



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

Ph.D. DISSERTATION

A Deep Representation Learning for
Unsupervised Anomaly Detection

비지도 이상 탐지를 위한 표현 학습론

BY

HYUNSOO CHO

FEBRUARY 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

A Deep Representation Learning for
Unsupervised Anomaly Detection

비지도 이상 탐지를 위한 표현 학습론

BY

HYUNSOO CHO

FEBRUARY 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

A Deep Representation Learning for Unsupervised Anomaly Detection

비지도 이상 탐지를 위한 표현 학습론

지도교수 이 상 구

이 논문을 공학박사 학위논문으로 제출함

2023년 2월

서울대학교 대학원

컴퓨터공학부

조 현 수

조현수의 공학박사 학위논문을 인준함

2023년 2월

위 원 장	강 유
부위원장	이 상 구
위 원	조 성 준
위 원	황 승 원
위 원	김 태 욱

Abstract

Anomaly detection is the task of identifying data instances that deviate from regular observations, commonly referred to as anomalies or outliers. Software systems in the wild encounter anomalies ceaselessly, which may lead to catastrophic failure in safety-critical applications, such as medical diagnosis or self-driving cars. Thus, the ability to discriminate anomalies is a cornerstone for securing the reliability of the software and maintaining a high user experience.

In this dissertation, we present a comprehensive analysis and novel methods for deep learning-based anomaly detection in an unsupervised manner, primarily concentrating on extracting a more distinctive representation from deep neural models. The first part of this thesis proposes self-supervised learning frameworks for two most common input types (i.e., images and sentences): Masked Contrastive Learning (MCL) and Layer-agnostic Contrastive Learning (LaCL). MCL (Chapter 3) generates a mask that adjusts the repelling ratio in contrastive loss to form label-wisely dense representations and boosts the overall performance via image rotation inference. The latter, LaCL (Chapter 4), encourages intermediate features to learn layer-specialized representations and assembles them into a single representation to absorb rich information in the pre-trained language model.

The remaining half of this dissertation delves into investigating the methods of utilizing large-scale pre-trained models for anomaly detection, reflecting the current surging interest in large models due to their versatility. In Chapter 5, we first analyze the capability of large pre-trained language models (PLMs) as outlier detectors in various perspectives, including their behavior in conjunction

with recent parameter-efficient transfer learning methods, and share several intriguing findings and limitations. On the basis of previous findings, Chapter 6 introduces a novel transfer learning method for large pre-trained models dubbed prompt augmented linear probing (PALP), where its underlying mechanism is inspired by the recent prompting methods, which manipulate large PLMs by prepending extra prefixes. PALP exhibits robustness to anomalies and high generalizability in both data scarcity and data-abundant scenarios without any access or adaptation of the model parameters.

The improvements proposed in this dissertation would advance the robustness of various real-world applications.

Keywords: machine learning, deep learning, anomaly detection, out-of-distribution detection, distributional shift, representation learning, natural language understanding, classification, large-scale pre-trained language models.

Student Number: 2019-37513

Contents

Abstract	i
Chapter 1 Introduction	1
1.1 Anomaly Detection	1
1.2 Deep Anomaly Detection	2
1.3 Dissertation Outline	5
1.4 Related Publications	6
Chapter 2 Background	8
2.1 Problem Formulation	8
2.2 Taxonomy of Deep Anomaly Detection	10
2.2.1 Data Type of The Input	11
2.2.2 Availability of Trainable Outliers	12
2.2.3 Techniques for Deep Anomaly Detection	13
2.3 General Paradigm	13
2.4 Scope of the Dissertation	15
2.5 Challenges and Relevant Literature	15
2.5.1 Challenges	15
2.5.2 Related Research Fields	16

Part I: Deep Self-Supervised Anomaly Detection Methods (Chapter 3 & 4)	18
Preliminaries	19
Contrastive Learning	19
Supervised Contrastive Learning	21
Chapter 3 A Deep Self-Supervised Anomaly Detection for Images	22
3.1 Introduction	22
3.2 Masked Contrastive Learning	26
3.2.1 Class-Conditional Mask	26
3.2.2 Stochastic Positive Attraction	27
3.2.3 Training Auxiliary task in MCL	31
3.3 Inference	32
3.3.1 Scoring Function in MCL	32
3.3.2 Self Ensemble Inference	33
3.4 Experiment	34
3.4.1 Multi-Class Anomaly Detection	37
3.4.2 Ablation Study	39
3.4.3 Qualitative results	43
3.5 Related Work	45
Chapter 4 A Deep Self-Supervised Anomaly Detection for Sentences	48
4.1 Introduction	48
4.2 Related Work	51
4.3 Layer-agnostic Contrastive Learning	53
4.3.1 Intuition	53

4.3.2	Global Compression Layer	55
4.3.3	Correlation Regularization Loss	55
4.3.4	Classification & OOD Scoring	56
4.3.5	Augmentation for Contrastive Learning	57
4.4	Experiments	59
4.4.1	Implementation Details	59
4.4.2	Dataset and Metrics	59
4.4.3	Competing Methods	62
4.4.4	Main Results	63
4.5	Analysis	67
4.5.1	Layer-wise Performance	67
4.5.2	Ablation study	67
4.5.3	Distribution Visualization	69
4.5.4	Case study	71
	Part I (Chapter 3 & 4) Summary	72

Part II: Exploiting Large Pre-trained Models for Anomaly Detection (Chapter 5 & 6) 74

Preliminaries	75
Parameter-Efficient Transfer Learning	75
In-Context Learning	76

Chapter 5 Investigating the Potential of Large Pre-trained Models as Anomaly Detectors 78

5.1	Introduction	78
5.2	Probing OOD Robustness	80
5.2.1	Backbones and Models	80
5.2.2	Dataset and Metrics	81

5.2.3	OOD Evaluation Methods	82
5.3	Analysis	84
5.3.1	OOD Robustness of PLMs without Supervision.	84
5.3.2	Evaluation methods for auto-regressive PLMs.	85
5.3.3	PETLs VS. Fine-tuning	86
5.4	Related Work	88

Chapter 6 Enriching the Anomaly Detection Ability of Large Pre-trained Models via Prompting 91

6.1	Introduction	91
6.2	Preliminary	94
6.2.1	Problem Formulation	94
6.2.2	Linear Classifiers	95
6.3	Prompt-Augmented Linear Probing	97
6.3.1	Motivation	97
6.3.2	Template-based Training Samples (PALP-T)	98
6.3.3	Demonstration-based Training Samples (PALP-D)	99
6.4	Experiments	101
6.4.1	Experimental Setup.	101
6.4.2	Few-shot Results.	104
6.4.3	Full-data Results.	106
6.4.4	Out-of-Distribution Detection Results.	108
6.5	Analysis & Ablations	109
6.5.1	Application to Small PLM.	109
6.5.2	Dataset Visualization.	110
6.6	Related Work	111
6.6.1	Black-box Tuning.	112

6.6.2 In-Context Learning.	113
Part II (Chapter 5 & 6) Summary	114
Chapter 7 Conclusion	116
Bibliography	119
초록	141

List of Figures

Figure 1.1	Performance Comparison of Deep learning-based algorithms Vs Traditional Algorithms (Chalapathy and Chawla, 2019)	3
Figure 2.1	Real-world inputs cover not only in-distribution space but also out-of-distribution space.	9
Figure 2.2	Taxonomy of deep learning-based anomaly detection. . .	10
Figure 2.3	General paradigm in deep anomaly detection. Methods in anomaly detection (1) extracts information from deep neural module and passes it to (2) the scoring function, which yields the fitness of the input and gives final decision whether the input is anomaly.	14
Figure 2.4	Illustration of adversarial attack. Predictive label for robin image changes into waffle iron with small perturbations.	17

Figure 2.5	Recognition algorithms generalize poorly to new environments. Cows in ‘common’ contexts (e.g. Alpine pastures) are detected and classified correctly (A), while cows in uncommon contexts (beach, waves and boat) are not detected (B) or classified poorly (C). Images are from (Beery et al., 2018)	19
Figure 2.6	Vanilla self-supervised contrastive learning and supervised contrastive learning. The supervised contrastive loss (right) contrasts the set of all samples from the same class as positives against the negatives from the remainder of the batch, while self-supervised contrastive loss contrasts a single positive for each anchor (i.e., an augmented version of the same image). The figure illustrates that taking class label information into account allows the aligning of representations from the same class more tightly. The image is from Khosla et al. (2020).	20
Figure 3.1	t -SNE visualization of CIFAR-10 trained representation. Each color denotes each class labels in CIFAR-10. Each component in our model makes each cluster more dense.	25
Figure 3.2	Description of MCL framework. Query view (gray-framed) attracts views connected with a blue-colored line and repels views connected with a red-colored line.	26

Figure 3.3	Samples from resized and fixed versions of ImageNet and LSUN for OOD detection. Because the resizing was done by subsampling, it caused an aliasing effect, resulting in unintended artifacts that made images almost indistinguishable even for humans. Fixed version used interpolation technique for the resizing.	36
Figure 3.4	4-way SEI and 8-way SEI.	40
Figure 3.5	More results on CIFAR-10 with CIFAR-100 as OOD. GT stands for the ground truth label. Each row shows a query image on the second column, followed by the top 3 most similar images of train data. Quite an amount of samples are either wrong, ambiguous, or vague. (a) Query images are from OOD, but the model took as an IND sample and classified with high confidence. (b) Wrongly classified samples from in-domain classification.	44
Figure 3.6	Case studies on OOD samples with high confidence and wrongly classified IND samples.	45
Figure 3.7	Histogram and PDF for correctly classified IND, wrongly classified IND and OOD distribution. MCL measures predictive uncertainty quite precisely.	45

Figure 4.1	Layer-wise and explicit ensemble (Shen et al., 2021) performance of baseline BERT. Explicit ensemble sometimes lead to worse AUROC (higher the better) than using a well-performing single layer representation, depending on the setting. Detailed explanations about setting and baseline model are elaborated in Sec.4.4.2 and Sec.4.4.3 individually.	50
Figure 4.2	Overall structure of Layer-agnostic Contrastive Learning (LaCL). The global compression layer trains the SCL loss in a <i>layer-agnostic</i> manner by engaging entire layers in the CL task. And the correlation regularization (CR) loss decorrelates each intermediate layer to avoid overlapping information between each layer.	54
Figure 4.3	Layer-wise AUROC score of baseline and our model with cosine scoring function. Explicit ensemble (baseline) tends to work well in relatively easy setting (far-OOD), while it yields worse performance than best performing single representation in harsh conditions (close-OOD). Implicit ensemble representation from LaCL outperforms other layers consistently.	66
Figure 4.4	Histogram of LaCL and Baseline model trained on Banking split 50% setting.	69
Figure 4.5	Illustration of various parameter efficient transfer learning methods. The image is from He et al. (2022)	75
Figure 4.6	Illustration of ICL.	77

Figure 5.1	OOD detection performance of PLMs without updating the model parameters.	84
Figure 5.2	OOD detection performance of PLMs without updating the model parameters.	86
Figure 6.1	Average accuracy (12 classification tasks) of various <i>black-box</i> transferring methods trained on GPT-J. ICL can not leverage the full train dataset due to the length limit, and their performance saturates quickly. Our method (PALP) is scalable with available training samples and minimizes the performance gap between ICL in a few-shot setting. The performance of the respective task is summarized in Table 6.3, 6.4.	93
Figure 6.2	Illustration of selecting demonstrations in PALP-D in binary classification task. We estimate the normal distribution for each class from available training samples and select the closest sample respectively.	100
Figure 6.3	<i>t</i> -SNE visualization of SST2 representation from GPT-J. Adding understandable prompts to the input can reshape the representation into a more task-specialized form. Demon-best is obtained by attaching the demonstration that showed the best performance, and Demon-worst is the opposite.	112

List of Tables

Table 2.1	Various real-world data and their corresponding input type and representative neural architectures.	11
Table 3.1	Test accuracy of in-domain classification and AUROC of OOD data for each model trained on CIFAR-10 dataset. .	38
Table 3.2	Ablation studies for each component in MCL.	39
Table 3.3	Ablation studies for SEI. MCL is trained with additional 4-way rotations auxiliary task.	41
Table 3.4	Ablation study for MCL with additional OOD metrics. . .	42
Table 3.5	Performance comparison between MCL and SupCLR. Models are trained on CIFAR-10 dataset with ResNet-18. . . .	42
Table 4.1	Dataset statistics.	59
Table 4.2	Overlapping classes between CLINC and Snips dataset. .	60
Table 4.3	IND / OOD performance of each model 3 different settings on close-OOD setting. The best performance in each method is indicated in bold and the global best is <u>underlined</u> . 63	63

Table 4.4	IND / OOD performance of each model 3 different settings on far-OOD setting. The best performance in each method is indicated in bold and the global best is <u>underlined</u> .	64
Table 4.5	IND / OOD performance of each model 3 different settings on Far-OOD setting. The best performance in each method is indicated in bold and the global best is <u>underlined</u> . LaCL outperforms other methods constantly in both IND and OOD metric.	65
Table 4.6	Ablation study on LaCL components in Banking split setting	68
Table 4.7	Data augmentaion results.	69
Table 4.8	Examples of OOD test samples misclassified as IND. The keywords that cause over-reliance are in bold . GT stands for the ground-truth label.	70
Table 4.9	Examples of IND test samples misclassified as OOD. Words related to their error type are highlighted in bold	70
Table 5.1	AUROC of each PLMs trained with LoRA adapter. En-ergergy function outperforms other evaluation methods consistently.	85
Table 5.2	AUROC of various PETL methods with various number of parameters. Performance was achieved by the energy function.	87
Table 6.1	Statistics of 15 different datasets used in our experiments.	101
Table 6.2	A list of template and verbalizer for each dataset.	103

Table 6.3	Experimental results on GPT-J in 4-shot per class settings. B, T, and D refers to a baseline, template, and demonstration individually. For each dataset, the best method is in bold and the <u>second best method</u> and is underlined.	104
Table 6.4	Experimental results on GPT-J in 8-shot per class settings.	105
Table 6.5	Experimental results of 11 different datasets on GPT-J in full datasets settings. B, T, and D refers to a baseline, PALP-Template, and PALP-Demonstration respectively. For each dataset, the best method is in bold and the <u>second best method</u> and is underlined.	106
Table 6.6	ICL cannot be applied to tasks with a large number of classes while various linear probing methods are capable of these tasks. For each dataset, the best method is in bold.	107
Table 6.7	OOD performance of PALP and other transfer learning methods.	108
Table 6.8	Results on GPT-2 (Large) in 4-shot per class setting. B, T, and D refers to a baseline, template, and demonstration individually. Our method is transferable to smaller model.	110
Table 6.9	Best and worst performing example prefixes from SST2. .	112

Chapter 1

Introduction

1.1 Anomaly Detection

The reliability of the software system depends heavily on the test phase. While most errors in the system are tractable before deployment through internal tests (e.g., debugging, alpha or beta test), some are not. The vast portion of real-world applications is deployed with these potential issues in hand and encounter ceaseless inputs, which often incorporate myriad abnormal instances that deviate from the initial scope or objective of the system. Such instances are commonly and interchangeably referred to as *anomalies* or *outliers* in the contemporary machine learning context, and they arise from many different reasons. While most anomalies occur by chance or mistakes, some outliers are maliciously designed to penetrate the vulnerability of the system to trigger system malfunction. Anomaly detection aims to discriminate outliers based on previous standard observations, enabling models to circumvent latent threats or catastrophic failures. As such, it is a vital research area for securing the high

reliability of the software and maintaining a pleasant user experience.

As these abilities are some of the paramount aspects of well-designed software systems, anomaly detection has been explored in a plethora of applications for decades. Since the seminal works (Hawkins, 1980; Knox and Ng, 1998), comprehensive surveys on general anomaly detection (Aggarwal, 2017; Boukerche et al., 2020; Chalapathy and Chawla, 2019; Chandola et al., 2009) and also within specific target domains have been presented, including but not limited to fraud detection (Adewumi and Akinyelu, 2017), medical diagnosis (Litjens et al., 2017), military surveillance (Mohammadi et al., 2018), autonomous driving (Kiran et al., 2018), cyber-intrusions (Kwon et al., 2019), or intention classification.

Early studies on anomaly detection usually employ a broad spectrum of traditional machine learning techniques, such as nearest neighbors, clustering, principal component analysis (PCA), and support vector machines (SVM). Specifically, considering the input data type, objective, or additional restrictions induced by the application domain, earlier works employed distinct strategies more suited for the desired task. On the other hand, most recent works employ deep neural networks (DNNs) exclusively due to their dominance in performance (Fig. 1.1) regardless of the nature of tasks. Following the recent trend, this dissertation also explores DNN-based anomaly detection methods.

1.2 Deep Anomaly Detection

Deep learning-based models have recently achieved groundbreaking success in computer vision (CV) and natural language processing (NLP), reaching parity with human-level performances in sundry benchmarks (Lin et al., 2014; Wang et al., 2019b; Deng et al., 2009a; Wang et al., 2019a). The leading drivers behind

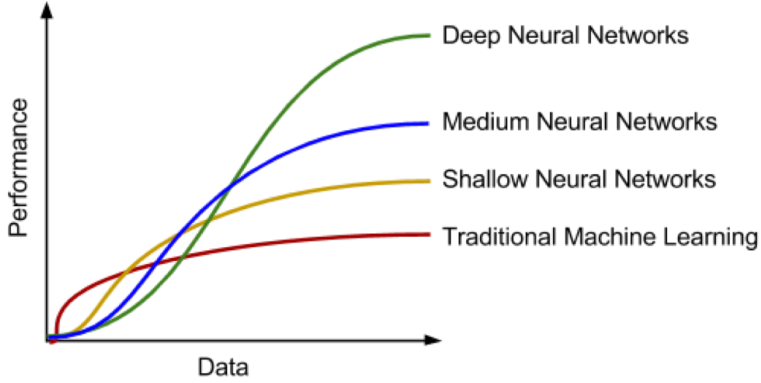


Figure 1.1: Performance Comparison of Deep learning-based algorithms Vs Traditional Algorithms (Chalapathy and Chawla, 2019)

its success are the (1) *deeper* models, which can capturing of highly complex relationships, and the (2) massive datasets, allowing the training of the *deep* model. While DNNs make a precise decision on the basis of the peripheral features, they contain no scientific meaning other than a statistical correlation between input and the label. Due to the aforementioned critical drawback, DNNs are notorious for their *black box* mechanism, implying that humans are incapable of understanding individual neurons or the final decision-making process. In addition to their lack of transparency or interpretability, DNN-based models are reported to be vulnerable to outliers or adversarial attacks and yield high-confidence predictions far from the training samples (Hein et al., 2019; Goodfellow et al., 2015). As these concerns grow, discerning suspicious inputs has also been regarded as integral research in DNNs, as it can compensate for DNNs’ poor reliability.

