{P_1} &= P_1(O_\text{F} + O), \\ T_{P_2} &= P_2(T_\text{F} + T). \end{aligned} \tag{7.10}$$

The resulting O_{P_1} and T_{P_2} are then fed into the UP, following a similar process as in LightMDETR, as described by:

$$\begin{aligned} O_{UP} &= UP(O_{P_1} \otimes t_\text{image}), \\ T_{UP} &= UP(T_{P_2} \otimes t_\text{text}). \end{aligned} \tag{7.11}$$

The proposed lightweight framework for open-vocabulary object detection is modular. Similar to MDETR, we use an end-to-end approach and the same loss function 7.4 to train both LightMDETR and LightMDETR-Plus. To validate these methods, we compare their performance to MDETR on downstream tasks, including phrase grounding, referring expression comprehension, and segmentation (ref. 7.4).

7.4 Results

7.4.1 Pre-training

For the pre-training task, we adopt the MDETR approach, which leverages modulated detection to identify and detect all objects referenced in the corresponding free-form text.

For a fair comparison, we use the same combined training dataset as in [58], which integrates multiple image collections, including Flickr30k [129], MS COCO [71], and Visual Genome (VG) [63]. Flickr30k contains 31,783 images with detailed annotations for 158,915 region descriptions, primarily focused on objects and actions within the scenes. MS COCO contributes approximately 118,000 images, annotated with over 886,000 segmentations covering a wide range of common objects in diverse contexts. Visual Genome adds 108,077 images, with more than 5.4 million region descriptions and dense object annotations. For annotations, referring expressions datasets for fine-grained object references is leveraged, VG regions for detailed object-location relationships, Flickr entities for linking text descriptions with image regions, and the GQA train balanced set, which provides 1.7 million questions linked to object and scene graphs, enhancing the dataset's ability to support complex reasoning tasks. This combined dataset ensures robust and comprehensive training, covering a diverse range of objects, contexts, and linguistic references.

For both LightMDETR and LightMDETR-Plus, a frozen pre-trained RoBERTa-base [75] is used as the text encoder, which consists of 12 transformer layers, each with a 768-dimensional

hidden state and 12 attention heads, totaling 125M parameters. The visual backbone is a frozen pre-trained ResNet-101 [47], with 44M parameters. The only trainable component in both models is the UP module (see Fig. 7.2 and 7.3), composed of four transformer layers with four attention heads, contributing 4M trainable parameters. In LightMDETR-Plus, projection layers P_1 and P_2 add a single transformer layer each, with 787,968 parameters. The modality tokens t_{image} and t_{text} are initialized randomly. By freezing both pre-trained encoders, the number of trainable backbone parameters is reduced from 169M in the original MDETR to 4M in LightMDETR and 5M in LightMDETR-Plus (ref. Fig. 7.4).

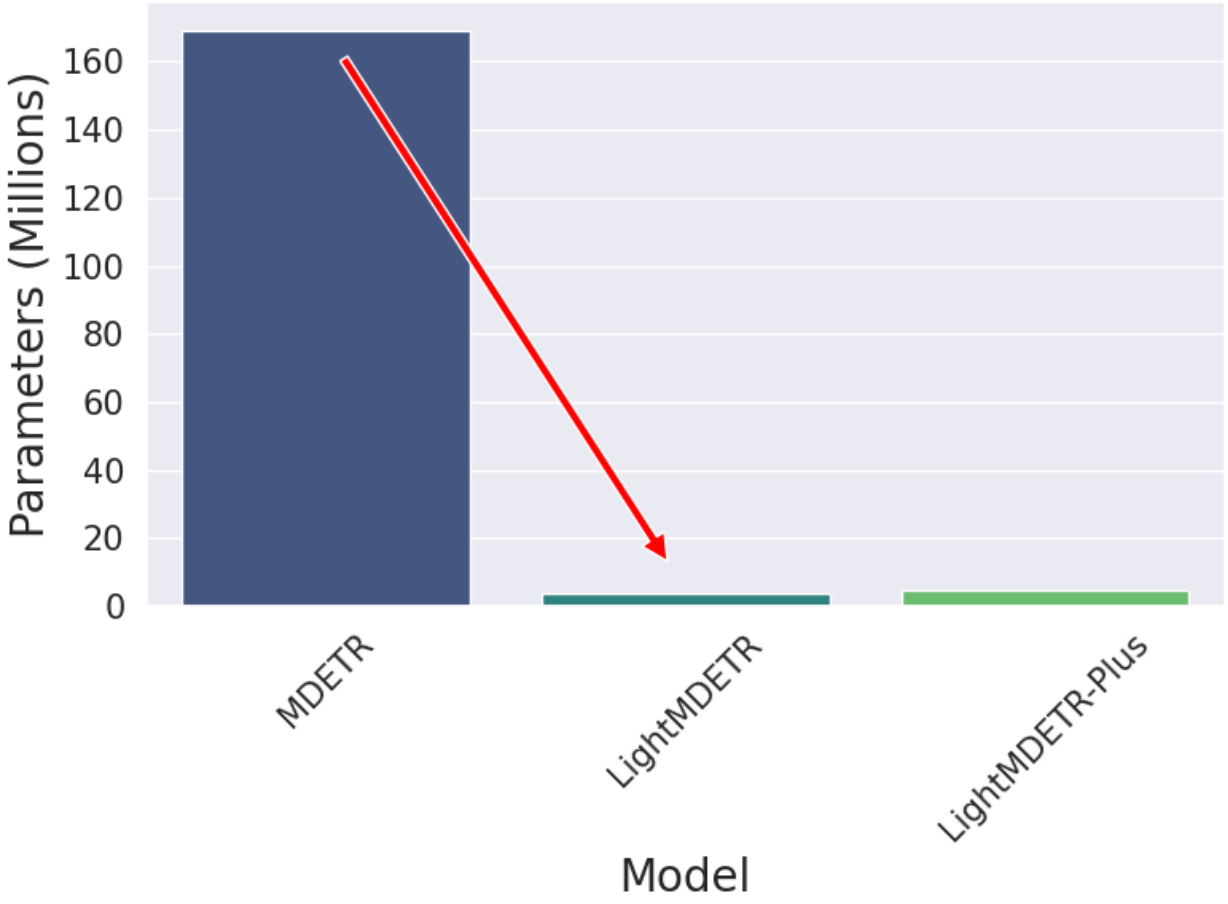


Figure 7.4: Comparison of trainable backbone parameters (in millions) during training between MDETR, LightMDETR, and LightMDETR-Plus.

For the fusion operation in the UP, as described in Equation 7.8, an addition method is

employed. All models are pre-trained for 40 epochs with an effective batch size of 64.

7.4.2 Dowstream Tasks

The proposed method is evaluated on three downstream tasks: phrase grounding, referring expression comprehension, and segmentation. To ensure a fair comparison, the same experimental setup as MDETR is adopted. Further details are available in the original paper.

Phrase grounding

is a task of identifying the fine-grained correspondence between phrases in a sentence and objects (or regions) in an image. We use the Flickr30k entities dataset for this task, and evaluate models performance in terms of Recall@k. For each sentence in the test set, 100 bounding boxes are predicted and use the soft token alignment prediction to rank the boxes according to the score given to the token positions.

Method	Val			Test		
	R@1	R@5	R@10	R@1	R@5	R@10
MDETR	82.5	92.9	**94.9**	83.4	93.5	**95.3**
LightMDETR	83.98	93.15	94.20	**83.87**	94.10	95.17
LightMDETR-Plus	**84.02**	**93.56**	94.9	83.80	**94.66**	95.23

Table 7.1: Comparison of phrase grounding performance on the Flickr30k dataset. Evaluation is reported using Recall at top 1, 5, and 10 predictions (R@1, R@5, R@10) on both validation and test splits.

As shown in Table 7.1, both LightMDETR and its extended version, LightMDETR-Plus, demonstrate competitive performance compared to MDETR. LightMDETR-Plus achieves the highest R@1 and R@5 on the validation set, with a slight improvement over LightMDETR and MDETR. On the test set, LightMDETR-Plus also outperforms the other models in R@5, demonstrating its effectiveness in grounding phrases more accurately. Overall, these results highlight that LightMDETR

and LightMDETR-Plus not only reduce the number of trainable parameters but also maintain or slightly improve performance on this task.

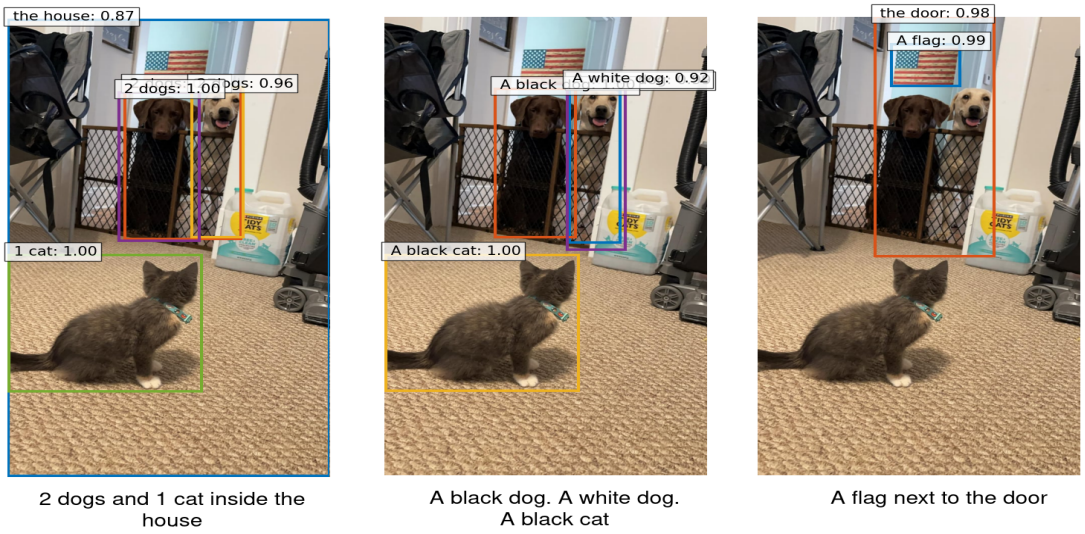


Figure 7.5: An illustration of LightMDETR on modulated detection. The model is designed to identify the root of a phrase as the positive token span, as demonstrated in these figures.

Referring expression comprehension

entails locating an object in an image using a textual description to predict a bounding box. We fine-tune both models on specific datasets—RefCOCO [59], RefCOCO+ [131], and RefCOCOg [85]—for five epochs, while keeping ResNet-101 and RoBERTa frozen. During inference, the models leverage the $\emptyset$ label to rank the 100 predicted bounding boxes, thereby improving the accuracy of object identification based on the provided expression. Table 7.2 presents a comparison of our models, LightMDETR and LightMDETR-Plus, against other detection models on RefCOCO, RefCOCO+, and RefCOCOg. RefCOCO and RefCOCO+ are evaluated using person vs. object splits: "testA" includes images with multiple people, while "testB" includes those with multiple objects. There is no overlap between training, validation, and testing images. RefCOCOg is split into two partitions.

Results presented in Table 7.3 showcase the precision performance of our models, LightMDETR and LightMDETR-Plus, in comparison to MDETR on the RefCOCO, RefCOCO+, and RefCOCOg datasets. Precision at rank k (P@k) indicates the percentage of correct predictions within the top

Method	RefCOCO			RefCOCO+			RefCOCOg	
	val	testA	testB	val	testA	testB	val	test
MAttNet [132]	76.65	81.14	69.99	65.33	71.62	56.02	66.58	67.27
ViLBERT [78]	-	-	-	72.34	78.52	62.61	-	-
VL-BERT [108]	-	-	-	72.59	78.57	62.30	-	-
UNITER [18]	81.41	87.04	74.17	75.90	81.45	66.70	74.86	75.77
VILLA [38]	82.39	87.48	74.84	76.17	81.54	66.84	76.18	76.71
ERNIE-ViL [130]	-	-	-	75.95	82.07	66.88	-	-
MDETR	86.75	**89.58**	81.41	79.52	84.09	70.62	81.64	**80.89**
LightMDETR	86.77	88.50	**82.00**	**79.56**	83.28	70.60	**82.02**	79.67
LightMDETR-Plus	**86.80**	88.76	81.78	79.10	**84.12**	**71.07**	81.06	80.81

Table 7.2: Accuracy performance comparison between our proposed models, LightMDETR and LightMDETR-Plus, and other detection models in the referring expression comprehension task on the RefCOCO, RefCOCO+, and RefCOCOg datasets. For testing, RefCOCO and RefCOCO+ datasets are evaluated using person vs. object splits: "testA" includes images with multiple people, while "testB" includes images with multiple objects from other categories. RefCOCOg features two distinct data partitions.

Method	RefCOCO			RefCOCO+			RefCOCOg		
	P@1	P@5	P@10	P@1	P@5	P@10	P@1	P@5	P@10
MDETR	85.90	95.41	96.67	**79.44**	**93.95**	**95.51**	80.88	94.19	95.97
LightMDETR	**85.92**	95.48	**96.76**	79.24	93.83	95.26	**80.97**	**94.87**	96.30
LightMDETR-Plus	85.37	**95.52**	96.73	77.98	93.85	95.47	80.24	94.26	**96.56**

Table 7.3: Precision performance comparison between our proposed models, LightMDETR and LightMDETR-Plus, and MDETR in the referring expression comprehension task on the RefCOCO, RefCOCO+, and RefCOCOg datasets.

k ranked results. Specifically, P@1 measures precision at the top-1 prediction, P@5 within the top 5, and P@10 within the top 10.

Proposed models demonstrate competitive performance, with LightMDETR achieving the highest precision at P@1 on RefCOCO (85.92%) and RefCOCOg (80.97%), surpassing MDETR slightly on these datasets. Furthermore, LightMDETR-Plus leads in P@5 on RefCOCO (95.52%) and P@10 on RefCOCOg (96.56%), highlighting the effectiveness of our lightweight approach. Although MDETR performs marginally better on RefCOCO+, LightMDETR closely follows, validating our hypothesis that freezing the backbone and training only the UP module allows our models to achieve comparable, if not superior, performance with reduced computational complexity.