A variety of deep anomaly detection methods have been proposed, and their common underlying mechanism is to extract meaningful information from a

deep neural model and convert it into a single scalar value to measure the *normality* of the input. Thus, the core of deep anomaly detection lies in extracting and training sharp data representations from the neural models. Deep supervised anomaly detection achieves this goal by leveraging a trainable outlier dataset to maximize the discrepancy of the representation between outliers and normal samples. While supervised methods (Hendrycks et al., 2019b; Papadopoulos et al., 2019) exhibit a satisfactory performance, they require access to trainable outlier datasets, which is an extreme assumption considering the scope of outliers covers nigh infinite space. Thus, most recent works tackle the problem in an unsupervised setting, which detects outliers solely based on intrinsic properties of the normal data instances.

In this regard, our dissertation concentrates on devising a framework that can train more distinctive representations and analyzing how to extract better representations within DNNs in an unsupervised fashion. Precisely, we have sought room for improvement in two orthogonal branches: **Part I** leverages self-supervised learning to train and extract more distinctive information from deep neural networks, and **Part II** exploits large pre-trained models (billions of parameters) as anomaly detectors. The first part of our thesis proposes two distinct methods of learning high-quality representations for the two most common input types for human interaction (i.e., images and sentences) leveraging self-supervised learning. The second part of this dissertation attempts to exploit the large pre-trained language models which inheres many intriguing extra functionalities, such as the ability to capture world knowledge (Petroni et al., 2019), generate codes (Poesia et al., 2022), or solve mathematical problems (Henighan et al., 2020), in addition to being proficient in recognizing linguistic patterns.

1.3 Dissertation Outline

The remaining contents of this dissertation comprise four main parts:

- **Background (Chapter 2):** As a preliminary, we overview the preliminaries that form the foundation of this thesis. Specifically, we explain the background of anomaly detection and its taxonomy depending on various ingredients. Lastly, we point related chapter with the aforementioned taxonomy of the task and briefly introduce research areas closely associated with anomaly detection.
- **Part I (Chapters 3 & 4) — Deep Self-Supervised Learning for Anomaly Detection:** The first part of this thesis proposes methods of learning high-quality representations for the two most common input types for human interaction (i.e., images and sentences). Specifically, in Chapter 3, we introduce a method called **Masked Contrastive Learning (MCL)**, a task-specific variant of contrastive learning (Chen et al., 2020a). Among self-supervised learning tactics, *contrastive learning* is one specific framework validating their superiority in learning *task-agnostic* features without labels. MCL learns a more distinctive representation suited for anomaly detection by exploiting the label information in the training batch in addition to various image data augmentations. In Chapter 4, we propose a novel framework dubbed **Layer-agnostic Contrastive Learning (LaCL)** that encourages intermediate features of the language model to learn layer-specialized representations. Then, LaCL assembles them *implicitly* into a single representation to absorb rich and diverse information in the pre-trained language model.
- **Part II (Chapters 5 & 6) — Exploiting Large Pre-trained Mod-**

els for Anomaly Detection: The second part of this dissertation attempts to exploit the large pre-trained models in various circumstances. In Chapter 5, we conduct in-depth investigations on the capability of large pre-trained language models (PLMs) as outlier detectors in various perspectives to elucidate the following uncharted questions: *how do large-scale PLMs cope with outlier?* Specifically, our experiments elucidate the correlation between the model’s size and transferring methods and share several intriguing findings and limitations. Based on previous findings, Chapter 6 introduces a transfer learning method for large pre-trained models dubbed **P**rompt **A**ugmented **L**inear **P**robing (**PALP**), which displays robustness to outliers and works in more harsh conditions (i.e., few-shot setting, without any update or access to the model parameters).

- **Conclusion (Chapter 7):** We finally conclude this thesis and discuss prospective avenues for future work.

1.4 Related Publications

Portions of this dissertation appeared in the following publications or pre-print:

- **Chapter 3:** Hyunsoo Cho, Jinseok Seol, and Sang-goo Lee. 2021. Masked contrastive learning for anomaly detection. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI*.
- **Chapter 4:** Hyunsoo Cho, Choonghyun Park, Jaewook Kang, Kang Min Yoo, Tae-uk Kim, and Sang-goo Lee. 2022. Enhancing out-of-distribution detection in natural language understanding via implicit layer ensemble. In *Findings of the Association for Computational Linguistics: EMNLP*.

- **Chapter 5:** Hyunsoo Cho, Choonghyun Park, Junyeob Kim, Hyuhng Joon Kim, Kang Min Yoo, and Sang-goo Lee. 2023b. Probing out-of-distribution robustness of language models with parameter-efficient transfer learning. *arXiv preprint*.
- **Chapter 6:** Hyunsoo Cho, Hyuhng Joon Kim, Junyeob Kim, Sang-woo Lee, Sang-goo Lee, Kang Min Yoo, and Taeuk Kim. 2023a. Prompt-augmented linear probing: Scaling beyond the limit of few-shot in-context learners. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI*.

Chapter 2

Background

This chapter provides several concepts and preliminaries to help understand this dissertation. We first present the taxonomy of deep anomaly detection (DAD) from various perspectives and describe the general paradigm. Finally, we discuss challenges in our topics and briefly introduce several relevant literatures with their commonalities and differences.

2.1 Problem Formulation

The main objective of DAD is to determine whether the inferring data point \mathbf{x}_{test} is normal. In order to do so, DAD method approximates the distribution of normal space $P_{\text{real}}(\mathbf{x})$ from the available training dataset $\mathcal{D}_{\text{train}}$, hypothesizing that the data distribution of the training dataset $P_{\text{data}}(\mathbf{x})$ is identical to both target normal space $P_{\text{real}}(\mathbf{x})$ and the real-world test distribution $P_{\text{test}}(\mathbf{x})$:

$$P_{\text{real}}(\mathbf{x}) \approx P_{\text{data}}(\mathbf{x}) \approx P_{\text{test}}(\mathbf{x}) \quad (2.1)$$

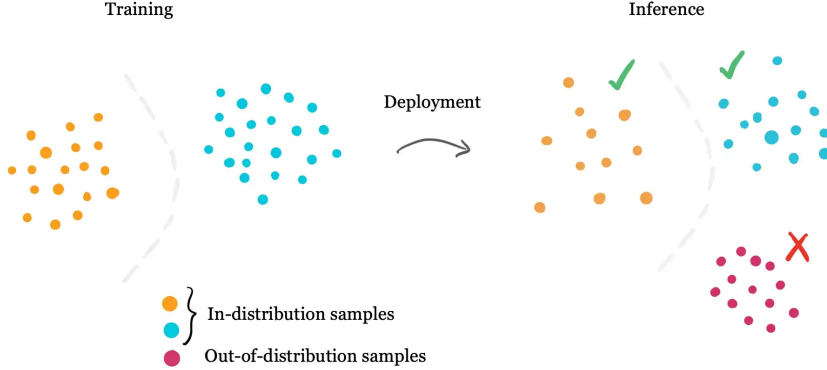


Figure 2.1: Real-world inputs cover not only in-distribution space but also out-of-distribution space.

This assumption is generally termed I.I.D (independent and identically distributed) assumption and is central to almost all machine learning algorithms. While the trained deep-learning based model exhibits decent performance when the test input comes from similar distribution to the training dataset, most real-world inputs incorporate novel input that deviates from the expected normal behavior commonly referred to as distributional shift. The cause of the distributional shift is diverse (e.g., by mistake, malicious input, or volatility of the input), leading to the degradation of the model’s reliability and generalization ability.

Given the circumstances, the primary objective of DAD model is to determine the normality of the test input \mathbf{x}_{test} solely based on the distribution estimated from the training dataset $P_{\text{data}}(\mathbf{x})$ without any clue regarding the outlier space:

$$\text{Model}(\mathbf{x}_{\text{test}}; P(\mathbf{x})) = \begin{cases} \text{inlier} & \text{if } \mathbf{x}_{\text{test}} \sim P_{\text{data}}(\mathbf{x}) \\ \text{outlier} & \text{otherwise,} \end{cases} \quad (2.2)$$

Thus DAD model heavily relies on the information in the representation from

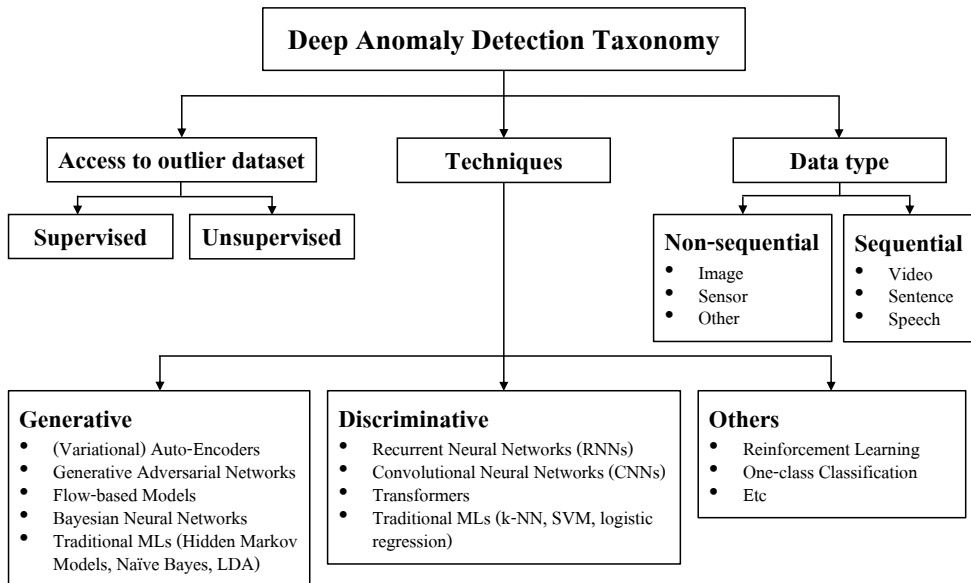


Figure 2.2: Taxonomy of deep learning-based anomaly detection.

the deep neural model and outputs binary decision of the test input or the normality score of the input.

2.2 Taxonomy of Deep Anomaly Detection

Methods for anomaly detection are determined by numerous factors: input data type, the availability of trainable outliers, specific techniques, or the additional requirements induced by the applying domains. Fig. 2.2 illustrates the overall taxonomy in DAD separated by several high-level factors. In this section, We take a closer look at the several key aspects that determine the taxonomy of DAD one by one to help understand the overall overview of our literature.

Data type	Model architectures	Example	Feature types
Sequential	CNN, RNN, Transformer	Video	Continuous
		Sentence	Discrete
		Speech	Continuous
		Stock Market	Categorical
		Network	Categorical, Continuous, Discrete
Non-sequential	CNN, RNN, Transformer, Generative models	Image	Continuous
		Sensor	Categorical, Continuous, Discrete
		Other	Categorical, Continuous, Discrete

Table 2.1: Various real-world data and their corresponding input type and representative neural architectures.

2.2.1 Data Type of The Input

All systems accept inputs ¹. Input, any signal or data instance with information the system receives for processing, comprises a set of *attributes*, interchangeably termed *variables*, *features*, or *dimensions*. Furthermore, the types of individual features can be further divided into binary, categorical, or continuous, and the input as well can be classified into univariate (single attribute) or multivariate (multiple attributes).

While there are tremendous types of data in the real world, they can be roughly separated into two high-level branches: sequential and non-sequential

¹also referred to as an object, record, point, vector, pattern, event, case, sample, observation, or entity (Chandola et al., 2009)

data. Tab. 2.1 summarizes some examples of real-world data with their characteristics. Sequential data generally utilize the recurrent neural network (Rumelhart et al., 1986) (RNN) or its variants (Cho et al., 2014; Hochreiter and Schmidhuber, 1997), as they exhibit temporal dynamic behavior by allowing output from middle nodes to affect subsequent input. The image data, a representative example of non-sequential type data, usually leverages a convolutional neural network (Fukushima and Miyake, 1982) as a backbone network, an example of brain-inspired ideas that mimic the human visual system. More recently, the transformer (Vaswani et al., 2017) has become the de-facto standard structure in universal applications as they exhibit decent performance regardless of their data type. The transformer dispenses recurrence or convolution and exclusively relies on *attention*, which enhances some parts of the input while diminishing others, motivated by the idea that the network should devote more to the small, but necessary regions of the data.

2.2.2 Availability of Trainable Outliers

According to the presence of trainable outlier samples, methodologies in anomaly detection can be branched into supervised or unsupervised. While supervised methods exhibit a more decent performance, they require access to the labeled outlier dataset. Collecting a labeled outlier dataset requires substantial cost and effort for many reasons: They are often manually labeled by a human expert, and the scope of outliers covers nigh infinite space; gathering the data in the whole OOD space is not only infeasible but also more challenging than collecting samples for whole normal behavior. Furthermore, anomalous behavior has evolving nature, e.g., the rise of new anomalies. On the other hand, unsupervised anomaly detection techniques do not require an outlier dataset and assume that the training data has labeled instances for only the normal

class. They usually make the implicit assumption of the normal instances solely based on intrinsic properties of the data instances and thus are most widely applicable.

2.2.3 Techniques for Deep Anomaly Detection

DAD methodologies leverage various approaches that can be broadly branched into discriminative and generative methods. Informally, *generative* models can generate new data instances (e.g., photos and videos), and *discriminative* models discriminate between different kinds of data instances (e.g., classifications).

Generative methods are commonly utilized in computer vision, while natural language processing applications utilize the discriminative method due to their discrete input space. In generative methods-based DAD methods, they utilize the error between reconstructed input and the original input, expecting them to generate precise reconstruction to the normal instances while fail to re-generate abnormal ones. Bayesian DAD methods, which inherit the flexibility of deep learning and the capability of probabilistic models to estimate predictive uncertainty, utilize the uncertainty of the input to sort out outliers. Discriminative DAD methods utilize the probability of the model prediction expecting that the model will derive a more confident probability of a normal input.

2.3 General Paradigm

While it is impossible to generalize them into a single paradigm, the most prevailing paradigm in deep unsupervised anomaly detection is to *extract* and *score*. (Fig. 2.3 provides a visual example of the paradigm mentioned above.)

Since many deep anomaly detectors leverage diverse backbone architectures, they extract different types of information from the various neural architectures that might be beneficial in determining outliers. For instance, most generative

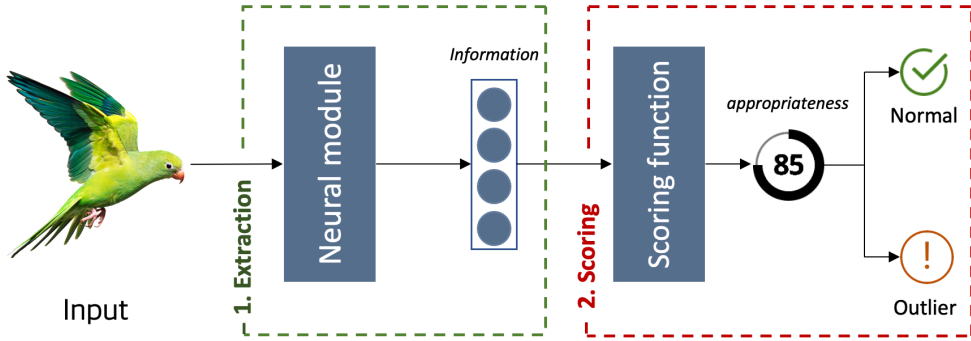


Figure 2.3: General paradigm in deep anomaly detection. Methods in anomaly detection (1) extracts information from deep neural module and passes it to (2) the scoring function, which yields the fitness of the input and gives final decision whether the input is anomaly.

models extract reconstruction error between the original input and its reconstruction, Bayesian methods utilize the uncertainty value, and discriminative methods utilize maximum soft probability or hidden representation from the architecture as their primary source to determine outliers. Then, the scoring function gauges the appropriateness of the input based on the extracted feature and decides whether the input is from the normal distribution or them. The most intuitive baseline scoring function is to utilize probabilities from softmax distributions (Hendrycks and Gimpel, 2017). Correctly classified examples tend to have greater maximum softmax probabilities (MSP) than anomalous inputs allowing the model to discern out-of-distribution examples. On top of this baseline method, numerous branches of scoring functions have been proposed, such as designing a more accurate scoring function (Liu et al., 2020; Tack et al., 2020; Shen et al., 2021) and calibrating the input via gradient-based post-processing (Lee et al., 2018b; Liang et al., 2017).

2.4 Scope of the Dissertation

Among various settings and methods in anomaly detection described above, we primarily focus on (1) training or extracting more meaningful representations in (2) an unsupervised fashion with (3) image and sentence datasets.