Referring expression segmentation

Referring expression segmentation involves pinpointing and delineating objects in images using textual cues, as demonstrated with the PhraseCut dataset [122]. This dataset features images sourced from VG, complete with segmentation masks for a variety of expressions, many of which refer to multiple objects. Following the approach of MDETR, our training unfolds in two phases. Initially,

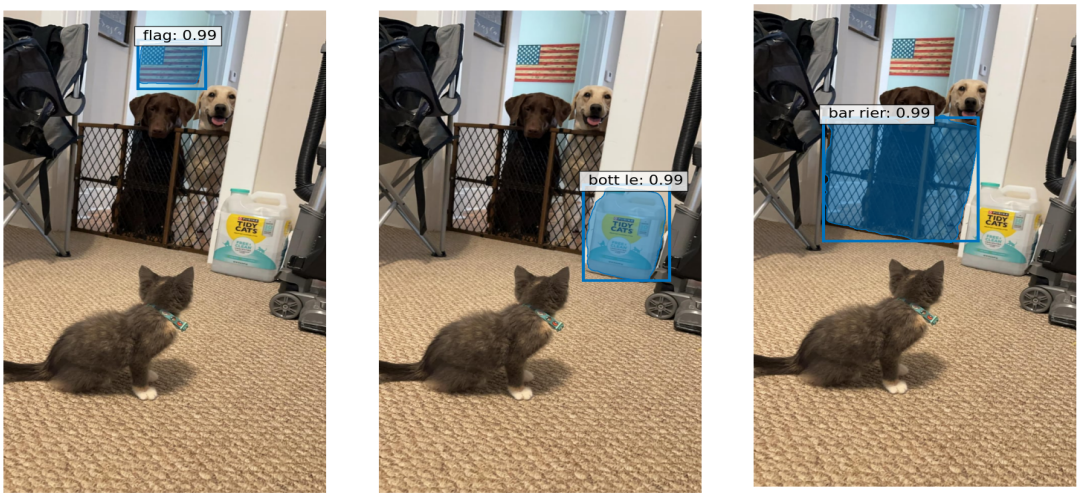


Figure 7.6: An illustration of LightMDETR on segmentation with the model fine-tuned on phrase-Cut.

we fine-tune our pre-trained model for 10 epochs while keeping ResNet-101 and RoBERTa frozen, optimizing for precise bounding box predictions and employing box AP for early stopping. In the subsequent phase, we freeze the network weights and focus on training a segmentation head for 35 epochs, implementing a learning rate adjustment at 25 epochs, supervised by a blend of Dice/F1 loss [34] and Focal loss [100]. During inference, we assign each predicted box a confidence score of $1-P(\emptyset)$, filtering out those below a threshold of 0.7. Ultimately, we consolidate the masks from the selected boxes into a unified binary mask corresponding to the referring expression. The results in

Method	M-IoU	Pr@0.5	Pr@0.7	Pr@0.9
RMI [13]	21.1	22.0	11.6	1.5
HULANet [132]	41.3	42.4	27.0	5.7
MDETR	53.1	56.1	38.9	**11.9**
LightMDETR	53.45	56.98	39.12	11.6
LightMDETR-Plus	**53.87**	**57.07**	**39.27**	11.82

Table 7.4: Validation of Referring Expression Segmentation using the mean intersection-over-union (IoU) between predicted and ground-truth masks, alongside precision Pr@I, where success is defined as the predicted mask achieving an IoU with the ground-truth that exceeds the threshold I.

Table 7.4 highlight the effectiveness of our proposed methods, LightMDETR and its enhanced variant LightMDETR-Plus. Both methods demonstrate superior performance compared to MDETR, achieving a mean intersection-over-union (M-IoU) of 53.45 and 53.87, respectively. Notably, they also exhibit improved precision at various thresholds, particularly at Pr@0.5 and Pr@0.7, with LightMDETR-Plus leading the metrics.

Downstream tasks such as phrase grounding, referring expression comprehension, and segmentation demonstrate that our proposed lightweight framework significantly enhances the efficiency of open-vocabulary object detection training. By considerably reducing the number of trainable parameters, it maintains or even improves performance on these tasks as illustrated in Fig. 7.5 and 7.6.

7.5 Discussion

A novel method for training open-vocabulary object detection systems is presented, significantly reducing the number of parameters to tune by leveraging prior knowledge. The approach employs specialized pre-trained encoders for text and images, which remain frozen during training. The only trainable component is a lightweight module, termed the "Universal Projection" (UP) module, designed to efficiently encode features from both text and image encoders using shared parameters.

A learnable parameter, referred to as the "modality token" (prior knowledge), is introduced to identify the source of each feature. This token is integrated into the UP representation, enabling seamless transitions between text and image feature processing. By relying on this lightweight design and the use of pre-trained encoders, the number of trainable parameters is minimized without compromising performance.

When applied to the MDETR model, this method achieves superior accuracy and precision across tasks such as phrase grounding, referring expression comprehension, and segmentation. Beyond MDETR, the approach is adaptable as a modular framework for other open-vocabulary object detection systems, reducing training costs while maintaining high performance.

This approach reinforces the principles outlined in Part I and Part II, demonstrating the critical role of leveraging prior knowledge in enhancing model representation. By utilizing pre-trained encoders and shared parameters, the method effectively capitalizes on existing knowledge, leading to improved performance across various tasks. This strategy not only strengthens the model's representational capacity but also addresses significant challenges in deep learning, such as the high cost of training and the extensive dataset requirements typically needed to develop robust models.

Reducing reliance on large-scale datasets and prolonged training cycles is particularly impactful, as it mitigates resource constraints while maintaining competitive accuracy and precision. By embedding prior knowledge into the architecture, the method aligns with modern trends in efficient deep learning, where performance improvements are achieved through intelligent design rather than brute-force data expansion. This highlights the importance of exploring similar approaches to further optimize the balance between computational efficiency and model effectiveness.

Conclusion

In this part, we focus on the advancements in open-vocabulary object detection (OVD) and a novel method that leverages prior knowledge to address key challenges in the field. OVD has emerged as a breakthrough in computer vision, allowing systems to detect objects not seen during training by associating visual features with textual descriptions, enabled by models like CLIP. Transformer-based architectures like MDETR and OWL-ViT enhance this capability by processing both visual and textual information together, improving detection accuracy. However, despite its potential, OVD faces challenges related to computational efficiency, resource consumption, and the complexity of aligning visual features with textual embeddings, particularly as the number of possible object categories expands.

The proposed method introduces a solution to these issues by leveraging pre-trained encoders for both text and images, reducing the need for extensive retraining. The core of the approach is the lightweight Universal Projection (UP) module, which efficiently encodes features using shared parameters, minimizing the number of trainable parameters. A modality token is also incorporated to identify the source of each feature, facilitating smooth transitions between text and image processing. This approach not only maintains high performance across various tasks, including phrase grounding and segmentation, but also reduces training costs and mitigates resource constraints.

By using pre-trained encoders and minimizing the number of parameters to tune, the method effectively capitalizes on existing knowledge, offering a scalable and efficient alternative to traditional OVD approaches. This aligns with the modern trend in deep learning to improve performance through intelligent design, reducing the reliance on large datasets and lengthy training cycles. Ultimately, the method enhances model representational capacity, improving accuracy and precision while addressing the computational and scalability challenges inherent in OVD.

Part IV

Conclusions and Future Directions

The challenges of training deep neural networks have drawn significant research interest, driven by the dual objectives of enhancing model representations for improved task adaptation and performance, and reducing the number of parameters to tune. Achieving these goals not only lowers training costs but also minimizes reliance on large datasets, broadening the models' applicability across diverse domains. This dissertation explores the integration of prior knowledge into training processes, examining its impact across a wide range of applications to validate the concept. The work centers on two primary problems: leveraging prior knowledge to normalize neural network activations during training for better representation, and incorporating prior knowledge in multimodal systems to reduce training costs and the dependency on large datasets while maintaining strong performance.

7.6 Enhancing Neural Network Representations with Prior Knowledge-Based Normalization.

In Chapter 3, we introduced two normalization techniques, Context Normalization (CN) and Context Normalization Extended (CN-X), which leverage predefined structural information, referred to as "contexts", to improve neural network representations. These methods incorporate prior knowledge to enhance the quality of normalization, resulting in better model performance. For scenarios where predefined contexts are unavailable or difficult to construct, we proposed an alternative: Adaptive Context Normalization (ACN), which dynamically constructs contexts and learns normalization parameters as part of the neural network's weights. While ACN offers flexibility, it is generally outperformed by CN and CN-X when meaningful predefined contexts are available, as demonstrated through various experiments.

The findings in this thesis open several avenues for future research that go beyond the immediate scope of image processing and have the potential to inform new lines of inquiry for subsequent researches.

1. Extending Normalization to Self-Supervised Learning Frameworks

The proposed normalization techniques can be adapted for self-supervised learning (SSL), where predefined or dynamically constructed contexts could serve as inductive biases to guide feature

learning. In SSL frameworks, such as contrastive or masked prediction models, integrating CN and CN-X may help encode more meaningful latent structures, particularly when data lacks explicit labels. Future research could investigate how context-based normalization affects the pretraining phase and its subsequent influence on downstream tasks.

2. Exploring Normalization in Federated Learning (FL)

In federated learning, models are trained across distributed, decentralized datasets while preserving privacy. Introducing CN and CN-X into FL settings could improve local model representations by incorporating domain-specific contexts at each client. Research could explore how context-aware normalization impacts convergence rates, generalization, and privacy guarantees in FL.

3. Integration with Large-Scale Graph Neural Networks (GNNs)

Graph Neural Networks (GNNs) often process data with complex structural dependencies. Extending CN and CN-X to graph settings could involve defining contexts as node clusters, communities, or hierarchical graph structures. This approach may enhance representation quality for applications like social network analysis, drug discovery, or recommendation systems. Similarly, ACN could adaptively construct contexts in dynamic graphs, opening up new directions for scalable graph processing.

4. Normalization in Dynamic and Nonstationary Environments

ACN's adaptability could be refined to handle dynamic and nonstationary environments where data distributions evolve over time. For instance, ACN could be applied to continual learning settings, where contexts adapt to new tasks or domains without forgetting previous knowledge. This research could investigate how context construction and parameter initialization strategies influence model stability and plasticity.

5. Context-Driven Model Compression

The integration of predefined contexts in CN and CN-X could inspire new techniques for model compression. By leveraging context-based representations, it may be possible to design models with fewer parameters but comparable or improved performance. This could extend to creating efficient architectures for edge devices, where computational and storage constraints are critical.

6. Optimization and Theoretical Analysis of Context Selection

One of the key open questions is how to construct and select contexts optimally. Future research

could formalize the relationship between context quality, representation power, and model performance. This could involve developing optimization algorithms for automated context selection or exploring theoretical guarantees for context-based normalization in neural networks.

7. Broader Applications Across Modalities

While the thesis focused on image processing, the principles behind CN, CN-X, and ACN could extend to domains such as robotics, reinforcement learning, and complex control systems. For example, contexts could represent task hierarchies or state abstractions in reinforcement learning environments, enabling more efficient policy learning.

8. Extending ACN with Meta-Learning Approaches

The performance of ACN is highly dependent on initialization. Future work could explore meta-learning techniques to enable ACN to learn optimal initialization strategies across tasks. By doing so, ACN could generalize more effectively to unseen domains, making it a robust alternative to CN and CN-X in real-world applications.

The perspectives outlined here demonstrate the broad applicability and potential impact of normalization techniques based on prior knowledge. By extending these ideas to self-supervised learning, federated learning, GNNs, and other emerging fields, future research can open new frontiers in neural network design and optimization, driving progress across a wide range of AI applications.

7.7 Leveraging Prior Knowledge to Reduce Training Costs in Multimodal Systems

In Chapters 5 and 7, we presented a novel approach for training multimodal systems at a lower cost by significantly reducing the number of parameters to tune compared to traditional methods. This was achieved through the integration of prior knowledge into model representations, enabling the reuse of shared parameters to encode multiple modalities. This design leverages trainable modality-specific parameters, referred to as "modality tokens," which allow the model to adapt effectively to

different input types without requiring a separate set of parameters for each modality.

By reducing the number of parameters to tune, this approach addresses a critical bottleneck: the reliance on large paired datasets, which are often scarce in certain domains, thereby limiting the applicability of multimodal systems. Beyond reducing computational costs, the incorporation of prior knowledge enhances model performance, outperforming traditional, resource-intensive methods.

For future work, this strategy could be extended to **large language models (LLMs)** and other foundational AI architectures, with the goal of eliminating the need for modality-specific models and moving towards a **universal encoder** capable of processing diverse data types. Integrating modality tokens into such systems would enable seamless handling of multimodal inputs—such as text, images, audio, and more—using a shared architecture, significantly reducing development and training overhead.

Key propositions for future exploration include:

1. Developing a universal encoder framework: Designing a single, adaptable encoder that can process different modalities by leveraging modality tokens and prior knowledge. This would eliminate the need to build and maintain separate models for each modality.

2. Extending modality tokens to unstructured and semi-structured data: Investigating how these tokens could represent diverse data types such as time-series, or tabular data in addition to text, images and videos.

3. Dynamic token learning: Creating mechanisms to dynamically learn and adjust modality tokens during training, allowing the universal encoder to generalize across unseen modalities or tasks.

4. Unified multimodal LLMs: Incorporating the universal encoder approach into LLM architectures to process and generate multimodal content, leveraging transfer learning across modalities.

5. Low-resource domain adaptation: Applying this methodology to domains with limited paired data, such as combining medical imaging with textual diagnostics or low-resource languages with audio.

6. Efficient fine-tuning: Exploring how the integration of prior knowledge and a universal encoder design can streamline fine-tuning for multimodal tasks, further reducing costs and training time.

7.8 Exploring Neural Network Design through Game Theory and Statistical Mechanics

While normalization techniques like CN, CN-X, and ACN have proven effective in enhancing neural network representations, they may not fully capture the complexity of feature interactions in high-dimensional spaces. To address this, we are exploring a novel neural network architecture, **NEUROGAME**, which integrates principles from **game theory** and **statistical mechanics** to create more efficient and accurate models.

In this framework, neurons are conceptualized as players in a cooperative game, where their activation values correspond to a set of actions. Neural layers are treated as sequential games, and the learning process is driven by a payoff function quantified through the **Shapley value**, linked to an energy function. During training, neurons are iteratively evaluated and filtered based on their contributions to the overall network objective. Only the most contributive neurons—those forming strong coalitions—propagate information to the next layer, reducing redundant computations and improving both efficiency and accuracy.