(1) Although designing a novel scoring function is also promising research, designing a ubiquitous scoring function is infeasible as the nature of the representation varies depending on the training methods or other factors. Meaning the scoring function should change adaptively, considering the characteristics of the representations. Moreover, the study of the scoring function can be regarded as an attempt to maximize the outlier detection capability from the extracted representation. For this reason, we focus more on extracting or training the representations in DNNs. (2) Moreover, our work tackles deep anomaly detection in an unsupervised manner for practicality as we barely know which types of outliers the model will encounter, and constructing a trainable OOD dataset also costs a fortune and effort. (3) Finally, among various input types, we focus on studying the method of processing images and sentences, representing sequential and non-sequential data and the most frequently used inputs when humans interact.

2.5 Challenges and Relevant Literature

2.5.1 Challenges

Several factors make anomaly detection a complex and long-standing problem:

1. **Machine learning’s strong reliance on the close-set assumption.**

The machine learning techniques approximate the target distribution based on the currently available data and consider this distribution as a real-world target distribution. Speaking otherwise, they regard acquired

instances are independent and identically distributed (i.e., IID assumption). However, real-world inputs deviate from the approximated distribution. Thus, even well-trained neural models are known to be vulnerable to test inputs from an unknown distribution and assign arbitrarily high probability on the unfamiliar test samples (Hendrycks and Gimpel, 2017; Hein et al., 2019).

2. **The definition of outlier is subjective.** Since the criteria for determining normal and abnormal are usually subjective, setting a firm decision boundary between normal and anomalous behavior is often imprecise. Due to this nature, methods are often not transferable to other domains as they have to meet additional constraints.
3. **Evolving nature of outliers.** In the real-world scenario, inputs to the model evolve ceaselessly, causing variation between current concepts of normal and the future.
4. **Maliciously designed inputs.** Some outliers are maliciously designed to fool the model and look similar to normal inputs. They are tough to distinguish compared to other outliers, which occur by chance or mistake.
5. **Noise in the training dataset.** The dataset often contains noise, erroneous data, or outliers, which tend to be similar to the actual anomalies.

2.5.2 Related Research Fields

The previously-mentioned problems make some other literature challenging as well as anomaly detection, and sometimes they factors considered independent and important sub-problems. For example, adversarial machine learning is an area of research that deals with the third factor intensely. Adversarial machine

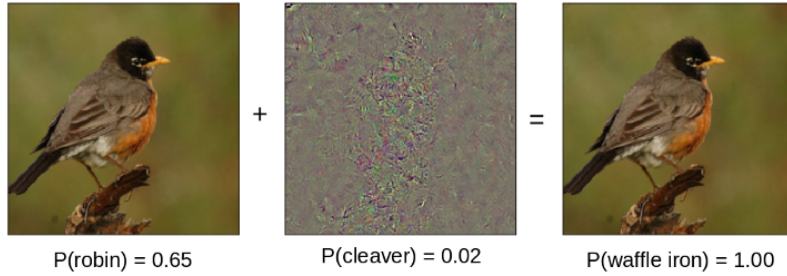


Figure 2.4: Illustration of adversarial attack. Predictive label for robin image changes into waffle iron with small perturbations.

learning researches malicious attacks on DNNs and aims to make DNNs more robust against such attacks. Adversarial examples, adding imperceptible perturbations to original inputs (See Fig. 2.4 for an example of adversarial sample²), can easily fool DNNs and prompt the model to make a wrong prediction with high confidence. Likewise, noise removal or data cleansing is a research area that aims to detect and correct (or remove) corrupted or inaccurate instances by identifying incomplete, incorrect, or irrelevant samples. In addition, domain adaptation (or generalization) is also closely-related to anomaly detection. Domain adaptation is to maximize or preserve the performance from the trained *source domains* to a different (but related) *target domain*, which can be considered as a subcategory of transfer learning. While the primary goal of anomaly detection is to find distributional shifted inputs to maximize reliability, domain adaptation attempts to maximize the generalization ability so that the model can achieve maximum performance even when the input is shifted from the training dataset.

²The images are from <https://towardsdatascience.com/know-your-enemy-7f7c5038bdf3>.

Part I: Deep Self-Supervised Anomaly Detection Methods (Chapter 3 & 4)

Part I of this dissertation (**Chapter 3 & 4**) primarily focuses on (1) developing self-supervised anomaly detection frameworks in an (2) unsupervised manner for (3) image and sentence datasets. Among the numerous approaches in deep unsupervised anomaly detection, self-supervised learning (SSL) is spurring interest and validating its superiority over previous DAD methods (Minderer et al., 2020; Geirhos et al., 2020; Golan and El-Yaniv, 2018; Zhan et al., 2021; Schwag et al., 2021) due to its ability to learn complex and diverse representations without additional labels. Moreover, self-supervised learning is known to restrain the DNNs from learning *shortcuts* (Geirhos et al., 2020) — decision rules that perform well on train distribution but fail to transfer to others — which cause spurious output to both IND and OOD inputs from the neural models. DNNs are prone to learn shortcuts, such as detecting grass instead of cows, as shown in Fig.2.5, since leveraging them is much easier and takes the *least effort* (Geirhos et al., 2020; Kluckhohn, 1950) for DNNs to yield the intended solution. These shortcuts can be mitigated by making the task more complex, and multi-tasking pre-text tasks (SSL) in tandem with the original



Figure 2.5: Recognition algorithms generalize poorly to new environments. Cows in ‘common’ contexts (e.g. Alpine pastures) are detected and classified correctly (A), while cows in uncommon contexts (beach, waves and boat) are not detected (B) or classified poorly (C). Images are from (Beery et al., 2018)

task exhibits good efficiency in alleviating shortcuts and improving the quality of the trained representations additionally.

Leveraging the aforementioned characteristics of self-supervised learning, **Part I** of this dissertation proposes two self-supervised frameworks for image input and sentence input, respectively. Before delving into the specifics, we introduce some self-supervised learning techniques frequently used in this part of the dissertation as a preliminary.

Preliminaries.

Contrastive Learning.

Recent contrastive learning algorithms, *e.g.*, SimCLR (Chen et al., 2020a), learn representations by maximizing the agreement between differently augmented views of the same image while repelling others in the batch. Specifically, each

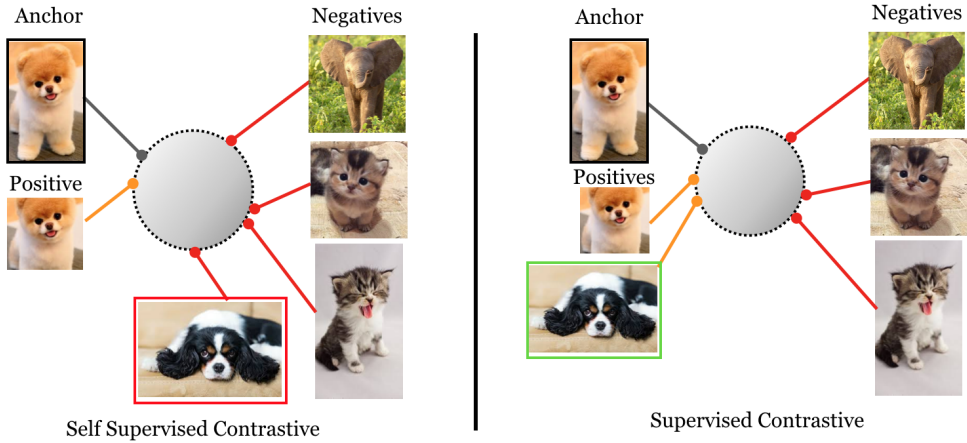


Figure 2.6: Vanilla self-supervised contrastive learning and supervised contrastive learning. The supervised contrastive loss (right) contrasts the set of all samples from the same class as positives against the negatives from the remainder of the batch, while self-supervised contrastive loss contrasts a single positive for each anchor (i.e., an augmented version of the same image). The figure illustrates that taking class label information into account allows the aligning of representations from the same class more tightly. The image is from Khosla et al. (2020).

image \mathbf{x}_k from randomly sampled batch $\mathcal{B} = \{(\mathbf{x}_k, y_k)\}_{k=1}^N$ is augmented twice, generating an independent pair of views $(\bar{\mathbf{x}}_{2k-1}, \bar{\mathbf{x}}_{2k})$ and augmented batch $\bar{\mathcal{B}} = \{(\bar{\mathbf{x}}_k, \bar{y}_k)\}_{k=1}^{2N}$, where labels of augmented views $\bar{y}_{2k-1}, \bar{y}_{2k}$ are equal to original label y_k . The augmented pair of views, $\bar{\mathbf{x}}_{2k-1} = t(\mathbf{x}_k)$ and $\bar{\mathbf{x}}_{2k} = t'(\mathbf{x}_k)$, are generated via independent transformation instance t and t' , drawn from pre-defined augmentation function family \mathcal{T} . $(\bar{\mathbf{x}}_{2k-1}, \bar{\mathbf{x}}_{2k})$, then are passed sequentially through encoder network f_θ and projection head g_ϕ , yielding latent vectors $(\mathbf{z}_{2k-1}, \mathbf{z}_{2k})$ that are utilized for the contrastive loss (i.e., NT-Xent):

$$\ell(i, j) = -\log \frac{\exp(\text{sim}_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}_{i,k}/\tau)}, \quad (2.3)$$

where $\text{sim}_{i,j} = \mathbf{z}_i^\top \mathbf{z}_j / (\|\mathbf{z}_i\| \|\mathbf{z}_j\|)$ denotes cosine similarity between pair of latent vectors in $(\mathbf{z}_i, \mathbf{z}_j)$ and τ stands for temperature hyper-parameter. The final objective is to minimize Eq. 2.3 over positive pairs, which maps the input into effective individual representation in a *task-agnostic* way:

$$\mathcal{L}_{\text{SimCLR}} = \frac{1}{2N} \sum_{k=1}^N [\ell(2k-1, 2k) + \ell(2k, 2k-1)]. \quad (2.4)$$

Supervised Contrastive Learning.

Supervised contrastive learning (SCL) is a supervised variant of vanilla contrastive learning, which employs label information of the input to group samples into known classes more tightly. Thus, SCL can learn *data-label* relationships as well as *data-data* relationships as in CL.

In SCL, each batch $\mathcal{B} = \{(\mathbf{x}_b, y_b)\}_{b=1}^{|\mathcal{B}|}$ in the dataset, where \mathbf{x}_b, y_b denotes a sentence and a label for index b respectively, generates an augmented batch $\bar{\mathcal{B}} = \{(\bar{\mathbf{x}}_b, \bar{y}_b)\}_{b=1}^{|\bar{\mathcal{B}}|}$, where labels of augmented views are preserved as the original one. The augmented batch $\bar{\mathcal{B}}$ consists of two augmented input; $\bar{\mathbf{x}}_{2b-1} = t_1(\mathbf{x}_b)$ and $\bar{\mathbf{x}}_{2b} = t_2(\mathbf{x}_b)$, where t_1, t_2 indicate data augmentation functions. Then, $(\bar{\mathbf{x}}_{2b-1}, \bar{\mathbf{x}}_{2b})$ are passed through PLM and projector, generating latent vectors $(\mathbf{z}_{2b-1}, \mathbf{z}_{2b})$ that are utilized to calculate the supervised contrastive loss:

$$\mathcal{L}_{\text{SCL}} = -\log \sum_{j \in P(i)} \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{\sum_{k=1}^{|\bar{\mathcal{B}}|} \mathbb{1}_{[k \neq i]} \exp(\mathbf{z}_i \cdot \mathbf{z}_k / \tau)}, \quad (2.5)$$

where $P(i) = \{p \in \mathcal{B} : \bar{y}_j = \bar{y}_i\}$ is the set of indices of all positives in the augmented batch with query index i and τ represents temperature hyper-parameter.

Chapter 3

A Deep Self-Supervised Anomaly Detection for Images

3.1 Introduction

Over the past few years, machine learning has achieved immense success surpassing human-level performance in many tasks, such as classification, segmentation, and object detection (Tan and Le, 2019; Tan et al., 2020; Chen et al., 2020a). However, such a well-trained model assigns arbitrary high probability (Hein et al., 2019) on the unfamiliar test samples, since most machine learning systems generally depend on the closed-set assumption (*i.e.*, i.i.d. assumption). This phenomenon may lead to a fatal accident in safety-critical applications like medical-diagnosis or autonomous driving. *Anomaly detection*¹ is a research area that aims to circumvent such symptoms by identifying whether the test samples come from in-distribution or not. A flurry of recent deep-learning based models,

¹also termed *out-of-distribution detection*, *novelty detection*, or *outlier detection* in the contemporary machine learning context.

including reconstruction based (Oza and Patel, 2019; Li et al., 2018), density estimation based (Malinin and Gales, 2018), post-processing methods (Lee et al., 2018b; Liang et al., 2017), and self-supervised learning methods (Golan and El-Yaniv, 2018; Hendrycks et al., 2019a; Tack et al., 2020; Winkens et al., 2020), have been proposed for the task and have shown noticeable progress.

Among the numerous approaches mentioned, self-supervised learning (SSL) is in the limelight and validating its superiority over previous methods in various research areas (Devlin et al., 2019; Chen et al., 2020a). Since it is unfeasible to access *out-of-distribution* (OOD) data in most real-world scenarios, the ability of SSL to learn complex and diverse representations without additional labels is receiving much attention from anomaly detection lately. (Golan and El-Yaniv, 2018) is one of the earlier works to identify the potential of SSL and has proposed a simple yet effective technique that aims to learn intrinsic features within in-distribution (IND) samples via auxiliary tasks (*e.g.*, predicting flip, rotation, or translation of input data). Furthermore, (Hendrycks et al., 2019a) confirmed that using such auxiliary tasks not only helps to determine anomalous samples but also helps to defend against adversarial attacks. More recent works (Tack et al., 2020; Winkens et al., 2020) exploit contrastive learning (CL), especially SimCLR (Chen et al., 2020a), that learns individual data representations in a *task-agnostic* way by maximizing the agreement between differently augmented views of the same image while repelling others in the batch.

SimCLR obtains effective individual representation for each data point, as well as clustered representations for each class, even without any human label or supervision. (See Fig. 3.1a.) However, its *task-agnostic* feature results in blurry boundaries between each cluster, so it requires a fine-tuning process used for some downstream tasks (*e.g.*, multi-class classification). Such process undermines expression ability well-learned through SimCLR, given that most

of the fine-tuning procedure leverages *cross-entropy loss* and it solely considers class labels of the data without taking into account unique characteristics of the data or similarity between them. Consequently, the fine-tuned model often assigns high confidence probabilities to OOD input, reducing the distributional discrepancy between IND and OOD (Hein et al., 2019).

Our foremost insight is that forming dense clusters without fine-tuning while preserving individual representations by inheriting the advantages of SimCLR will shape a more meaningful visual representation contrary to the *pre-train then tune* paradigm, thus contributing to the effective detection of anomalous data. To this end, we propose a *task-specific* variant of contrastive learning called *masked contrastive learning* (MCL), which can shape more clear boundaries between each class. (See Fig.3.1d) The core idea of MCL is to generate a mask that can adjust the repelling-ratio properly by considering class labels in the batch. Experimental results show that MCL is more befitted to anomaly detection than SimCLR or its other *task-specific* variant (*i.e.*, SupCLR), which still exhibits blurry decision boundaries (See Fig.3.1b).

Moreover, contrary to the previous belief that the auxiliary self-supervision task (*e.g.*, predicting flip, rotation, or translation of input data) does not substantially improve label classification accuracy (Hendrycks et al., 2019a), we observe that it is possible to considerably improve both IND and OOD performance with a proper inference method. To this end, we propose *self-ensemble inference* (SEI) that fully exploits ability learned from simple auxiliary self-supervision task in the inference phase. SEI enhances model performance in all situations without losing generality and can be used in any classifier. By combining our models, we can outperform previous state-of-the-art methods.

Our main contributions are summarized as below:

- We propose a novel extension to contrastive learning dubbed *masked con-*

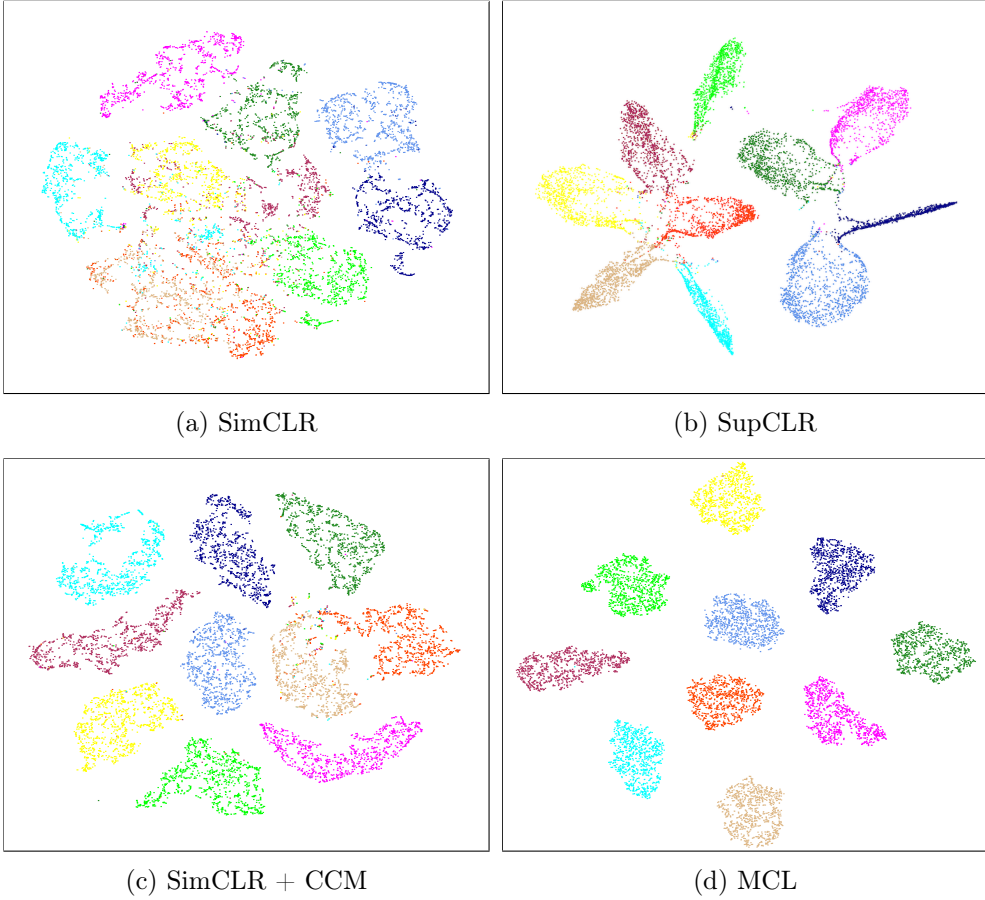


Figure 3.1: t -SNE visualization of CIFAR-10 trained representation. Each color denotes each class labels in CIFAR-10. Each component in our model makes each cluster more dense.

trastive learning which can shape dense class-conditional clusters.