The **NEUROGAME** framework draws inspiration from statistical mechanics, where the flow of information between neurons is governed by probabilistic principles. The transmission of activation signals across layers follows a Gibbs distribution, introducing a natural form of regularization. This design enables the network to dynamically balance exploration and exploitation, stabilizing training and mitigating overfitting.

Potential Research Directions:

- **Combining Normalization with Game-Theoretic Learning:** Investigate how CN, CN-X and ACN can be combined with NEUROGAME, where normalization provides a stabilized training signal, and game-theoretic selection refines the network topology.
- **Dynamic Network Pruning and Regularization:** Use the NEUROGAME framework for dynamic pruning, where underperforming neurons are gradually excluded based on their

marginal contribution. The winning coalition forms a sparse yet powerful representation, and neurons outside the coalition are naturally dropped, providing adaptive regularization.

- **Generalization and Transfer Learning:** Explore whether the coalition-building process in NEUROGAME enhances generalization, particularly in low-data regimes or transfer learning settings, by promoting more structured, high-impact feature representations.
- **Scalability and Large-Scale Architectures:** Adapt the approach to scale with larger architectures (e.g., transformers or graph neural networks), leveraging game-theoretic dynamics to control complexity in deep, multimodal models.
- **Energy-Efficient Neural Networks:** Investigate how the statistical mechanics-inspired signal transmission can lead to energy-efficient models, where computational resources are focused only on the most impactful neurons, potentially enabling deployment on resource-constrained devices.

By bridging theoretical insights from game theory and statistical mechanics with practical advancements in neural network design, NEUROGAME represents a promising step forward. This hybrid approach opens new possibilities for building more adaptive, interpretable, and high-performance AI systems, capable of learning and evolving in dynamic, real-world environments.
The detailed methodology and initial results for the NEUROGAME framework can be found in our ongoing research paper: https://arxiv.org/abs/2410.12264.

References

[1] Jean-Baptiste Alayrac et al. "Flamingo: a visual language model for few-shot learning". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 23716–23736.

[2] Elad Amrani et al. "Noise estimation using density estimation for self-supervised multimodal learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 8. 2021, pp. 6644–6652.

[3] Rohan Anil et al. "PaLM 2 Technical Report". In: *arXiv preprint arXiv:2305.10403* (2023).

[4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. "Layer normalization". In: *arXiv preprint arXiv:1607.06450* (2016).

[5] Alexei Baevski et al. "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations." In: *NeurIPS*. Ed. by Hugo Larochelle et al. 2020.

[6] Max Bain et al. "Frozen in time: A joint video and image encoder for end-to-end retrieval". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 1728–1738.

[7] Hang Bao et al. "BEiT: BERT Pre-Training of Image Transformers". In: *International Conference on Learning Representations (ICLR)*. 2022.

[8] Yoshua Bengio, Nicolas Boulanger-Lewandowski, and Razvan Pascanu. "Advances in optimizing recurrent networks". In: *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE. 2013, pp. 8624–8628.

[9] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. "Is space-time attention all you need for video understanding?" In: *ICML*. Vol. 2. 3. 2021, p. 4.

[10] Sébastien Bubeck et al. "Sparks of Artificial General Intelligence: Early experiments with GPT-4". In: *arXiv preprint arXiv:2303.12712* (2023).

[11] Nicolas Carion et al. "End-to-End Object Detection with Transformers". In: *European Conference on Computer Vision (ECCV)* (2020), pp. 213–229.

[12] David Chen and William B Dolan. "Collecting highly parallel data for paraphrase evaluation". In: *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*. 2011, pp. 190–200.

[13] Ding-Jie Chen et al. "See-through-text grouping for referring image segmentation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 7454–7463.

[14] Sikai Chen, Yue Leng, and Samuel Labi. "A deep learning algorithm for simulating autonomous driving considering prior knowledge and temporal information". In: *Computer-Aided Civil and Infrastructure Engineering* 35.4 (2020), pp. 305–321.

[15] Ting Chen et al. "A simple framework for contrastive learning of visual representations". In: *International conference on machine learning*. PMLR. 2020, pp. 1597–1607.

[16] Xinlei Chen et al. "Improved baselines with momentum contrastive learning". In: *arXiv preprint arXiv:2003.04297* (2020).

[17] Xinlei Chen et al. "Microsoft coco captions: Data collection and evaluation server". In: *arXiv preprint arXiv:1504.00325* (2015).

[18] Yen-Cheng Chen et al. "UNITER: UNiversal Image-TExt Representation Learning". In: *European Conference on Computer Vision*. 2020.

[19] HaeChun Chung, JooYong Shim, and Jong-Kook Kim. "Cross-Modal Contrastive Representation Learning for Audio-to-Image Generation". In: *arXiv preprint arXiv:2207.12121* (2022).

[20] Jaejun Chung et al. "ALIGN: Activating Latent Innovations for Multimodal Contrastive Learning". In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021.

[21] Antonia Creswell et al. "Generative adversarial networks: An overview". In: *IEEE signal processing magazine* 35.1 (2018), pp. 53–65.

[22] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum likelihood from incomplete data via the EM algorithm". In: *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B* 39.1 (1977), pp. 1–38.

[23] Jia Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[24] Karan Desai and Justin Johnson. "Virtex: Learning visual representations from textual annotations". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 11162–11173.

[25] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)* (2019), pp. 4171–4186.

[26] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *International Conference on Learning Representations*. 2021.

[27] Yu Du et al. "Learning to prompt for open-vocabulary object detection with vision-language model". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 14084–14093.

[28] Zhengxiao Du et al. "GLM-130B: An Open Bilingual Pre-Trained Model". In: *arXiv preprint arXiv:2210.02414* (2022).

[29] Yiqun Duan et al. "Dewave: Discrete eeg waves encoding for brain dynamics to text translation". In: *arXiv preprint arXiv:2309.14030* (2023).

[30] Yiqun Duan et al. "Dewave: Discrete encoding of eeg waves for eeg to text translation". In: *Advances in Neural Information Processing Systems* 36 (2024).

[31] Maksim Dzabraev et al. "Mdmmt: Multidomain multimodal transformer for video retrieval". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 3354–3363.

[32] Mark Everingham et al. "The Pascal Visual Object Classes Challenge: A Retrospective". In: *International Journal of Computer Vision* 111.1 (Jan. 2015), pp. 98–136. DOI: 10.1007/s11263-014-0733-5.

[33] Abolfazl Farahani et al. "A brief review of domain adaptation". In: *Advances in Data Science and Information Engineering: Proceedings from ICDATA 2020 and IKE 2020* (2021), pp. 877–894.

[34] N Fausto Milletari and Ahmadi Seyed-Ahmad V-Net. *Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation.*

[35] Li Fei-Fei, Rob Fergus, and Pietro Perona. "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories". In: *2004 conference on computer vision and pattern recognition workshop*. IEEE. 2004, pp. 178–178.

[36] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. 2017, pp. 1126–1135.

[37] Valentin Gabeur et al. "Multimodal transformer for video retrieval". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*. Springer. 2020, pp. 214–229.

[38] Zhe Gan et al. "Large-scale adversarial training for vision-and-language representation learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6616–6628.