- We also propose an inference method called *self-ensemble inference* (SEI) that fully leverages ability learned from auxiliary self-supervision tasks in test time. SEI can further boost both IND and OOD performance.
- We validate our approaches on various image benchmark datasets, where we obtain significant performance gain over the previous state-of-the-art.

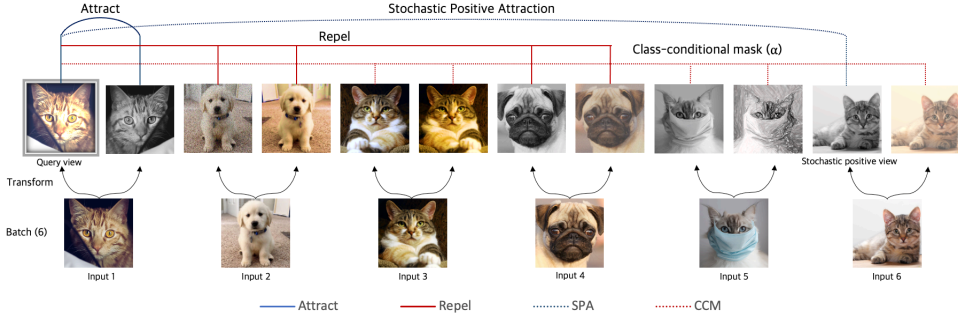


Figure 3.2: Description of MCL framework. Query view (gray-framed) attracts views connected with a blue-colored line and repels views connected with a red-colored line.

3.2 Masked Contrastive Learning

As the name implies, our method adopts contrastive learning, particularly SimCLR, with two additional components: *class-conditional mask* and *stochastic positive attraction*; see Fig. 3.2. In this section, we provide detailed explanations of each component in MCL. (See Sec. 3.5 for further details regarding contrastive learning.)

3.2.1 Class-Conditional Mask

The benefit of CL in anomaly detection has been reported recently (Winkens et al., 2020; Tack et al., 2020). Nonetheless, we found that well-formed representations from CL, which facilitate distinguishing anomalous data, are lost during the fine-tuning procedure. (See Sec. 3.4.2 for more detailed results.) Due to the *task-agnostic* nature of CL, however, the fine-tuning steps are essential, making it difficult to avoid the aforementioned phenomenon. MCL mitigates such symptoms by injecting *task-specific* characteristics to existing CL, resulting in fine-tuning procedure inessential. One key component in MCL is *class-conditional mask* (CCM) which is a simple yet effective masking technique that

adaptively determines the repelling-ratio considering the label information in each $\bar{\mathcal{B}}$. CCM can be defined as follows:

$$\text{CCM}(i, j) = \begin{cases} \alpha & \text{if } \bar{y}_i = \bar{y}_j \\ 1/\tau & \text{if } \bar{y}_i \neq \bar{y}_j, \end{cases} \quad (3.1)$$

where $0 < \alpha < 1/\tau$. CCM alters the temperature for the same label views to a smaller value α , so that query view repels views with the same label relatively small amount compared to views with different labels. The generated CCM is then multiplied to the similarity score in Eq. 2.3, modifying the previous SimCLR loss to the following equation.

$$p_{\text{ccm}}(i, j) = \frac{\exp(\text{sim}_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}_{i,k} \text{CCM}(i, k))}, \quad (3.2)$$

$$\ell_{\text{ccm}}(i, j) = -\log p_{\text{ccm}}(i, j), \quad (3.3)$$

$$\mathcal{L}_{\text{ccm}} = \frac{1}{2N} \sum_{k=1}^N [\ell_{\text{ccm}}(2k-1, 2k) + \ell_{\text{ccm}}(2k, 2k-1)]. \quad (3.4)$$

Penalizing a small ratio α to positive views restrains respective representation in the same cluster from being too similar to each other, making individual data representation more distinctive.

3.2.2 Stochastic Positive Attraction

As can be seen in Fig. 3.1c, CCM promotes a more label-wise cluster compared to SimCLR. Even in CCM, however, the core operating principle is still identical to SimCLR in that it only attracts the view from the same image while it repels remaining views within the batch. Due to this repulsive nature, each data representation gets more distant as training continues, and it leads to the formation of unsatisfactory scattered clusters. To alleviate this phenomenon, we add

another component named *stochastic positive attraction* (SPA), an additional attraction with the stochastically sampled view in the batch. Specifically, SPA attracts query $\bar{\mathbf{x}}_i$ with stochastic positive sample $(\bar{\mathbf{x}}_j, \bar{y}_j) \sim \mathcal{U}(\bar{\mathcal{B}}_i^+)$ in the positive augmented batch $\bar{\mathcal{B}}_i^+$ for query $\bar{\mathbf{x}}_i$, where \mathcal{U} refers to the discrete uniform distribution. The positive augmented batch $\bar{\mathcal{B}}_i^+$ for query $\bar{\mathbf{x}}_i$ contains views with the same label except views from its parent image $\mathbf{x}_{(i-1)\setminus 2}$, where the symbol \setminus denotes integer quotient operator:

$$\bar{\mathcal{B}}_i^+ = \{(\bar{\mathbf{x}}_k, \bar{y}_k) \in \bar{\mathcal{B}} \mid \bar{y}_k = \bar{y}_i \text{ and } (k-1) \setminus 2 \neq (i-1) \setminus 2\}. \quad (3.5)$$

CCM is also used for negative views with the additional constraint which excludes views from its parent image. SPA for query view $\bar{\mathbf{x}}_i$ now can be defined as follows:

$$p_{\text{spa}}(i, j) = \frac{\exp(\text{sim}_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[(k-1)\setminus 2 \neq (i-1)\setminus 2]} \exp(\text{sim}_{i,k} \text{CCM}(i, k))}, \quad (3.6)$$

$$\ell_{\text{spa}}(i) = \mathbb{E}_{(\bar{\mathbf{x}}_j, \bar{y}_j) \sim \mathcal{U}(\bar{\mathcal{B}}_i^+)} [-\log p_{\text{spa}}(i, j)]. \quad (3.7)$$

The complete version of MCL is acquired by combining CCM and SPA, where the overall loss term being as follows:

$$\mathcal{L}_{\text{MCL}} = \mathcal{L}_{\text{ccm}} + \frac{\lambda}{2N} \sum_{k=1}^{2N} \ell_{\text{spa}}(k), \quad (3.8)$$

where λ denotes weight hyper-parameter for SPA loss.

SPA possesses the potential to make respective clusters denser when α meet certain conditions. Otherwise, SPA either loses the ability to assemble scattered clusters, or forces every data points to converge near the centroid of each cluster, making the data points within the cluster indistinguishable. We explore the two aforementioned conditions theoretically in the following section:

Attraction Condition

This section details the condition to maintain the ability of SPA to assemble a scattered cluster. Note that the role of SPA is to alleviate the same-class repulsion in MCL.

Let q be a query index in the augmented batch $\bar{\mathcal{B}}$ of size $2N$ and r be the index of a stochastic positive sample from $\bar{\mathcal{B}}_q^+$. If the sign of the gradient of $\mathcal{L}_{\text{spa}} = \frac{1}{2N} \sum_{i=1}^{2N} \ell_{\text{spa}}(i)$ w.r.t. $\text{sim}_{q,r}$ is negative, then the cosine similarity between two vectors $\bar{\mathbf{z}}_q, \bar{\mathbf{z}}_r$ gets higher as training continues. Such fact indicates that the ability of SPA comes when the gradient of SPA loss is negative.

From SPA loss,

$$p_{\text{spa}}(i, j) = \frac{\exp(\text{sim}_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[(k-1)\setminus 2 \neq (i-1)\setminus 2]} \exp(\text{sim}_{i,k} \text{CCM}(i, k))}, \quad (3.9)$$

$$\begin{aligned} \ell_{\text{spa}}(i) &= \mathbb{E}_{(\bar{\mathbf{x}}_j, \bar{\mathbf{y}}_j) \sim \mathcal{U}(\bar{\mathcal{B}}_i^+)} [-\log p_{\text{spa}}(i, j)] \\ &\doteq \frac{1}{|\bar{\mathcal{B}}_i^+|} \sum_{(\bar{\mathbf{x}}_j, \bar{\mathbf{y}}_j) \in \bar{\mathcal{B}}_i^+} -\log p_{\text{spa}}(i, j). \end{aligned} \quad (3.10)$$

Then, gradient of \mathcal{L}_{spa} w.r.t. $\text{sim}_{q,r}$ can be calculated as follows:

$$\begin{aligned} \frac{\partial \mathcal{L}_{\text{spa}}}{\partial \text{sim}_{q,r}} &= \frac{1}{2N} \sum_{i=1}^{2N} \frac{\partial \ell_{\text{spa}}(i)}{\partial \text{sim}_{q,r}} = \frac{1}{2N} \frac{\partial \ell_{\text{spa}}(q)}{\partial \text{sim}_{q,r}} \\ &= \frac{1}{2N} \frac{1}{|\bar{\mathcal{B}}_q^+|} \left(-\frac{1}{\tau} + |\bar{\mathcal{B}}_q^+| \alpha p_{\text{spa}}(q, r) \right) \\ &= \frac{1}{2N} \left(\alpha p_{\text{spa}}(q, r) - \frac{1}{\tau |\bar{\mathcal{B}}_q^+|} \right) \\ &< \frac{1}{2N} \left(\alpha - \frac{1}{\tau |\bar{\mathcal{B}}_q^+|} \right), \end{aligned} \quad (3.11)$$

where $\text{CCM}(q, r) = \alpha$ and $p_{\text{spa}}(i, j) < 1$ for any (i, j) .

To this end, α should be selected carefully which met following condition:

$$\alpha < \frac{1}{\tau|\bar{\mathcal{B}}_q^+|}. \quad (3.12)$$

Convergence Condition

This section details the condition to restrain SPA from forcing every data point to converge near the centroid of each cluster.

When α is set to an extremely small value, the attraction from SPA becomes dominant compared to the repulsion of CCM, leading respective data points within each cluster to converges near its centroid. In concrete, let \bar{i} be the index for another augmented view from same parent image $x_{(i-1)\setminus 2}$. Then we can revise MCL loss as follows:

$$\begin{aligned} \mathcal{L}_{\text{MCL}} &= \mathcal{L}_{\text{ccm}} + \lambda \mathcal{L}_{\text{spa}} \\ &= \frac{1}{2N} \sum_{i=1}^{2N} \ell_{\text{ccm}}(i, \bar{i}) + \lambda \mathcal{L}_{\text{spa}}. \end{aligned} \quad (3.13)$$

The overall gradient for MCL becomes:

$$\frac{\partial \mathcal{L}_{\text{MCL}}}{\partial \text{sim}_{q,r}} = \frac{1}{2N} \sum_{i=1}^{2N} \frac{\partial \ell_{\text{ccm}}(i, \bar{i})}{\partial \text{sim}_{q,r}} + \frac{\partial \mathcal{L}_{\text{spa}}}{\partial \text{sim}_{q,r}}. \quad (3.14)$$

By the definition of ℓ_{ccm} ,

$$p_{\text{ccm}}(i, j) = \frac{\exp(\text{sim}_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}_{i,k} \text{CCM}(i, k))}, \quad (3.15)$$

$$\ell_{\text{ccm}}(i, j) = -\log p_{\text{ccm}}(i, j), \quad (3.16)$$

where $\bar{\mathcal{B}}_q^+$ guarantees $r \neq \bar{q}$. Then, the gradient of $\partial \ell_{\text{ccm}}(i, \bar{i})$ *w.r.t.* $\text{sim}_{q,r}$ becomes:

$$\frac{1}{2N} \sum_{i=1}^{2N} \frac{\partial \ell_{\text{ccm}}(i, \bar{i})}{\partial \text{sim}_{q,r}} = \frac{1}{2N} \frac{\partial \ell_{\text{ccm}}(q, \bar{q})}{\partial \text{sim}_{q,r}} = \frac{1}{2N} \alpha p_{\text{ccm}}(q, r). \quad (3.17)$$

Thus, the respective term of MCL can be rewritten as follows:

$$\begin{aligned} \frac{\partial \mathcal{L}_{\text{MCL}}}{\partial \text{sim}_{q,r}} &= \frac{1}{2N} \alpha p_{\text{ccm}}(q, r) + \frac{\lambda}{2N} \left(\alpha p_{\text{spa}}(q, r) - \frac{1}{\tau |\bar{\mathcal{B}}_q^+|} \right) \\ &< \frac{1}{2N} \alpha + \frac{\lambda}{2N} \left(\alpha - \frac{1}{\tau |\bar{\mathcal{B}}_q^+|} \right) \\ &= \frac{(1+\lambda)}{2N} \left(\alpha - \frac{\lambda}{\tau(1+\lambda) |\bar{\mathcal{B}}_q^+|} \right) \\ &\leq \frac{(1+\lambda)}{2N} \left(\alpha - \frac{\lambda}{\tau(1+\lambda)(2N-2)} \right) \end{aligned} \quad (3.18)$$

When $\alpha < \lambda \cdot \{\tau(1+\lambda)(2N-2)\}^{-1}$, the overall gradient of MCL becomes negative and urges every data points to gather near the centroid of each cluster. So hyper-parameter α should be carefully selected to avoid the following condition:

$$\alpha < \frac{\lambda}{\tau(1+\lambda)(2N-2)}. \quad (3.19)$$

3.2.3 Training Auxiliary task in MCL

Training simple auxiliary self-supervision task along with the main downstream task is possible in MCL by adding constraint in CCM. Let T_{main} be the main task with C_{main} number of classes, T_{aux} be an auxiliary task with C_{aux} number of classes and corresponding augmented batch be $\bar{\mathcal{B}} = \{(\bar{x}_i, \bar{y}_i^{\text{main}}, \bar{y}_i^{\text{aux}})\}_{i=1}^{2N}$ with additional auxiliary task label. Then CCM with auxiliary self-supervision task can be defined as follows:

$$\text{CCM}_{\text{aux}}(i, j) = \begin{cases} \alpha & \text{if } \bar{y}_i^{\text{main}} = \bar{y}_j^{\text{main}} \text{ and } \bar{y}_i^{\text{aux}} = \bar{y}_j^{\text{aux}} \\ \beta & \text{if } \bar{y}_i^{\text{main}} = \bar{y}_j^{\text{main}} \text{ and } \bar{y}_i^{\text{aux}} \neq \bar{y}_j^{\text{aux}} \\ 1/\tau & \text{otherwise.} \end{cases} \quad (3.20)$$

By simply setting $\beta = 1/\tau$, it is possible to train $C_{\text{main}} \times C_{\text{aux}}$ distinctive clusters for each $(\bar{y}_j^{\text{main}}, \bar{y}_j^{\text{aux}})$ pairs. Since the auxiliary task plays a complementary role to the main task, it is more plausible to form grouped clusters for respective main labels and to have distinctive clusters for each auxiliary label inside them. With appropriate constraint (*i.e.*, $0 < \alpha < \beta < 1/\tau$), MCL forms hierarchical clusters by dint of CCM.

3.3 Inference

3.3.1 Scoring Function in MCL

Since there is no task-specific final layer in MCL, classification or anomaly detection are conducted via class-wise density estimation analogous to (Lee et al., 2018b), utilizing negative Mahalanobis distance $-d_M$ as a scoring function s :

$$s_i(\mathbf{z}) = -d_M(\mathbf{z}, \mu_i; \Sigma_i) = (\mathbf{z} - \mu_i)^\top \Sigma_i^{-1} (\mathbf{z} - \mu_i), \quad (3.21)$$

$$S(\mathbf{x}) = [s_1(\mathbf{x}), s_2(\mathbf{x}), \dots, s_{C_{\text{main}}}(\mathbf{x})], \quad (3.22)$$

where $\mathbf{z} = g_\phi(f_\theta(\mathbf{x}))$, and μ_i, Σ_i refer to mean and covariance matrix of n -dimensional multivariate normal distribution (MND) $\mathcal{N}(\mu_i, \Sigma_i)$ for class $i \in I = \{1, 2, \dots, C_{\text{main}}\}$. Note that calculating MNDs for each class is a one-time operation acquired from training data. The vector $S(\mathbf{x})$ contains scores of each label for image \mathbf{x} and the class label with highest score $i^* = \operatorname{argmax}_{n \in I} S_n(\mathbf{x})$ is selected as a predictive label, where $S_n(\mathbf{x})$ denotes n -th element of $S(\mathbf{x})$. The corresponding IND score for predictive label $s_{i^*}(\mathbf{x})$ measures the confidence for predictive label i^* which are used to distinguish OOD data, following the binary decision function h_δ from below:

$$h_\delta(\mathbf{x}) = \begin{cases} \text{IND} & S_{i^*}(\mathbf{x}) \geq \delta \\ \text{OOD} & S_{i^*}(\mathbf{x}) < \delta, \end{cases} \quad (3.23)$$

where δ denotes anomaly threshold.

3.3.2 Self Ensemble Inference

The key idea of SEI is to exploit the model’s ability to discriminate within IND, learned through an auxiliary self-supervision task, in the inference phase. For example, consider predicting 4-directional rotations (from 0° , 90° , 180° , to 270°) is employed for an auxiliary task. Then SEI ensembles the results from corresponding the 4-rotated test images and derives calibrated index i^* and corresponding score s_{i^*} . Specifically, let $i \in I = \{1, 2, \dots, C_{\text{main}}\}$ be the main task label, and $j \in J = \{1, 2, \dots, C_{\text{aux}}\}$ be the auxiliary task label. Then, $C_{\text{main}} \times C_{\text{aux}}$ number of MNDs $\mathcal{N}(\mu_i^{(j)}, \Sigma_i^{(j)})$ are calculated for every label combinations. The test image \mathbf{x} is augmented C_{aux} times, yielding $\{\mathbf{x}^{(j)}\}_{j=1}^{C_{\text{aux}}}$. Each augmented test image $\mathbf{x}^{(m)}$ with the class label $y^{\text{aux}} = m$ is fed into the corresponding MND $\mathcal{N}(\mu_i^{(m)}, \Sigma_i^{(m)})$, where $i \in I$ and $j = m$, yielding a score vector $S^{(m)}(\mathbf{x})$:

$$s_i^{(j)}(\mathbf{x}) = -d_M(g_\phi(f_\theta(\mathbf{x})), \mu_i^{(j)}; \Sigma_i^{(j)}), \quad (3.24)$$

$$S^{(m)}(\mathbf{x}) = \left[s_1^{(m)}(\mathbf{x}), s_2^{(m)}(\mathbf{x}), \dots, s_{C_{\text{main}}}^{(m)}(\mathbf{x}) \right]. \quad (3.25)$$

Our goal is to aggregate $\{S^{(j)}(\mathbf{x})\}_{j=1}^{C_{\text{aux}}}$ properly to make model more robust and reliable. We considered 3 different aggregation methods to extract predictive label i^* . The foremost intuitive way is averaging the main label scores across $\{S^{(j)}(\mathbf{x})\}_{j=1}^{C_{\text{aux}}}$.

$$i_{\text{avg}}^*(\mathbf{x}) = \operatorname{argmax}_{i \in I} \frac{1}{|J|} \sum_{m \in J} S_i^{(m)}(\mathbf{x}). \quad (3.26)$$

Another variation is to select label index from the highest IND score:

$$i_{\text{max}}^*(\mathbf{x}) = \operatorname{argmax}_{i \in I} \left\{ \max_{m \in J} S_i^{(m)}(\mathbf{x}) \right\}. \quad (3.27)$$

The last aggregation is the weighted-average, which gives adaptive weights to each score in S . Weights for each score are computed per j using the harmonic mean to penalize exceptionally low scores for better calibration:

$$W^{(m)}(\mathbf{x}) = \left(\sum_{n \in I} \frac{1}{S_n^{(m)}(\mathbf{x})} \right)^{-1}. \quad (3.28)$$

$$i_{\text{w-avg}}^*(\mathbf{x}) = \operatorname{argmax}_{i \in I} \frac{\sum_{m \in J} W^{(m)}(\mathbf{x}) S_i^{(m)}(\mathbf{x})}{\sum_{m \in J} W^{(m)}(\mathbf{x})}. \quad (3.29)$$

Corresponding score to predictive label i^* is used to distinguish OOD data following Eq. 3.23. Depending on the aggregation method, the effect that the model can achieve varies.

3.4 Experiment

In this section, we demonstrate the effectiveness of MCL and SEI on several multi-class image classification datasets.

Experiment configurations. In the following experiments, we adopt ResNet-34 (He et al., 2016) with a single projection head, following structure used to train CIFAR-10 in SimCLR. We also fixed hyper-parameters related to contrastive learning following SimCLR, which include transformation $\mathcal{T} = \{\text{color jittering, horizontal flip, grayscale, inception crop}\}$, the strength of color distortion to 0.5, batch size to 1024, and temperature τ to 0.2, to keep our experiment tractable. Unlike SimCLR, we used SGD optimizer with learning rate $1.2 (0.3 \times \text{batch size} / 256)$, decay $1\text{e-}6$, and momentum 0.9. Furthermore, we use a cosine annealing scheduler without any warm-up.

For MCL hyper-parameters, we set α to 0.05, β to 2.5, and λ to 1 which meets 2 conditions for MCL. We carefully selected the remaining MCL-related hyper-parameters, α and β , that meet 2 conditions from the previous subsection.

To calculate exact value, we substitute the corresponding hyper-parameter value to Eq. 3.12 and Eq. 3.19. We approximated $|\bar{\mathcal{B}}_q^+| \approx 2N/C = 51.2$, since the total number of classes C in MCL with auxiliary task is $C = 10 \times 4 = 40$. Then, Eq. 3.12 is substituted as follows:

$$\alpha < \frac{1}{\tau|\bar{\mathcal{B}}_q^+|} \approx 0.098 \quad (3.30)$$

The final value for α is set to 0.05 which meets both conditions in Eq. 3.12 and Eq. 3.19. Remaining hyper-parameter β is set to $1/(2 \cdot \tau) = 2.5$.

Evaluation metrics. To evaluate IND detection performance, we measured the label classification accuracy. To verify OOD performance, we utilized following metrics:

- **AUROC:** AUROC stands for *area under the receiver operating characteristic curve* (AUROC), which is a threshold (δ in Sec. 3.3.1) free metric and the most common metric in anomaly detection literature. Higher AUROC indicates the better OOD performance.
- **FPR@95:** The ROC curve is a graph where the x-axis and y-axis in a graph indicate FPR and TPR respectively. FPR@95 is a specific FPR point of the ROC curve when TPR is set to 95. Lower FPR indicates fewer false positives arises.
- **AUPR:** AUPR stands for *area under the precision-recall curve* (PR-curve). In particular, two PR-curves are produced, from IND perspective PR-curve and OOD perspective PR-curve. Higher AUPR indicates high precision and recall value.

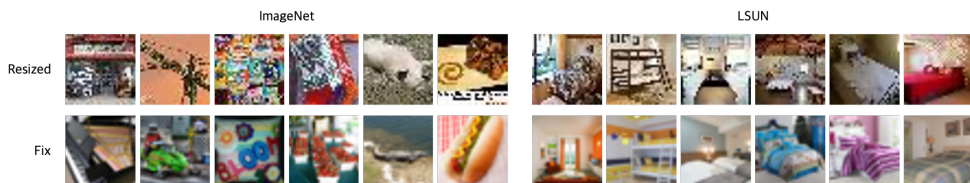


Figure 3.3: Samples from resized and fixed versions of ImageNet and LSUN for OOD detection. Because the resizing was done by subsampling, it caused an aliasing effect, resulting in unintended artifacts that made images almost indistinguishable even for humans. Fixed version used interpolation technique for the resizing.

Dataset Description We conduct experiments on various multi-class benchmark datasets: CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), SVHN (Netzer et al., 2011), Tiny-ImageNet (Deng et al., 2009b), and LSUN (Yu et al., 2015). CIFAR-10, CIFAR-100, Tiny-ImageNet dataset is a widely used multi-class natural image (*e.g.*, dog, tiger, plane) dataset with 10,100, and 200 classes label for each dataset. SVHN dataset stands for Street View House Numbers (SVHN), which consist of ten-digit classes. LSUN dataset contains millions of color images with 10 scene categories and 20 object categories. Since the size of each dataset differs, several data sets have to be resized to unify to fit the targeted size (32 by 32). (Liang et al., 2017) released resized version of Tiny-ImageNet and LSUN dataset, however, as can be seen in Fig. 3.3, those version contains unintended artifacts produced during the resizing process. (Tack et al., 2020) handled the aforesaid problem and released a fixed version of those datasets dubbed Tiny-ImageNet(F) and LSUN(F).

Competing methods. To investigate the performance of our model, we compare our model (MCL) along with several recently proposed methods and baseline model.

- Baseline (Hendrycks and Gimpel, 2017) classifies OOD input by its max-

imum softmax probability (MSP).

- Mahalanobis Distance (Lee et al., 2018b) classifies OOD input by measuring the distance between the input and the estimated normal distribution of the train data.
- Auxiliary Rotation (Hendrycks et al., 2019a) trains classifier with addition self-supervised task to predicts the degree of the input rotation.
- CSI (Tack et al., 2020) leverage SupCLR (Khosla et al., 2020) and well-selected self-supervision task befitted for anomaly detection.
- ODIN (Liang et al., 2017) is a postprocessing method by calibrating the softmax probability with the gradient of the input.
- Outlier Exposure (Hendrycks et al., 2019b) utilize additional supervised OOD data to calibrate OOD input to yield low probability.

Selecting self-supervision tasks. Since the complex auxiliary task is not our primary concern, we considered rotation, horizontal flip, and translation as candidates for auxiliary tasks, which are simple and commonly used in the area of anomaly detection (Golan and El-Yaniv, 2018). However, MCL contains inception crop (Szegedy et al., 2015) and horizontal flip in contrastive transformation \mathcal{T} , so using translations or horizontal flip as an auxiliary task only confuses the model. To this end, we employed predicting 4-directional rotations (0° , 90° , 180° , and 270°) as our auxiliary task.

3.4.1 Multi-Class Anomaly Detection

We trained our model on CIFAR-10 (Krizhevsky et al., 2009) as IND, and used CIFAR-100, SVHN (Netzer et al., 2011), ImageNet (Deng et al., 2009b), and

Training Method	Test Acc.	AUROC			
		SVHN	LSUN	ImageNet	CIFAR100
Baseline	93.6	89.9	84.3	88.0	86.4
ODIN*	93.6	85.8	83.2	87.7	85.8
Mahalanobis Distance*	94.1	99.1	87.9	90.9	88.2
Auxiliary Rotation	94.3	97.3	94.5	94.7	90.7
Outlier Exposure [†]	-	98.4	-	-	93.3
SupCLR [‡]	93.8	97.3	91.6	90.5	88.6
CSI + SupCLR [‡]	94.8	96.5	92.1	92.4	90.5
CSI + SupCLR + Ens [‡]	96.1	97.9	93.5	94.0	92.2
Auxiliary Rotation + SEI	95.8	98.4	95.7	95.8	92.3
MCL + SEI (ours) [‡]	95.9	98.9	96.0	95.9	93.1
MCL + SEI (ours)	96.4	99.3	96.3	96.5	94.0

* refers models with additional post-processing procedure.

[†] denotes models with additional supervised (OOD) data.

[‡] use ResNet-18 as a backbone network, otherwise ResNet-34 are used.

Table 3.1: Test accuracy of in-domain classification and AUROC of OOD data for each model trained on CIFAR-10 dataset.

LSUN (Yu et al., 2015) datasets for OOD. Note that all the classes in OOD datasets are disjoint with CIFAR-10.

Table 3.1 summarizes our experimental results of our model and competing methods. First, MCL outperforms other methods by a significant margin regardless of the datasets. MCL shows performance improvement in IND metric (accuracy) as well as in OOD metrics, suggesting that there is a potential for expansion in other tasks. Interestingly, our model outperforms the combination of CSI (augmentation optimized for OOD detection) and another task-specific variant SupCLR. Such observation indicates that MCL, which can learn unique individual representation for each data point, is more suited for OOD detection than SupCLR. Moreover, MCL outperforms supervised method (Hendrycks et al., 2019b), which utilizes additional explicit OOD data.

Model	Acc	AUROC
Baseline (w/o NT-Xent)	93.61	86.40
SimCLR (CE fine-tuned)	93.88	87.56
SimCLR (joint fine-tuned)	93.91	87.81
MCL (w/o SPA, w/o aux)	91.41	85.03
MCL (w/o aux)	94.35	89.49
MCL (CE fine-tuned)	94.22	88.89
MCL	94.03	91.12

Table 3.2: Ablation studies for each component in MCL.

3.4.2 Ablation Study

In this section, we perform an ablation study on our proposed methods, along with baselines. In all experiments, we treated CIFAR-10 as IND and CIFAR-100 as OOD.

Masked Contrastive Learning

We conduct ablation experiments to explore the effectiveness of the main components in MCL. We sequentially added each component in MCL that are CCM, SPA, auxiliary 4-way rotation task, and SEI. Tab. 3.2 reports our ablation experiments, along with baseline models. For SimCLR based models, we fine-tuned pre-trained model in two ways. One way is to use *cross-entropy loss* which is traditional fine-tuning method in classification task, and the other way is to use *cross-entropy loss* along with SimCLR loss (Eq. 2.4) jointly (Winkens et al., 2020). As (Hendrycks et al., 2019a) argued, learning the auxiliary task makes model more robust and improves AUROC but it does not improve accuracy.

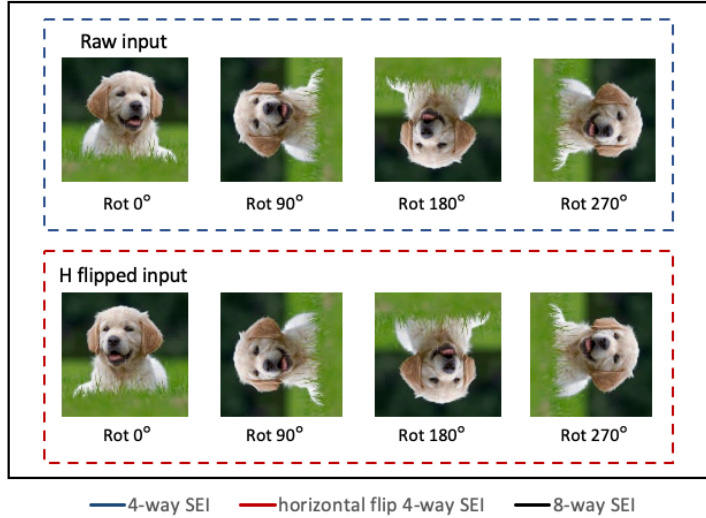


Figure 3.4: 4-way SEI and 8-way SEI.

Self-Ensemble Inference 1

In this part, we share our findings on SEI with its variations (average, maximum, and weighted-average). Since we employed a 4-directional rotation prediction as our auxiliary task, SEI is done in a 4-way correspondingly. It is also possible to add additional 4-way SEI with a horizontally flipped image, leading to 8-way SEI. (Fig. 3.4 provides a visual explanation.) 8-way SEI follows the same strategy introduced earlier. The only difference is the number of augmented image to ensemble. As claimed in previous experiments, learning the auxiliary task alone can not improve accuracy. But with the help of SEI, we were able to achieve performance gains in both accuracy and AUROC regardless of its variations as can be seen in Tab. 3.3, . Moreover, the performance gain differs depending on the aggregation method. Especially, average SEI makes the model robust to input variation which is a commonly known benefit of the ensemble, bringing noticeable AUROC gain compared to accuracy. Meanwhile, maximum

Model	Agg	Acc	AUROC
MCL + w/o SEI	-	94.03	91.12
MCL + 4-way SEI	avg	94.68	93.20
	max	95.97	92.30
	w-avg	96.12	93.29
MCL + 8-way SEI	avg	94.74	93.37
	max	96.40	92.00
	w-avg	96.43	94.06

Table 3.3: Ablation studies for SEI. MCL is trained with additional 4-way rotations auxiliary task.

SEI yields a significant gain in accuracy, which indicates MCL’s prediction with high confidence score is quite precise. The weighted-average SEI absorbed both ensemble effect and confidence gain by assigning adaptive weights to its score. Moreover, SEI can also be used with general *empirical-risk minimization* classifier. As shown in Tab. 3.1, the auxiliary task (4-way rotations) trained classifier can achieve significant performance gain in both IND and OOD with SEI.

Self-Ensemble Inference 2

We also conduct ablation experiments on SEI and the relationship with the auxiliary task. We applied SEI to 3 differently trained model as follows:

- **MCL without auxiliary task:** MCL is trained neither with auxiliary task nor data augmentation
- **MCL with data augmentation:** MCL trained with additional 4-way rotated images without rotation labels.
- **MCL with auxiliary task:** MCL is trained with additional 4-way rotated images with rotation labels.

Model	Acc \uparrow	CIFAR-100			
		AUROC \uparrow	FPR@95 \downarrow	AUPR (IND) \uparrow	AUPR (OOD) \uparrow
MCL (w/o SPA, w/o aux)	91.41	85.03	60.70	84.26	83.65
MCL (w/o aux)	94.35	90.49	52.69	91.28	87.58
MCL (w/o aux) + DA	92.55	90.07	54.93	91.17	87.98
MCL (w/o aux) + DA + SEI	94.70	92.08	44.94	92.92	90.55
MCL	94.03	91.12	48.16	91.80	89.24
MCL + fine-tuning	94.22	88.89	62.84	90.20	84.70
MCL + SEI	96.43	94.06	32.67	94.59	93.21

Table 3.4: Ablation study for MCL with additional OOD metrics.

Training Method	Test Acc.	AUROC			
		SVHN	LSUN	ImageNet	CIFAR100
SupCLR	93.8	97.3	91.6	90.5	88.6
MCL (Ours)	93.1	97.9	93.8	93.6	90.8
MCL + SEI (Ours)	95.9	98.9	96.0	95.9	93.1

Table 3.5: Performance comparison between MCL and SupCLR. Models are trained on CIFAR-10 dataset with ResNet-18.

Tab. 3.4 summarizes our extended ablation study for SEI. Applying SEI to MCL without auxiliary task only confuses the model and undermines both IND and OOD performance substantially, as rotated images are unseen during training phase. If the rotated image is augmented without a label, the model performance is degraded but the SEI shows a slight improvement. Finally, when the auxiliary task is explicitly trained with the main downstream task, SEI shows significant gain without any performance degradation. Such experimental results reveal that it is necessary to additionally train the auxiliary task to use SEI properly.




















Comparison with SupCLR

We compared MCL’s performance with SupCLR (Khosla et al., 2020), another *task-specific* variant of CL. Table 3.5 summarizes performance comparison with SupCLR. Since SupCLR forces all positive views to have a high similarity to the query view, the ability to distinguish data within the same class is diluted.