[39] Rohit Girdhar et al. "Imagebind: One embedding space to bind them all". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 15180–15190.

[40] Ross Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[41] Yuan Gong, Yu-An Chung, and James Glass. "AST: Audio Spectrogram Transformer". In: *Proc. Interspeech 2021*. 2021, pp. 571–575. DOI: 10.21437/Interspeech.2021-698.

[42] Ian Goodfellow et al. "Generative adversarial networks". In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[43] Xiuye Gu et al. "Open-vocabulary object detection via vision and language knowledge distillation". In: *arXiv preprint arXiv:2104.13921* (2021).

[44] Andrey Guzhov et al. "Audioclip: Extending clip to image, text and audio". In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, pp. 976–980.

[45] Andrey Guzhov et al. "Esresnet: Environmental sound classification based on visual domain models". In: *2020 25th international conference on pattern recognition (ICPR)*. IEEE. 2021, pp. 4933–4940.

[46] Jiaming Han et al. "Onellm: One framework to align all modalities with language". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 26584–26595.

[47] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[48] Yuze He et al. "T^3 Bench: Benchmarking Current Progress in Text-to-3D Generation". In: *arXiv preprint arXiv:2310.02977* (2023).

[49] Martin Heusel et al. "GANs trained by a two time-scale update rule converge to a local Nash equilibrium". In: *Advances in Neural Information Processing Systems*. 2017.

[50] Gao Huang et al. "Densely connected convolutional networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4700–4708.

[51] Lei Huang et al. "Normalization techniques in training dnns: Methodology, analysis and application". In: *IEEE transactions on pattern analysis and machine intelligence* 45.8 (2023), pp. 10173–10196.

[52] Nail Ibrahimli, Julian FP Kooij, and Liangliang Nan. "MuVieCAST: Multi-View Consistent Artistic Style Transfer". In: *2024 International Conference on 3D Vision (3DV)*. IEEE. 2024, pp. 1136–1145.

[53] Sergey Ioffe. "Batch renormalization: Towards reducing minibatch dependence in batch-normalized models". In: *Advances in Neural Information Processing Systems*. 2017, pp. 1945–1953.

[54] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. 2015, pp. 448–456.

[55] Chao Jia et al. "Scaling up visual and vision-language representation learning with noisy text supervision". In: *International conference on machine learning*. PMLR. 2021, pp. 4904–4916.

[56] Haojun Jiang et al. "Cross-modal adapter for text-video retrieval". In: *arXiv preprint arXiv:2211.09623* (2022).

[57] Mahdi M Kalayeh and Mubarak Shah. "Training faster by separating modes of variation in batch-normalized models". In: *IEEE transactions on pattern analysis and machine intelligence* 42.6 (2019), pp. 1483–1500.

[58] Aishwarya Kamath et al. "Mdetr-modulated detection for end-to-end multi-modal understanding". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 1780–1790.

[59] Alireza Kazemzadeh et al. "ReferItGame: Referring to Objects in Photographs of Real-World Scenes". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 786–795.

[60] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[61] Mustafa Taha Koçyigit et al. "Unsupervised batch normalization". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, pp. 918–919.

[62] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. "Robust consistent video depth estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 1611–1621.

[63] Ranjay Krishna et al. "Visual genome: Connecting language and vision using crowdsourced dense image annotations". In: *International journal of computer vision* 123.1 (2017), pp. 32–73.

[64] Alex Krizhevsky and Vinod Nair. "Cifar-100 (canadian institute for advanced research). 30 [65] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25.1097-1105 (2012), p. 26.

[65] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. "Cifar-10 (canadian institute for advanced research)". In: *URL http://www. cs. toronto. edu/kriz/cifar. html* 5.4 (2010), p. 1.

[66] Zhenzhong Lan et al. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations". In: *International Conference on Learning Representations (ICLR)*. 2020.

[67] Ya Le and Xuan Yang. "Tiny imagenet visual recognition challenge". In: *CS 231N* 7.7 (2015), p. 3.

[68] Yann LeCun and Corinna Cortes. "MNIST handwritten digit database". In: (2010). URL: http://yann.lecun.com/exdb/mnist/.

[69] Yann LeCun et al. "Efficient backprop". In: *Neural networks: Tricks of the trade*. Springer, 2002, pp. 9–50.

[70] Liunian Harold Li et al. "Grounded language-image pre-training". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10965–10975.

[71] Tsung-Yi Lin et al. "Microsoft COCO: Common objects in context". In: *Lecture Notes in Computer Science* 8693 (2014), pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48.

[72] Wei Liu et al. "Ssd: Single shot multibox detector". In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[73] Xizhou Liu et al. "Evolving normalization-activation layers". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 15858–15870.

[74] Yang Liu et al. "Use What You Have: Video retrieval using representations from collaborative experts." In: *BMVC*. 2019, p. 279.

[75] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[76] Ze Liu et al. "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 9992–10002.

[77] Ilya Loshchilov and Frank Hutter. "Decoupled Weight Decay Regularization". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=Bkg6RiCqY7.

[78] Jiasen Lu et al. "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks". In: *Advances in neural information processing systems* 32 (2019).

[79] Huaishao Luo et al. "Clip4clip: An empirical study of clip for end to end video clip retrieval and captioning". In: *Neurocomputing* 508 (2022), pp. 293–304.

[80] Ping Luo et al. "Mode Normalization". In: *International Conference on Machine Learning*. 2019, pp. 4203–4212.

[81] Ping Luo et al. "Switchable normalization for learning-to-normalize deep representation". In: *IEEE transactions on pattern analysis and machine intelligence* 43.2 (2019), pp. 712–728.

[82] Yiwei Ma et al. "X-clip: End-to-end multi-grained contrastive learning for video-text retrieval". In: *Proceedings of the 30th ACM International Conference on Multimedia*. 2022, pp. 638–647.

[83] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. "Rectifier nonlinearities improve neural network acoustic models". In: *Proc. icml*. Vol. 30. 1. Atlanta, GA. 2013, p. 3.

[84] Mateusz Malinowski and Mario Fritz. "A multi-world approach to question answering about real-world scenes based on uncertain input". In: *Advances in neural information processing systems* 27 (2014).

[85] J. Mao et al. "Generation and Comprehension of Unambiguous Referring Expressions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 11–20.

[86] Matthias Minderer et al. "Simple open-vocabulary object detection". In: *European conference on computer vision*. Springer. 2022, pp. 728–755.

[87] Weizhi Nie et al. "T2TD: Text-3D generation model based on prior knowledge guidance". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).

[88] Aaron van den Oord and et al. "Representation learning with contrastive predictive coding". In: *arXiv preprint arXiv:1807.03748* (2018).

[89] OpenAI. "GPT-4 Technical Report". In: *arXiv preprint arXiv:2303.08774* (2023).

[90] Vassil Panayotov et al. "Librispeech: an asr corpus based on public domain audio books". In: *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2015, pp. 5206–5210.

[91] Omkar M Parkhi et al. "Cats and dogs". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2012, pp. 3498–3505.

[92] K. J. Piczak. "Environmental sound classification with convolutional neural networks". In: *Proceedings of the IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)* (2015), pp. 1–6.

[93] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[94] Alec Radford et al. "Learning transferable visual models from natural language supervision". In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.