In MCL, α in CCM suppresses positive samples having too high similarity with query view. Unlike SupCLR which does not repel positive views in a batch, MCL also repels positive views by a small ratio α . By penalizing a small amount α to positive views, MCL can discern each data in the same label cluster. Such technique is analogous to the smoothing (LS) technique a widespread strategy to prevent over-confident prediction problems in general supervised classifiers, where α plays similar a role to smoothing-ratio in LS. As a side note, the test accuracy can be further improved by increasing the number of stochastic samples in the MCL or by increasing λ in SPA. However, mentioned tactics decrease discrepancy between IND distribution and OOD distribution, undermining the AUROC score. Since test-accuracy is not the primary concern in anomaly detection, hyper-parameters and model components in MCL are set to the value that shows the best AUROC performance.

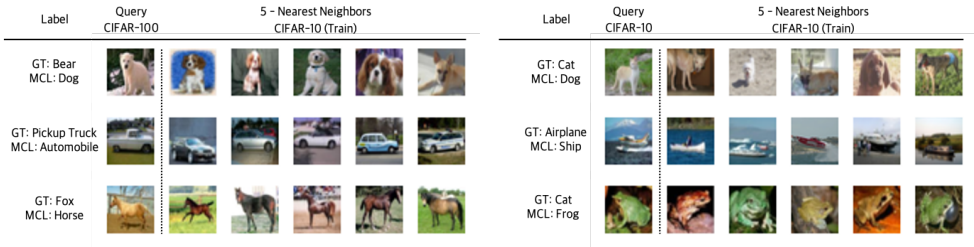
3.4.3 Qualitative results

In this section, we analyze the data distribution along with several failure cases of our model. As can be seen in Fig. 3.7, both OOD data distribution and wrongly classified data distribution have lower scores compared to correctly classified samples, which indicates that our model can measure predictive uncertainty quite precisely. Furthermore, to analyze our failure cases in detail, we conducted a case study on OOD samples with high confidence and wrongly classified IND samples. In MCL, representation is learned based on semantic similarity, so it is feasible to conjecture model’s decision-making process via observing nearest neighbors of the input. For most failure cases, their predicted label and ground-truth label (GT) belongs to the same super-class regardless of the aforementioned failure types. For example, MCL predicts a bear (OOD) as a dog (IND); likewise, a cat as a dog (both IND), which belongs to the same

Label	Query CIFAR-100	3 - Nearest Neighbors CIFAR-10 (Train)			Label	Query CIFAR-10	3 - Nearest Neighbors CIFAR-10 (Train)		
GT: Possum MCL: Cat					GT: Cat MCL: Dog				
GT: Possum MCL: Dog					GT: Deer MCL: Bird				
GT: Bear MCL: Dog					GT: Dog MCL: Horse				
GT: Fox MCL: Cat					GT: Dog MCL: Cat				
GT: Lion MCL: Cat					GT: Horse MCL: Deer				
GT: Hamster MCL: Cat					GT: Cat MCL: Bird				
GT: Fox MCL: Deer					GT: Cat MCL: Dog				
GT: Possum MCL: Cat					GT: Deer MCL: Cat				
GT: Pickup Truck MCL: Automobile					GT: Automobile MCL: Truck				
GT: Streetcar MCL: Truck					GT: Airplane MCL: Bird				
GT: Pickup Truck MCL: Automobile					GT: Ship MCL: Truck				
GT: Pickup Truck MCL: Automobile					GT: Automobile MCL: Truck				

(a) OOD samples with high IND scores. (b) Wrongly classified samples from IND.

Figure 3.5: More results on CIFAR-10 with CIFAR-100 as OOD. GT stands for the ground truth label. Each row shows a query image on the second column, followed by the top 3 most similar images of train data. Quite an amount of samples are either wrong, ambiguous, or vague. (a) Query images are from OOD, but the model took as an IND sample and classified with high confidence. (b) Wrongly classified samples from in-domain classification.



(a) OOD samples with high IND scores. (b) Wrongly classified samples from IND.

Figure 3.6: Case studies on OOD samples with high confidence and wrongly classified IND samples.

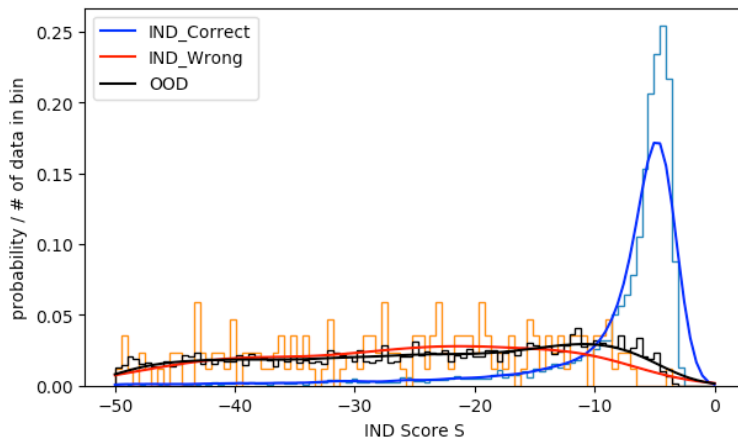


Figure 3.7: Histogram and PDF for correctly classified IND, wrongly classified IND and OOD distribution. MCL measures predictive uncertainty quite precisely.

super-class, mammal. As a side note, few data were mislabeled, as can be seen in Fig. 3.6.(a).(GT: Fox) Fig. 3.5 provides more case studies.

3.5 Related Work

Anomaly detection. Anomaly detection, also termed OOD, novelty, or outlier detection, is a research area that aims to identify anomalies by distinguish-

ing whether the test sample is drawn from in-distribution or not. Most of the recent methods in anomaly detection use deep-learning which can be branched into the following categories:

Generative and hybrid models use a generative model to measure the uncertainty of test data. The most common method is measuring reconstruction error, assuming that the model learns a proper mapping function that successfully reconstructs normal data samples with a very small reconstruction error (Oza and Patel, 2019; Li et al., 2018; Schlegl et al., 2017). Another methodology is to train the distribution of the normal data so that the model can assign a low likelihood to anomalous data (Zhang et al., 2020; Gal and Ghahramani, 2016; Malinin and Gales, 2018; Blundell et al., 2015).

Discriminative models leverage information from trained classifiers such as *maximum softmax probability* (MSP), or Mahalanobis distance of latent feature. (Hendrycks and Gimpel, 2017) presented the most natural baseline for a discriminative model (classifier) which distinguishes anomalies by the predictive probability (MSP) of the classifier. (Lee et al., 2018b; Liang et al., 2017) proposed a post-processing method that adds a small adversarial like perturbation on the input. (Hendrycks et al., 2019b) proposed a supervised OOD method by enforcing uniform distribution to anomalous data. More recently, Self-supervised Learning is spurring interest in anomaly detection, since access to OOD data in the most real-world scenario is quite unfeasible. (Golan and El-Yaniv, 2018), one of the earlier works to identify the potential of SSL, proposed a simple but effective technique that aims to discriminate within in-distribution (IND) samples through 3 auxiliary tasks. (*e.g.*, flip, rotation, and translation) Furthermore, (Hendrycks et al., 2019a) confirmed that using auxiliary tasks not only helps to determine OOD samples but also helps to defend against adversarial attacks. In more recent works, (Tack et al., 2020; Winkens et al., 2020) used

contrastive learning to get well-suited representation for anomaly detection. In (Tack et al., 2020), particularly, the performance improvement was achieved by considering transforms known to be harmful as negative. (*e.g.*, rotations, cutout) Our method belongs to self-supervised learning, which exploits both auxiliary self-supervision task and contrastive learning.

Contrastive learning. Contrastive learning is a specific framework of self-supervised learning, which has shown impressive results in visual representation learning tasks (Chen et al., 2020a,b). Most recent work in OOD (Tack et al., 2020; Winkens et al., 2020) report that employing CL improves OOD performance. Our work goes further from the previous papers and proposes a *task-specific* variant of CL. (Khosla et al., 2020) proposed SupCLR, another *task-specific* variant of SimCLR, which is a noteworthy work. Similar to MCL, SupCLR also leverages label information in the batch while training and shows superior accuracy over SimCLR. Despite its performance in IND accuracy, representation from SupCLR which is not entirely appropriate for discerning anomalous data, as it attracts all same label views which discrepancy in each class disappears.

Chapter 4

A Deep Self-Supervised Anomaly Detection for Sentences

4.1 Introduction

Natural language understanding (NLU) in dialog systems, which often formalizes as a classification task to identify intentions behind user input, is a vital component as their decision propagates to the downstream pipelines. Numerous works have achieved immense success on sundry tasks (*e.g.*, intention classification, NLI, QA) reaching parity with human performance (Wang et al., 2019b). Despite their success in many different benchmarks, neural models are known to be vulnerable to test inputs from an unknown distribution (Hendrycks and Gimpel, 2017; Hein et al., 2019), commonly referred to as outliers, since they depend strongly on the closed-world assumption (i.e., I.I.D assumption). Thus, out-of-distribution (OOD) detection (Aggarwal, 2017), which aims to discern outliers from the train distribution, is an essential research problem for ensuring a high-quality user experience and maintaining strong reliability as the systems

in the wild encounter myriad unseen data ceaselessly.

The most prevailing paradigm in OOD detection is to *extract* and *score*. Namely, it extracts the representation of the input from a neural model and passes it to a pre-defined scoring function. Then, the scoring function gauges the appropriateness of the input based on the extracted feature and decides whether the input is from the normal distribution. The most common rule of thumb for extracting representation from neural models is employing the the last layer, a simple and intuitive way to obtain a holistic representation, which is universally utilized in broad machine learning areas.

Meanwhile, previous studies (Tenney et al., 2019; Clark et al., 2019) revealed that the middle layers of the language model also conceal copious information. For instance, prior studies on language model probing suggest that syntactic linguistic knowledge is most prominent in the middle layers (Hewitt and Manning, 2019; Goldberg, 2019; Jawahar et al., 2019), and semantic knowledge in BERT is spread in all layers widely (Tenney et al., 2019). In this regard, leveraging intermediate layers can lead to a better OOD detection performance, as they retain some complementary information to the last layer feature, which might be beneficial in discriminating outliers. Several studies (Shen et al., 2021; Sastry and Oore, 2020; Lee et al., 2018b) have shown empirical evidence that intermediate representations are indeed beneficial in detecting outliers. Precisely, they attempted to utilize middle layers via naïvely aggregating the individual result of every single intermediate feature *explicitly*.

Although previous studies have shown the potential of intermediate layer representations in OOD detection, we confirmed that the aforementioned naïve ensemble scheme spawns several problems: (Fig. 4.1 illustrates OOD performance of the layer-wise and their explicit ensemble in two different datasets.) The first problem we observed is that the ensemble result (red bar) nor the last

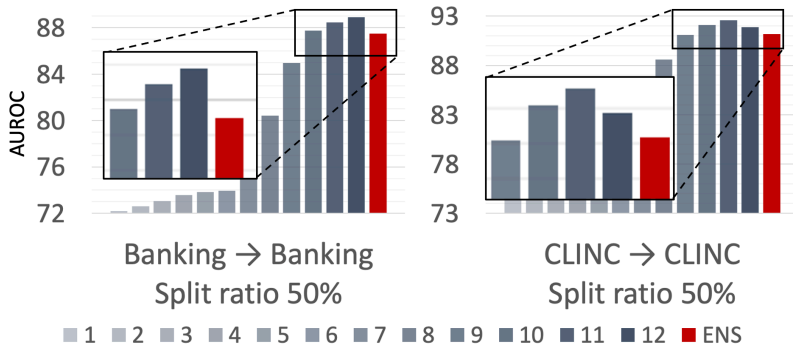


Figure 4.1: Layer-wise and explicit ensemble (Shen et al., 2021) performance of baseline BERT. Explicit ensemble sometimes lead to worse AUROC (higher the better) than using a well-performing single layer representation, depending on the setting. Detailed explanations about setting and baseline model are elaborated in Sec.4.4.2 and Sec.4.4.3 individually.

layer can not guarantee the best performance among the entire layer depending on the setting. Such a phenomenon raises the necessity for a more elaborate approach of deriving a more meaningful ensemble representation from various representations rather than a current simple summation or selecting a single layer. Secondly, even when this explicit ensemble gives a sound performance, it requires multiple computations of the scoring function by birth. Thus, explicit ensemble inevitably delays the detecting time, which is a critical shortcoming in OOD detection, as swift and precise decision-making is the cornerstone in this area.

To remedy the limitations of the explicit ensemble schemes, we propose a novel framework dubbed Layer-agnostic Contrastive Learning (LaCL). Our framework is inspired by the foundation of an ensemble, which seeks a more calibrated output by combining heterogeneous decisions from multiple models

(Kuncheva and Whitaker, 2003; Gashler et al., 2008). Specifically, LaCL regards intermediate layers as independent decision-makers and assembles them into a single vector to yield a more accurate prediction: LaCL makes middle-layer representations richer and more diverse by injecting the advantage of contrastive learning (CL) into intermediate layers while suppressing inter-layer representations from being similar through additional regularization loss. Then, LaCL assembles them into a single ensemble representation *implicitly* to circumvent multiple computations of the scoring function.

We demonstrate the effectiveness of our approach in 9 different OOD scenarios where LaCL consistently surpasses other competitive works and their explicit ensemble performance by a significant margin. Moreover, we conducted an in-depth analysis of LaCL to elucidate its behavior in conjunction with our intuition.

4.2 Related Work

OOD detection. Methodologies in OOD detection can be divided into supervised (Hendrycks et al., 2019b; Lee et al., 2018a; Dhamija et al., 2018) and unsupervised settings according to the presence of training data from OOD. Since the scope of OOD covers nigh infinite space, gathering the data in the whole OOD space is infeasible. For this realistic reason, the most recent OOD detection studies generally discriminate OOD input in an unsupervised manner, including this work. Numerous branches of machine learning tactics are employed for unsupervised OOD detection: generating pseudo-OOD data (Chen and Yu, 2021; Zheng et al., 2020), Bayesian methods (Malinin and Gales, 2018), self-supervised learning based approaches (Moon et al., 2021; Manolache et al., 2021; Li et al., 2021; Zhou et al., 2021; Zeng et al., 2021; Zhan et al., 2021),

and novel scoring functions which measure the uncertainty of the given input (Hendrycks and Gimpel, 2017; Lee et al., 2018b; Liu et al., 2020; Tack et al., 2020).

Contrastive learning & OOD detection. Among the numerous approaches mentioned, contrastive learning (CL) based methods (Chen et al., 2020a; Zbon-tar et al., 2021; Grill et al., 2020) are recently spurring predominant interest in OOD detection research. The superiority of CL in OOD detection comes from the fact that it can guide a neural model to learn semantic similarity within data instances. Such property is also precious for unsupervised OOD detection, as there is no accessible clue regarding outliers or abnormal distribution. Despite its potential, CL has been utilized in the computer vision field (Cho et al., 2021; Schwag et al., 2021; Tack et al., 2020; Winkens et al., 2020) in the early works due to its high reliance on data augmentation. However, now it is also widely used in various NLP applications with the help of recent progress (Li et al., 2021; Liu et al., 2021a; Kim et al., 2021; Carlsson et al., 2020; Gao et al., 2021b; Sennrich et al., 2016). Specifically, Li et al. (2021) verified that CL is also helpful in the NLP field, and Zhou et al. (2021); Zeng et al. (2021) redesigned the contrastive-learning objective into a more appropriate form for OOD detection.

Potential of intermediate representation. The leading driver of the recent upheaval in NLP is the pre-trained language model (PLM), such as BERT (Devlin et al., 2019) and GPT (Radford et al., 2018), which trains a large-scale dataset on a transformer-based architecture (Vaswani et al., 2017). Numerous studies attempted to reveal the role and characteristics of each layer in PLMs and verified that diverse information is concealed in the middle layer, which is now a pervasive notion in the machine learning community. For instance, Tenney et al. (2019) showed that the different layers of the BERT network could

resolve syntactic and semantic structure within a sentence. Clark et al. (2019) proposed an attention-based probing classifier leveraging syntactic information in the middle layer of BERT. Several studies (Shen et al., 2021; Sastry and Oore, 2020; Lee et al., 2018b) have shown the potential of intermediate representations in OOD detection by explicitly aggregating the individual result of every single intermediate feature.

4.3 Layer-agnostic Contrastive Learning

4.3.1 Intuition

The prime objective of our framework is to assemble rich information in the entire layers into a single ensemble representation to derive a more reliable decision. Inspired by the foundation of ensemble learning, which seeks better predictive performance by combining the predictions from multiple models, we regard each intermediate layer as an independent model (or decision maker). To make each layer a better decision-maker, LaCL injects a sound representation learning signal (i.e., supervised contrastive learning) to the entire layer by training objective function in a layer-agnostic manner to engage every layer more directly. Additionally, we propose correlation regularization loss (CR loss) which decorrelates a pair of strongly correlated adjacent representations to encourage each layer to learn layer-specialized representations from complementary information of each layer. Then, the global compression layer (GCL) *implicitly* assembles various features in each layer into a single calibrated ensemble representation. In the following subsections, we explain the components of our model in detail.