[95] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[96] Mengye Ren et al. "Normalizing the normalizers: Comparing and extending network normalization schemes". In: *arXiv preprint arXiv:1611.04520* (2016).

[97] Shaoqing Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (2015), pp. 1137–1149.

[98] Nicolae-Cătălin Ristea, Andrei Anghel, and Radu Tudor Ionescu. "Cascaded cross-modal transformer for audio–textual classification". In: *Artificial Intelligence Review* 57.9 (2024), p. 225.

[99] Becca Roelofs et al. "Adamatch: A unified approach to semi-supervised learning and domain adaptation". In: *Proc. of the Int. Conf. on Learning Representations (ICLR)*. 2022.

[100] T-YLPG Ross and GKHP Dollár. "Focal loss for dense object detection". In: *proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2980–2988.

[101] J. Salamon, C. Jacoby, and J. P. Bello. "A dataset and taxonomy for urban sound research". In: *Proceedings of the 22nd ACM international conference on Multimedia*. 2014, pp. 1041–1044.

[102] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv preprint arXiv:1910.01108* (2019).

[103] Elvis Saravia et al. "CARER: Contextualized affect representations for emotion recognition". In: *Proceedings of the 2018 conference on empirical methods in natural language processing*. 2018, pp. 3687–3697.

[104] Pierre Sermanet, Soumith Chintala, and Yann LeCun. "Convolutional neural networks applied to house numbers digit classification". In: *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*. IEEE. 2012, pp. 3288–3291.

[105] Cheng Shi and Sibei Yang. "Edadet: Open-vocabulary object detection using early dense alignment". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 15724–15734.

[106] Oleksii Sidorov et al. "Textcaps: a dataset for image captioning with reading comprehension". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer. 2020, pp. 742–758.

[107] R. Socher et al. "Recursive deep models for semantic compositionality over a sentiment treebank". In: *Proceedings of the 2013 conference on empirical methods in natural language processing*. Citeseer. 2013, pp. 1631–1642.

[108] Weiyue Su et al. "VL-BERT: Pre-training of Generic Visual-Linguistic Representations". In: *International Conference on Learning Representations*. 2020.

[109] Mingxing Tan and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks". In: *International conference on machine learning*. PMLR. 2019, pp. 6105–6114.

[110] Zhan Tong et al. "Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training". In: *Advances in neural information processing systems* 35 (2022), pp. 10078–10093.

[111] Hugo Touvron et al. "Llama: Open and Efficient Foundation Language Models". In: *arXiv preprint arXiv:2302.13971* (2023).

[112] Hugo Touvron et al. "Training data-efficient image transformers and distillation through attention". In: *International Conference on Machine Learning (ICML)*. 2021.

[113] G. Tzanetakis and P. Cook. *Musical genre classification of audio signals*. 2002.

[114] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. "Instance normalization: The missing ingredient for fast stylization". In: *arXiv preprint arXiv:1607.08022* (2016).

[115] Generalized Intersection Over Union. "A metric and a loss for bounding box regression". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 658–666.

[116] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[117] Sergey Verbitskiy, Vladimir Berikov, and Viacheslav Vyshegorodtsev. "Eranns: Efficient residual audio neural networks for audio pattern recognition". In: *Pattern Recognition Letters* 161 (2022), pp. 38–44.

[118] E. M. Voorhees and D. K. Harman. "The TREC-8 question answering track report". In: *TREC*. 1999.

[119] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. "You Only Learn One Representation: Unified Network for Multiple Tasks". In: *Journal of Information Science and Engineering* (2023).

[120] Zhepei Wang et al. "Unsupervised improvement of audio-text cross-modal representations". In: *2023 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE. 2023, pp. 1–5.

[121] Xi Wei et al. "Multi-Modality Cross Attention Network for Image and Sentence Matching". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[122] Chenyun Wu et al. "Phrasecut: Language-based image segmentation in the wild". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 10216–10225.

[123] Shengqiong Wu et al. "NExT-GPT: Any-to-Any Multimodal LLM". In: *Proceedings of the International Conference on Machine Learning*. 2024, pp. 53366–53397.

[124] Yuxin Wu and Kaiming He. "Group normalization". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 3–19.

[125] Zhibiao Wu and Martha Palmer. "Verbs semantics and lexical selection". In: *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics. 1994, pp. 133–138.

[126] Jun Xu et al. "Msr-vtt: A large video description dataset for bridging video and language". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 5288–5296.

[127] Xueyuan Yang, Chao Yao, and Xiaojuan Ban. "Spatial-Related Sensors Matters: 3D Human Motion Reconstruction Assisted with Textual Semantics". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 9. 2024, pp. 10225–10233.

[128] Zhilin Yang et al. "XLNet: Generalized Autoregressive Pretraining for Language Understanding". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf.

[129] Peter Young et al. "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions". In: *Transactions of the Association for Computational Linguistics* 2 (2014), pp. 67–78.

[130] Fei Yu et al. "Ernie-vil: Knowledge enhanced vision-language representations through scene graphs". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35. 4. 2021, pp. 3208–3216.

[131] L. Yu et al. "Modeling Context with Deep Neural Networks for Referring Expression Understanding". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 1195–1204.

[132] Licheng Yu et al. "Mattnet: Modular attention network for referring expression comprehension". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1307–1315.

[133] Lu Yuan et al. "Florence: A new foundation model for computer vision". In: *arXiv preprint arXiv:2111.11432* (2021).

[134] Sergey Zagoruyko and Nikos Komodakis. "Wide residual networks". In: *arXiv preprint arXiv:1605.07146* (2016).

[135] Alireza Zareian et al. "Open-vocabulary object detection using captions". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 14393–14402.

[136] Biao Zhang and Rico Sennrich. "Root mean square layer normalization". In: *Advances in Neural Information Processing Systems* 32 (2019).

[137] Gaowei Zhang, Yue Pan, and Limao Zhang. "Deep learning for detecting building façade elements from images considering prior knowledge". In: *Automation in Construction* 133 (2022), p. 104016.

[138] Yiyuan Zhang et al. "Meta-transformer: A unified framework for multimodal learning". In: *arXiv preprint arXiv:2307.10802* (2023).

[139] Yuhao Zhang et al. "Contrastive learning of medical visual representations from paired images and text". In: *Machine Learning for Healthcare Conference*. PMLR. 2022, pp. 2–25.

[140] Yuxin Zhang et al. "Inversion-based style transfer with diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 10146–10156.

[141] Yiwu Zhong et al. "Regionclip: Region-based language-image pretraining". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16793–16803.

[142] Xizhou Zhu et al. "Deformable DETR: Deformable transformers for end-to-end object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 833–842.

Summary

Deep learning has revolutionized various fields, yet several challenges remain, particularly in terms of training efficiency, model adaptability, and scalability in multimodal and open-vocabulary settings. This thesis addresses these challenges by incorporating prior knowledge into deep learning architectures, focusing on three primary areas: normalization techniques, multimodal representation learning, and open-vocabulary object detection. Through these innovations, we aim to enhance model performance, reduce computational costs, and improve generalization in resource-constrained environments.