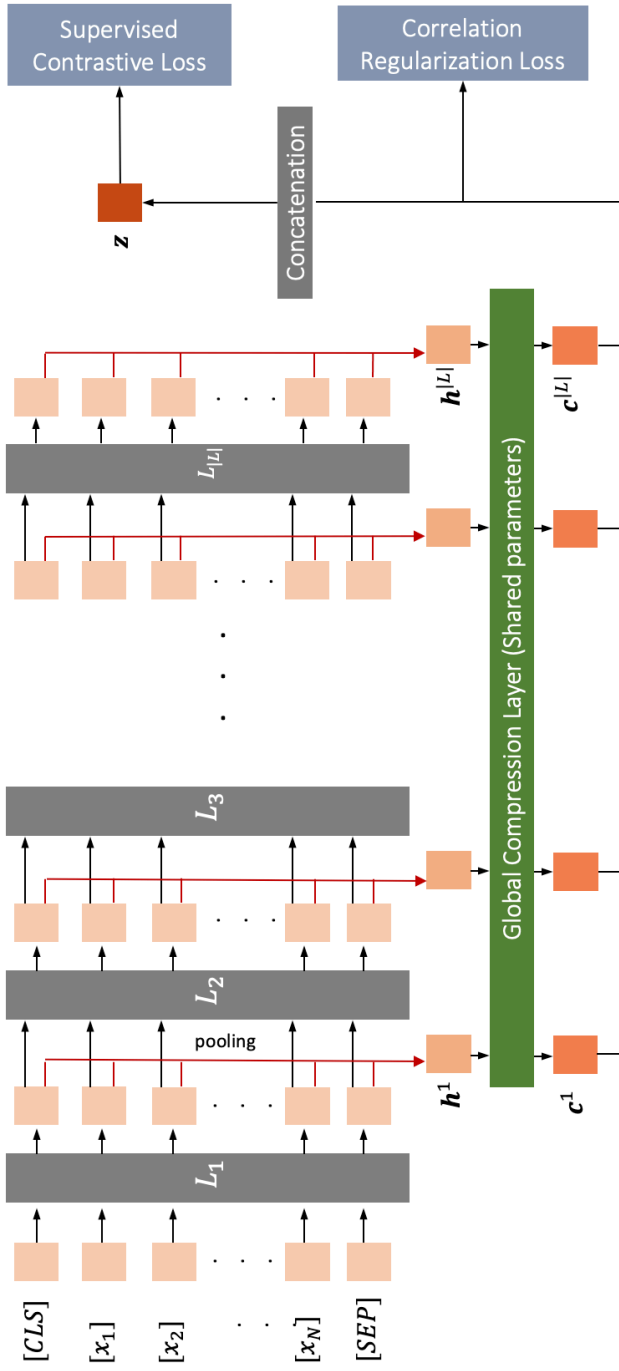


Figure 4.2: Overall structure of Layer-agnostic Contrastive Learning (LaCL). The global compression layer trains the SCL loss in a *layer-agnostic* manner by engaging entire layers in the CL task. And the correlation regularization (CR) loss decorrelates each intermediate layer to avoid overlapping information between each layer.

4.3.2 Global Compression Layer

The global compression layer (GCL) is a two-layer MLP that is directly connected to entire layers to assemble intermediate representations into a single representation \mathbf{z} . GCL can be viewed as a particular type of projection head in contrastive learning. By linking the projection head to the entire layer, GCL facilitates layer-agnostic training to engage every middle layer in a training objective directly.

The process of extracting the final latent vector \mathbf{z} with GCL is as follows: (The batch index term b is omitted for brevity from now.)

First, each layer l ($l \in |L|$, where $|L|$ refers to the cardinality of the layers) in PLM, outputs token embeddings $\mathbf{H}^l = [\mathbf{h}_1^l, \mathbf{h}_2^l, \dots, \mathbf{h}_{len(\mathbf{x})}^l]$ for sentence \mathbf{x} . Then we combine token embeddings \mathbf{H}^l into a single vector $\mathbf{h}^l = \text{pool}(\mathbf{H}^l)$ by applying the pooling function (i.e., mean pooling). Lastly, GCL receives the pooled token embedding of each layer \mathbf{h}^l (where, $\mathbf{h}^l \in R^{|D|}$) as an input and outputs compact low-dimensional representation \mathbf{c}^l (where, $\mathbf{c}^l \in R^{|D|/|L|}$). And we concatenate all compact representations \mathbf{c}^l to generate a single sentence representation \mathbf{z} from \mathbf{x} :

$$\mathbf{z}(\mathbf{x}) = [\mathbf{c}^1 \oplus \mathbf{c}^2 \oplus \mathbf{c}^3 \oplus \dots \oplus \mathbf{c}^{|L|}], \quad (4.1)$$

where \oplus indicates concatenation and $\mathbf{z} \in R^{|D|}$.

LaCL trains the SCL loss with the final representation from GCL \mathbf{z} , which inheres information from entire layers.

4.3.3 Correlation Regularization Loss

The correlation regularization (CR) loss restrains a pair of features from each adjacent layer from being similar, following the intuition of an ensemble where its performance boost springs from various decisions (Kuncheva and Whitaker,

2003; Gashler et al., 2008). Specifically, it encourages adjacent layers to activate different dimensions given the same input. First, we define the correlation in the dimension d of the adjacent layer (l and $l + 1$) as follows:

$$cor_{(l,l+1)}^d = \frac{\sum_b \mathbf{c}_{b,d}^l \cdot \mathbf{c}_{b,d}^{l+1}}{\sqrt{\sum_b (\mathbf{c}_{b,d}^l)^2} \sqrt{\sum_b (\mathbf{c}_{b,d}^{l+1})^2}}. \quad (4.2)$$

where d indicates the index of hidden embedding dimension ($d \in |D|/|L|$, where $\mathbf{c}^l \in R^{|D|/|L|}$) and b refers to a data index of the augmented batch $\tilde{\mathcal{B}}$.

Then, the CR loss selects a strongly correlated dimension set S by picking the dimensions that exceed the pre-set margin value m and decorrelates set S iterating over every adjacent layer:

$$S = \{d \in |D| : cor_{(l,l+1)}^d \geq m\}$$

$$\mathcal{L}_{\text{CR}} = \sum_l \sum_{d \in S} cor_{(l,l+1)}^d. \quad (4.3)$$

Finally, the overall loss term for LaCL can be described as follows:

$$\mathcal{L}_{\text{LaCL}} = \mathcal{L}_{\text{SCL}} + \lambda_1 \mathcal{L}_{\text{CR}}, \quad (4.4)$$

where λ_1 denote weights for CR loss.

4.3.4 Classification & OOD Scoring

Since there is no task-specific final layer (i.e., classification layer for cross-entropy loss) in LaCL, classification and anomaly detection are conducted via a cosine similarity scoring function (Tack et al., 2020). Employing the cosine similarity scoring function in LaCL is straightforward and shows good compatibility, as the model trained with contrastive learning can measure meaningful cosine similarity between data instances.

For input \mathbf{x} , we first extract the implicit ensemble representation $\mathbf{z}(\mathbf{x})$ and find the nearest neighbor instance \mathbf{x}_{nn} , i.e., $\max_{nn} \text{sim}(\mathbf{z}(\mathbf{x}), \mathbf{z}(\mathbf{x}_{nn}))$, from the

training dataset. Then we classify label of \mathbf{x} as the label of the nearest neighbor y_{nn} . And for the OOD detection, we use the similarity between input and its nearest neighbor as follows:

$$\text{Score}(\mathbf{x}) = \text{sim}(\mathbf{z}(\mathbf{x}), \mathbf{z}(\mathbf{x}_{nn})) \quad (4.5)$$

Finally, we decide whether the input \mathbf{x} is outlier or not through following the binary decision function I_δ :

$$I_\delta(\mathbf{x}) = \begin{cases} \text{IND} & \text{Score}(\mathbf{x}) \geq \delta \\ \text{OOD} & \text{Score}(\mathbf{x}) < \delta, \end{cases} \quad (4.6)$$

where δ denotes anomaly threshold, usually obtained from a score of the training instance which is in the boundary of the pre-set *true positive rate*.

4.3.5 Augmentation for Contrastive Learning

Augmentation is a crucial factor in CL that directly influence the model performance. To find the most effective data augmentation for OOD, we carefully select six data augmentation tactics for contrastive learning: back-translation (BT) (Li et al., 2021), dropout (DO) (Gao et al., 2021b), token cutoff (Yan et al., 2021; Shen et al., 2020), random span masking (RSM) (Liu et al., 2021a), and token shuffling (Lee et al., 2020):

Back-Translation is a method of translating a raw sentence into another language and then re-translating it back into the same language. Precisely, we translate raw sentence into german and re-translate it back into english utilizing 'transformer.wmt19.en-de.single_model', 'transformer.wmt19.de-en.single_model' from fairseq (Ott et al., 2019). In order to avoid the BT sentence from being completely identical to the original sentence, we generated top-5 sentences and sampled from them after checking the duplicates.

Dropout (Gao et al., 2021b) utilize dropout layers in transformers (Vaswani et al., 2017) to extract stochastically different representation. Due to dropout layer, giving the same input to the same model yields slight different representation and dropout utilize those inputs as a augmentation. Note that, dropout is always applied by default.

Random Span Masking (RSM) first randomly select some span, i.e., k continuous characters, in the input sequence. Then, they randomly replaced with [MASK] token. In general, RSM is apply in one instance of the two augmented instances, as it was proposed in the original MirrorBERT paper (Liu et al., 2021a) In this paper, we additionally consider applying it on both side of a pair.

Token Shuffling Token shuffling randomly shuffles the order of the input tokens (positional embedding) in the input sequence.

Token Cutoff In token cutoff is a simple strategy that eliminates some input tokens randomly.

As our final data augmentation tactics, we greedily combined two best-performing augmentations, i.e., BT and RSM.

Instance 1 (t_1): Raw data + RSM + DO

Instance 2 (t_2): BT + RSM + DO Note that DO is always applied by default unless the dropout probability is specified to 0 manually since it utilizes a dropout layer inside the transformer (Vaswani et al., 2017). We explain each augmentation and report their performance in the Sec. 4.5.2.

Dataset	Avg length	# Domain	# Intent	# Class
CLINC150	8.31	10	15	150
Banking77	11.9	1	77	77
Snips	9.05	7	-	7

Table 4.1: Dataset statistics.

4.4 Experiments

4.4.1 Implementation Details

In the following experiments, we adopt BERT-base (Devlin et al., 2019) as a backbone of our network. We fixed the dimension of the first layer in GCL to 1024 and the dimension of the second layer to $64 = 768 / (num_layers)$ so that the dimension of the concatenated vector \mathbf{z} is 768 (BERT-base embedding dimension). We used mean pooling as a token embedding pooling function, set temperature τ to 0.05, CR loss weight λ_2 to 1, and margin m in CR loss to 0.5. Moreover, we set the batch size to 128 and used AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate 1e-5 with a cosine annealing scheduler.

4.4.2 Dataset and Metrics

In order to investigate the performance of our model in many different situations, we conduct experiments on intention classification datasets. Generally, intention classification classes are organized hierarchically, which often consist of domains (*e.g.*, banking, travel, reservation) and intents (*e.g.*, banking - transfer money, banking - check account) where one domain serve as a parent category of multiple intents. It is much demanding to distinguish unknown intent under equivalent domain than discerning unseen domain (Zhang et al., 2022a), as the domain is a high-level concept. Considering the above facts, we carefully se-

CLINC150	Snips
play_music	PlayMusic
update_playlist	AddToPlaylist
weather	GetWeather
confirm_reservation restaurant_reservation cancel_reservation accept_reservations	BookRestaurant

Table 4.2: Overlapping classes between CLINC and Snips dataset.

lected CLINC150 (Larson et al., 2019), Banking77 (Casanueva et al., 2020), and Snips (Coucke et al., 2018) datasets, which comprise distinct class hierarchies and are also commonly used in OOD detection literature.

Specifically, the CLINC150 dataset contains various domains and intents, so it is a favorable dataset to measure overall model performance. In the case of the Banking77 dataset, it consists of fine-grained 77 intents under a single banking domain. On the other hand, the Snips dataset comprises seven different domains, making each class relatively easy to discern. (See Tab. 4.1 for statistics about each dataset.)

Utilizing the selected dataset, we measure OOD performance in 9 different scenarios that can be categorized into the following two settings that are widely used in OOD detection:

- **Close-OOD setting (splitting dataset)** refers to a setting when the test distribution (OOD distribution) is *close* to the train distribution. Usually, close-OOD setting is simulated by partitioning one dataset into 2 disjoint datasets (i.e., IND / OOD dataset) based on the class label. Since the IND and

OOD datasets originated from the equivalent dataset, they share similar distributions and properties, making the task more demanding. In our experiments, we randomly partitioned the class labels in each dataset with three different ratios (25%, 50%, and 75%), following the validation sets-up in previous works (Shu et al., 2017; Fei and Liu, 2016; Lin and Xu, 2019).

- **Far-OOD setting (distinct dataset)** refers to a setting when the test distribution (OOD distribution) is far from the IND train distribution. So far-OOD is relatively easy to discern test samples from the normal distribution. Usually, far-OOD setting is simulated by regarding the disjoint dataset as a test dataset (OOD dataset). i.e., CLINC150 (IND) \rightarrow Banking77 (OOD) or Snips (OOD). In some scenarios, we verified that some intents belong to both IND and OOD, so we manually removed overlapping intents before training. (Details about removed intents in each scenario are in the Tab.4.2.) We also categorize CLINC150 (OOD) \rightarrow CLINC150 OOD split (OOD)¹ as far-OOD, since previous work (Zhang et al., 2022a) manually confirmed that the distribution of CLINC OOD split is highly unrelated to CLINC train split.

Metrics: To evaluate IND performance, we measured the classification accuracy. And for OOD metrics, we adopt two metrics that are commonly used in recent OOD detection literature:

- **FPR@95.** The false-positive rate at the true-positive rate of 95% (FPR@95) measures the probability of classifying OOD input as IND input when the true-positive rate is 95%.
- **AUROC.** The area under the receiver operating characteristic curve (AUROC) is a threshold-free metric that indicates the ability of the model to discriminate outliers from IND samples.

¹CLINC150 dataset has an internal OOD split dataset to measure the OOD performance.

4.4.3 Competing Methods

Recent OOD detection methods can be divided into scoring function and model training methods. We compare LaCL with their combinations to investigate the effectiveness in a holistic view.

Scoring functions:

- **Mahalanobis distance** discerns abnormal input via class-wise density estimation assuming the representation follows the multivariate normal distributions (Lee et al., 2018b). It is a multi-dimensional generalization of quantifying *how many standard deviations away from the mean of the distribution*. We also cover the explicit ensemble of the Mahalanobis (Shen et al., 2021), which is a simple aggregation of the Mahalanobis distance (D) of intermediate representations:

$$D_{ens}(x) = D(f^{|L|}(x)) + \sum_{1 \leq l < |L|} D(\tanh(f^l(x))) \quad (4.7)$$

Notably, they place the nonlinear *tanh* layer to map the features of each transformer layer.

- **Cosine similarity** determines outliers by utilizing the similarity between the nearest neighbor of the known instance (usually from the training dataset) and the inferring input. Sec. 4.3.4 elaborates the details of the cosine similarity scoring function. We also cover an explicit ensemble version of the cosine scoring function, which determines OOD with an aggregation of cosine similarity of intermediate representations analogous to Eq. 4.7 but without *tanh* function in the last term.

Training methods: We set a cross-entropy loss trained model and a sigmoid based 1-vs-rest classifier (Shu et al., 2017) as a baseline model. Additionally, we compare our method with 6 recent CL based methods (Gao et al., 2021b; Liu et al., 2021a; Yan et al., 2021; Li et al., 2021; Zhang et al., 2022a; Zhou

BERT-base	ACC	IND : CLINC split (50%) → OOD : CLINC split (50%)							
		Cosine-single		Cosine-ENS		Mahalanobis-single		Mahalanobis-ENS	
		FPR@95 ↓	AUROC ↑	FPR@95 ↓	AUROC ↑	FPR@95 ↓	AUROC ↑	FPR@95 ↓	AUROC ↑
Baseline	96.74±0.36	38.97±2.88	92.10±0.53	37.18 ±0.46	92.39 ±0.27	39.33±1.31	91.74±0.22	40.14±0.33	91.05±0.39
DOC (Shu et al., 2017)	95.68±0.32	48.19±1.72	89.81±0.40	42.11 ±1.22	90.79 ±0.26	48.02±1.24	89.61±0.43	46.87±1.45	89.45±0.35
ConSERT (Yan et al., 2021)	97.42±0.26	35.68±1.31	93.36 ±0.45	31.56 ±1.42	93.08±0.26	34.35±1.66	93.34±0.51	34.40±1.01	92.23±0.45
SimCSE (Gao et al., 2021b)	96.79±0.26	40.65±0.64	92.11±0.45	36.05 ±1.16	92.32 ±0.07	39.70±0.48	92.27±0.45	38.92±1.62	91.02±0.26
MirrorBERT (Liu et al., 2021a)	97.60±0.30	34.22±0.92	93.75±0.28	30.49 ±1.49	93.38±0.22	33.86±1.63	93.82 ±0.29	33.92±1.17	92.77±0.31
Li et al. (2021)	97.31±0.20	36.10±1.59	93.02 ±0.44	32.84 ±1.79	92.82±0.33	35.14±1.46	92.98±0.51	36.38±1.34	91.64±0.62
Zhou et al. (2021)	96.56±0.24	36.10±2.43	93.15±0.53	39.22±1.01	92.46±1.11	35.62 ±3.34	93.21 ±0.39	40.34±1.39	92.02±0.94
Zeng et al. (2021)*	96.47±0.44	45.30±3.07	90.01 ±0.42	-	-	45.09 ±2.15	89.34±0.39	-	-
LaCL (ours)	98.04±0.11	26.59 ±1.27	94.93 ±0.15	30.81±1.62	93.77±0.21	28.03±1.15	94.49±0.53	37.40±1.51	92.07±0.26
BERT-base	ACC	IND : Banking split (50%) → OOD : Banking split (50%)							
		Cosine-single		Cosine-ENS		Mahalanobis-single		Mahalanobis-ENS	
		FPR@95 ↓	AUROC ↑	FPR@95 ↓	AUROC ↑	FPR@95 ↓	AUROC ↑	FPR@95 ↓	AUROC ↑
Baseline	94.61±0.74	55.56±1.80	88.64±0.31	52.52 ±2.78	90.57 ±0.29	56.22±2.19	88.62±0.18	60.49±4.75	87.16±0.70
DOC (Shu et al., 2017)	94.50±0.19	59.49±1.10	86.98±0.60	52.48 ±3.45	89.66 ±0.37	60.54±0.64	86.75±0.83	57.69±1.54	87.51±0.55
ConSERT (Yan et al., 2021)	94.91±0.20	50.09±4.34	90.18±0.35	46.33 ±2.02	91.30 ±0.10	53.08±3.61	89.75±0.92	58.82±1.03	88.23±0.37
SimCSE (Gao et al., 2021b)	94.83±0.47	54.23±2.03	90.27±0.76	46.01 ±1.90	91.34 ±0.32	52.78±2.64	90.08±0.82	59.72±2.99	87.88±0.79
MirrorBERT (Liu et al., 2021a)	95.29±0.47	48.55±0.81	90.81±0.26	43.67 ±0.79	91.58 ±0.20	48.70±2.25	90.53±0.21	55.75±1.64	88.59±0.06
Li et al. (2021)	95.42±0.36	46.33±2.48	91.33±0.22	42.91 ±1.19	91.95 ±0.23	45.68±2.66	91.12±0.15	57.48±1.16	89.02±0.07
Zhou et al. (2021)	93.82±0.69	52.86 ±4.07	89.43 ±0.18	55.28±2.11	88.94±1.47	55.15±2.04	88.86±0.21	58.67±2.39	87.60±1.32
Zeng et al. (2021)*	93.37±0.09	56.91 ±2.61	83.12 ±0.88	-	-	57.37±1.67	82.50±0.90	-	-
LaCL (ours)	95.51±0.27	35.71 ±0.61	92.86 ±0.16	42.67±1.67	91.58±0.11	47.73±7.12	89.88±1.21	69.00±0.60	83.66±0.94

* freezes the parameters of BERT, so we omitted ensemble evaluation.