The contributions of this work are organized into three key areas:

1. Normalization with Prior Knowledge

This thesis introduces novel normalization techniques that integrate prior knowledge to enhance training efficiency and representation quality. These techniques are designed to overcome the limitations of existing methods, which often assume simplistic data distributions that may not hold in complex, real-world scenarios:

- **Context Normalization (CN)** and **Context Normalization Extended (CN-X)**: These methods incorporate predefined domain-specific contexts to improve task performance and model stability.
- **Adaptive Context Normalization (ACN)**: This approach dynamically constructs context during training, allowing for better adaptability to changing data distributions or situations where predefined contexts are difficult to define.

Applications: Image classification, domain adaptation, and image generation.

2. Multimodal Representation Learning

In the realm of multimodal systems, the challenge lies in efficiently aligning different modalities such as text, image, audio, and video. This work presents **OneEncoder**, a lightweight framework designed to progressively align multimodal representations using minimal resources:

- **OneEncoder Framework**: A framework that uses simple addition for progressive modality alignment, reducing computational overhead.
- **OneEncoder-$\oplus$**: A variant of OneEncoder employing addition as the fusion technique for modality alignment.
- **OneEncoder-$\odot$**: A refined version using cross-attention as the fusion technique for more complex alignment.

Applications: Zero-shot classification, querying, and visual question answering across diverse modalities.

3. Open-Vocabulary Object Detection

In open-vocabulary object detection (OVOD), models are expected to generalize to new, unseen categories using textual descriptions rather than relying on pre-trained labels. This thesis presents the following advancements in OVOD:

- **LightMDETR**: A modular framework built on MDETR that reduces training complexity while enhancing object detection performance by leveraging prior knowledge.
- **LightMDETR-Plus**: An improved version of LightMDETR that incorporates attention mechanisms to enhance its adaptability to novel categories.

Applications: Phrase grounding, referring expression comprehension, and referring expression segmentation.

By addressing these challenges, this thesis demonstrates how prior knowledge can improve deep learning models' scalability, efficiency, and adaptability in multimodal and open-vocabulary settings. The proposed methods are validated across a range of tasks, showcasing their potential to improve performance in real-world applications, especially when computational resources are limited.

Acknowledgments

I would like to express my deepest gratitude to all the individuals who have contributed to the successful completion of this thesis.

Academic Support

- **Professor Hanane Azzag** — Thesis supervisor. Thank you for your invaluable guidance, expertise, and unwavering support throughout my research.
- **Professor Mustapha Lebbah** — DAVID Laboratory, UVSQ. Your innovative ideas and contributions were crucial to this research.
- **Associate Professor Fanchen Feng** — Co-supervisor. Your expert advice and thoughtful feedback greatly improved the quality of this work.
- **Professor Djamel Bouchaffra** — University Paris-Saclay, UVSQ. Your inspiring feedback helped me refine my approach and explore new perspectives.
- **A3 team at the LIPN computer science laboratory, Sorbonne Paris Nord University** — Thank you for your collaborative spirit and support.

Family and Personal Support

- **Insa FAYE** — My father. Thank you for your endless love and encouragement.
- **Boudouel DIOP** — My mother. Your unwavering belief in me has been my strength.
- **Aminata THIOUNE** — My wife. Thank you for your patience, love, and constant support.
- **Ndeye Marie FAYE** and **Lena FAYE** — My sisters. Your love and support mean the world to me.

In Loving Memory

- **Mr. Omar Aidara** — Your passion for technology and AI inspired me to pursue my studies with determination. May you rest in peace.
- **Mr. Jean Joseph Lapolice** — My first inspiring teacher. Your impact on me will never be forgotten. May you rest in peace.
- **Gérard Gardarin** — Like a father to me. Your warmth and support were invaluable. May you rest in peace.

Friends and Community

- **Yola Gardarin** — Thank you for your love and kindness throughout my journey.
- **Mambodj DIOP** and **Mbathio Sow** — Thank you for welcoming me in France and making me feel at home.
- My friends and relatives in Senegal and France — Your encouragement and support have been a pillar of strength.

To all those who have contributed to my research and personal growth, directly or indirectly, I offer my deepest thanks.

Research Implementations

During this thesis, several research implementations were developed to support and validate the proposed methodologies. The following GitHub repositories contain the source code of these implementations, enabling reproducibility and further exploration of the research contributions:

- **Context Normalization:** *Implementation of a normalization layer based on prior knowledge to enhance neural network representations.*
  Source code available: https://github.com/b-faye/prior-knowledge-norm.
- **OneEncoder:** *Lightweight framework for progressive alignment of modalities in deep learning models.*
  Source code available: https://github.com/b-faye/OneEncoder.
- **LightMDETR:** *Modular framework for low-cost training of open-vocabulary object detection.*
  Source code available: https://github.com/b-faye/lightmdetr.
- **NEUROGAME:** *Game theory and statistical mechanics-driven neural network design for efficient learning and adaptive regularization.*
  Source code available: https://github.com/b-faye/neurogame.

Author publications

• International Conferences

- **Bilal Faye**, Mustapha Lebbah, Hanane Azzag. *Supervised Batch Normalization.* International Congress on Information and Communication Technology (ICICT), 2025. Link
- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah, Djamel Bouchaffra. *Adaptative Context Normalization: A Boost for Deep Learning in Image Processing.* IEEE International Conference on Image Processing (ICIP), 2024. Link
- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah, Djamel Bouchaffra. *Lightweight Cross-Modal Representation Learning.* European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), 2024. Link
- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah, Fangchen Feng. *UAN: Unsupervised Adaptive Normalization.* World Congress on Computational Intelligence (WCCI), 2024. Link
- Nicolas Ballier, Dahn Cho, **Bilal Faye**, et al. *The SPECTRANS System Description for the WMT21 Terminology Task.* EMNLP 2021 Sixth Conference on Machine Translation (WMT21), 2021, pp. 815–820. Link

• Journals

- Djamel Bouchaffra, Fayçal Ykhlef, **Bilal Faye**, Hanane Azzag, Mustapha Lebbah. *Game Theory Meets Statistical Mechanics in Deep Learning Design.* Neural Networks, 2025. Link
- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah, Djamel Bouchaffra. *OneEncoder: A Lightweight Framework for Progressive Alignment of Modalities.* Neural Computing and Applications Journal, 2025. Link

- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah, Fangchen Feng. *Context Normalization: A New Approach for the Stability and Improvement of Neural Network Performance.* Data & Knowledge Engineering (DKE), 2024. Link

• National Conferences

- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah, Fangchen Feng. *Normalisation Contextuelle : Une Nouvelle Approche pour la Stabilité et l'Amélioration des Performances des Réseaux de Neurones.* Revue des Nouvelles Technologies de l'Information (RNTI-E-40), 2024. Link

• Workshops

- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah, Mohamed-Djallel Dilmi, Djamel Bouchaffra. *Context Normalization Layer with Applications.* IEEE International Conference on Data Mining Workshops (ICDMW), 2023. Link

• Preprinted Articles

- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah. *Value-Free Policy Optimization via Reward Partitioning.* Arxiv, 2025. Link

- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah. *A Lightweight Modular Framework for Low-Cost Open-Vocabulary Object Detection Training.* Arxiv, 2025. Link

- **Bilal Faye**, Hanane Azzag, Mustapha Lebbah, Djamel Bouchaffra. *Enhancing Neural Network Representations with Prior Knowledge-Based Normalization.* Arxiv, 2025. Link