Table 4.3: IND / OOD performance of each model 3 different settings on close-OOD setting. The best performance in each method is indicated in **bold** and the global best is underlined.

et al., 2021): Precisely, Gao et al. (2021b); Liu et al. (2021a); Yan et al. (2021) suggest a general CL framework, while Li et al. (2021); Zhang et al. (2022a); Zhou et al. (2021) introduce CL for OOD detection, which redesigns the loss function to maximize the discrepancy between IND and OOD.

For unsupervised CL methods, we train them with the cross-entropy loss additionally to give a signal about training distribution as in OOD specific frameworks. We extract the mean-pooled representation of the last layer features for all methods and pass it to a scoring function. On the other hand, LaCL exploits an implicit ensemble representation \mathbf{z} from GCL.

4.4.4 Main Results

This section reports the performance of LaCL with other competing methods in two different settings. Tab. 4.3 summarizes IND and OOD performance in close-OOD scenarios when the split ratio is 50% and Tab. 4.4 summarizes IND and OOD performance in three far-OOD scenarios. Performance report with

BERT-base	ACC \uparrow (IND)	IND : CLINC \rightarrow OOD : CLINC internal OOD							
		Cosine-single		Cosine-ENS		Mahalanobis-single		Mahalanobis-ENS	
		FPR@95 \downarrow	AUROC \uparrow	FPR@95 \downarrow	AUROC \uparrow	FPR@95 \downarrow	AUROC \uparrow	FPR@95 \downarrow	AUROC \uparrow
Baseline	95.62 \pm 0.39	12.13 \pm 1.63	97.50 \pm 0.14	10.20 \pm 0.53	97.84 \pm 0.04	12.67 \pm 0.83	97.32 \pm 0.22	11.23 \pm 0.40	97.61 \pm 0.09
Shen et al. (2021) [†]	96.66	-	-	-	-	10.88	97.43	10.12	97.77
DOC (Shu et al., 2017)	94.69 \pm 0.48	19.23 \pm 1.34	96.63 \pm 0.12	11.47 \pm 1.59	97.62 \pm 0.05	19.07 \pm 1.07	96.65 \pm 0.11	16.73 \pm 0.67	97.07 \pm 0.13
ConSERT (Yan et al., 2021)	96.21 \pm 0.09	11.00 \pm 0.72	97.61 \pm 0.10	8.37 \pm 0.15	98.00 \pm 0.05	10.33 \pm 0.76	97.67 \pm 0.12	9.60 \pm 0.46	97.78 \pm 0.13
SimCSE (Gao et al., 2021b)	95.80 \pm 0.30	12.50 \pm 0.10	97.60 \pm 0.14	8.80 \pm 0.20	97.96 \pm 0.02	11.63 \pm 0.21	97.65 \pm 0.10	10.50 \pm 0.50	97.67 \pm 0.03
MirrorBERT (Liu et al., 2021a)	96.50 \pm 0.15	10.33 \pm 0.61	97.78 \pm 0.04	8.40 \pm 0.17	98.09 \pm 0.06	9.60 \pm 0.17	97.82 \pm 0.03	8.67 \pm 0.40	97.94 \pm 0.04
Li et al. (2021)	96.08 \pm 0.19	11.03 \pm 0.65	97.68 \pm 0.12	8.90 \pm 0.36	98.01 \pm 0.16	10.27 \pm 0.55	97.70 \pm 0.11	9.33 \pm 0.64	97.81 \pm 0.14
Zhou et al. (2021)	95.28 \pm 0.27	10.93 \pm 0.74	97.65 \pm 0.15	9.60 \pm 0.28	97.67 \pm 0.18	10.43 \pm 0.90	97.66 \pm 0.18	10.80 \pm 0.71	97.51 \pm 0.37
Zeng et al. (2021)*	94.58 \pm 0.58	19.87 \pm 1.51	96.43 \pm 0.18	-	-	23.40 \pm 1.97	95.75 \pm 0.20	-	-
LaCL (ours)	96.96 \pm 0.39	6.67 \pm 0.51	98.27 \pm 0.16	7.43 \pm 0.06	98.15 \pm 0.07	8.30 \pm 0.61	98.00 \pm 0.17	12.33 \pm 0.12	97.31 \pm 0.11
IND : CLINC \rightarrow OOD : Banking									
Baseline	96.67 \pm 0.13	13.10 \pm 1.61	97.42 \pm 0.17	10.69 \pm 0.77	97.62 \pm 0.11	13.85 \pm 1.66	97.17 \pm 0.10	12.28 \pm 0.48	97.54 \pm 0.06
ConSERT (Yan et al., 2021)	96.74 \pm 0.33	10.71 \pm 2.58	97.83 \pm 0.36	9.96 \pm 1.73	97.64 \pm 0.18	9.53 \pm 2.44	98.02 \pm 0.33	9.59 \pm 2.35	97.78 \pm 0.26
SimCSE (Gao et al., 2021b)	96.55 \pm 0.07	12.19 \pm 2.14	97.69 \pm 0.38	10.01 \pm 2.17	97.66 \pm 0.33	10.94 \pm 2.50	97.79 \pm 0.42	10.80 \pm 2.69	97.61 \pm 0.41
MirrorBERT (Liu et al., 2021a)	97.00 \pm 0.10	9.29 \pm 1.19	98.04 \pm 0.28	9.63 \pm 1.37	97.76 \pm 0.18	8.47 \pm 1.42	98.14 \pm 0.27	9.75 \pm 0.88	97.90 \pm 0.23
DOC (Shu et al., 2017)	95.32 \pm 0.32	19.55 \pm 3.25	96.75 \pm 0.51	13.45 \pm 2.51	97.37 \pm 0.36	18.27 \pm 3.77	96.87 \pm 0.50	16.74 \pm 3.95	97.12 \pm 0.59
Li et al. (2021)	96.58 \pm 0.10	12.09 \pm 2.12	97.68 \pm 0.30	9.92 \pm 2.05	97.62 \pm 0.30	10.34 \pm 1.62	97.87 \pm 0.22	10.63 \pm 1.87	97.62 \pm 0.25
Zhou et al. (2021)	96.31 \pm 0.34	11.08 \pm 1.00	97.79 \pm 0.12	10.15 \pm 1.40	97.86 \pm 0.32	8.40 \pm 1.53	98.09 \pm 0.09	8.67 \pm 2.53	98.11 \pm 0.46
Zeng et al. (2021)*	95.20 \pm 0.25	22.39 \pm 4.09	95.87 \pm 0.43	-	-	23.06 \pm 3.32	95.64 \pm 0.38	-	-
LaCL (ours)	96.90 \pm 0.49	4.86 \pm 0.15	98.57 \pm 0.06	9.05 \pm 1.09	97.79 \pm 0.12	12.19 \pm 4.72	97.51 \pm 0.67	11.23 \pm 0.96	97.44 \pm 0.14
IND : CLINC \rightarrow OOD : Snips									
Baseline	95.83 \pm 0.08	27.10 \pm 1.44	96.11 \pm 0.03	11.54 \pm 1.08	97.65 \pm 0.16	20.33 \pm 0.98	96.68 \pm 0.19	11.54 \pm 0.33	97.82 \pm 0.12
DOC (Shu et al., 2017)	94.34 \pm 0.21	29.08 \pm 3.75	95.71 \pm 0.59	18.46 \pm 4.30	96.86 \pm 0.62	27.29 \pm 3.23	95.99 \pm 0.52	22.64 \pm 3.06	96.57 \pm 0.61
ConSERT (Yan et al., 2021)	96.14 \pm 0.24	18.20 \pm 1.87	97.08 \pm 0.25	11.10 \pm 0.87	98.00 \pm 0.15	15.93 \pm 1.81	97.42 \pm 0.26	12.97 \pm 1.15	97.92 \pm 0.18
SimCSE (Gao et al., 2021b)	95.80 \pm 0.28	20.99 \pm 3.85	96.84 \pm 0.51	10.81 \pm 2.64	97.94 \pm 0.34	15.20 \pm 4.34	97.46 \pm 0.34	10.59 \pm 2.77	98.04 \pm 0.22
MirrorBERT (Liu et al., 2021a)	96.57 \pm 0.21	18.17 \pm 2.64	97.14 \pm 0.37	10.07 \pm 1.34	98.04 \pm 0.31	14.25 \pm 2.76	97.56 \pm 0.38	10.81 \pm 1.89	98.10 \pm 0.24
Li et al. (2021)	96.06 \pm 0.24	16.96 \pm 2.53	97.24 \pm 0.17	9.85 \pm 0.94	98.09 \pm 0.22	13.19 \pm 2.02	97.62 \pm 0.22	10.26 \pm 1.77	98.11 \pm 0.19
Zhou et al. (2021)	95.09 \pm 0.43	19.20 \pm 3.20	96.87 \pm 0.39	19.01 \pm 3.85	97.00 \pm 0.45	13.59 \pm 3.47	97.57 \pm 0.34	14.39 \pm 2.85	97.59 \pm 0.31
Zeng et al. (2021)*	93.72 \pm 0.16	28.77 \pm 1.09	94.75 \pm 0.13	-	-	29.43 \pm 1.36	94.48 \pm 0.21	-	-
LaCL (ours)	96.62 \pm 0.45	8.06 \pm 1.59	98.24 \pm 0.17	8.17 \pm 0.63	98.40 \pm 0.08	11.06 \pm 0.92	98.02 \pm 0.11	11.36 \pm 0.64	98.11 \pm 0.05

* freezes the parameters of BERT, so we omitted ensemble evaluation.

[†] performance report from original paper.

Table 4.4: IND / OOD performance of each model 3 different settings on far-OOD setting. The best performance in each method is indicated in **bold** and the global best is underlined.

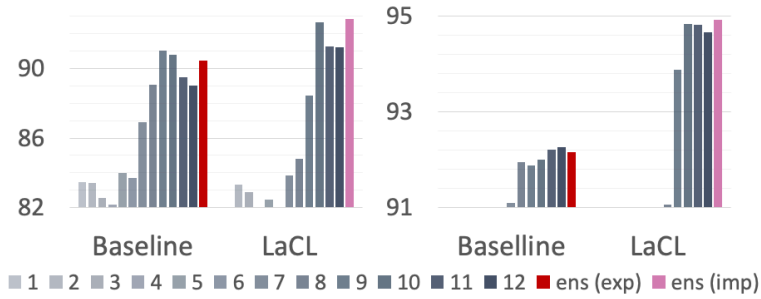
the remaining ratios, i.e., 25%, and 75%, are in Tab. 4.5. We report the average and standard deviation of 5 trials as a model performance for reproducibility.

From the results, we verified that LaCL with a cosine scoring (single) function consistently surpasses other methods significantly. We also confirmed that most methods (excluding LaCL) exhibit better performance with the explicit ensemble methods, indicating the potential of intermediate representations in OOD detection, as suggested in past studies (Shen et al., 2021; Sastry and Oore, 2020; Lee et al., 2018b). However, the performance of LaCL degrades with the explicit ensemble evaluations, proving that our ensemble method can gather more distinctive and calibrated information from entire layers than the naïve aggregation, and the explicit ensemble only acts as noise. It is also worth

BERT-base	ACC	IND : CLINC split (25%) → OOD : CLINC split (75%)							
		Cosine-single		Cosine-ENS		Mahalanobis-single		Mahalanobis-ENS	
		FPR@95 ↓	AUROC ↑	FPR@95 ↓	AUROC ↑	FPR@95 ↓	AUROC ↑	FPR@95 ↓	AUROC ↑
Baseline	98.11±0.05	27.59±0.20	94.49±0.03	25.65 ±0.49	94.46±0.02	26.76±0.47	94.62 ±0.08	27.36±1.18	93.83±0.01
DOC (Shu et al., 2017)	97.28±0.05	33.95±0.66	93.24±0.02	33.56±0.54	93.58 ±0.02	33.78±0.54	93.24±0.01	32.67 ±0.29	93.03±0.02
ConSERT (Yan et al., 2021)	99.00±0.14	23.22±0.06	95.01±0.06	23.57±0.70	94.73±0.04	22.79±0.80	95.23 ±0.06	22.28 ±1.30	94.50±0.10
SimCSE (Gao et al., 2021b)	98.36±0.05	25.43 ±1.07	94.51±0.18	25.98±0.73	94.63±0.04	25.65±0.33	94.78 ±0.13	26.39±0.40	94.22±0.06
MirrorBERT (Liu et al., 2021a)	99.11±0.05	22.39±0.52	95.49±0.34	22.99±1.38	95.10±0.19	21.57±0.75	95.64 ±0.33	21.41 ±1.25	95.00±0.27
Li et al. (2021)	98.80±0.48	25.58±0.28	94.75±0.05	24.97 ±0.69	94.69±0.05	25.93±0.56	94.99 ±0.10	26.08±1.17	94.38±0.15
Zhou et al. (2021)	97.53±1.35	25.55 ±1.13	95.03 ±0.11	26.15±2.01	94.57±0.35	26.98±1.85	94.86±0.21	27.22±1.05	94.05±0.16
Zeng et al. (2021)	98.61±0.24	28.57 ±2.13	94.01 ±0.60	-	-	29.48±2.41	93.72±0.50	-	-
LaCL (ours)	99.09±0.14	18.03 ±0.66	96.31 ±0.04	18.80±0.55	96.27±0.08	21.18±0.69	96.16±0.18	25.94±0.32	94.99±0.14
IND : CLINC split (75%) → OOD : CLINC split (25%)									
Baseline	96.52±0.42	39.58±1.36	91.42±0.04	46.00±0.77	91.45±0.18	36.33±1.54	91.63±0.09	35.33 ±0.18	93.24 ±0.01
DOC (Shu et al., 2017)	94.97±0.14	51.03±0.75	89.47±0.25	50.69±0.96	89.80±0.24	50.86±0.80	91.12±0.31	44.19 ±0.76	91.02 ±0.26
ConSERT (Yan et al., 2021)	96.70±0.21	38.44±1.00	92.35±0.22	37.67±1.04	92.13±0.16	38.97±1.17	91.31±0.35	35.42 ±0.92	92.88 ±0.15
SimCSE (Gao et al., 2021b)	96.09±0.29	40.20±1.57	91.97±0.41	40.05±1.80	91.85±0.45	37.97±0.67	91.00±0.39	35.53 ±0.99	92.85 ±0.23
MirrorBERT (Liu et al., 2021a)	96.85±0.26	36.89±0.24	93.18±0.00	38.00±0.87	93.03±0.01	35.39±1.03	92.17±0.33	32.69 ±0.68	93.38 ±0.19
Li et al. (2021)	96.76±0.10	41.06±0.10	92.70±0.07	41.00±0.72	92.52±0.07	38.58±0.84	91.76±0.10	34.61 ±1.54	93.04 ±0.07
Zhou et al. (2021)	95.72±0.62	45.53±5.89	90.70±0.94	43.75±1.80	91.22±0.05	41.91±0.63	91.48±0.26	41.75 ±0.68	91.94 ±0.40
Zeng et al. (2021)	95.65±0.36	45.64±1.98	90.76±0.42	44.61 ±1.48	91.62 ±0.37	-	-	-	-
LaCL (ours)	97.38±0.16	26.50 ±0.33	95.81 ±0.14	27.72±0.72	95.44±0.06	41.70±1.04	92.02±0.35	32.53±0.24	93.98±0.12
IND : Banking split (25%) → OOD : Banking split (75%)									
Baseline	97.54±0.15	34.96±1.61	93.29±0.14	34.84 ±1.53	93.83 ±0.11	35.70±1.21	93.22±0.17	38.45±2.39	92.28±0.20
DOC (Shu et al., 2017)	97.15±0.08	38.44±0.62	91.85±0.22	35.15 ±2.82	93.17 ±0.22	39.15±0.56	92.06±0.07	37.84±0.64	91.92±0.27
ConSERT (Yan et al., 2021)	97.72±0.20	26.81±0.95	94.43±0.30	29.81±2.03	94.12±0.12	26.37 ±2.20	94.48 ±0.19	38.92±1.15	92.40±0.03
SimCSE (Gao et al., 2021b)	97.24±0.60	28.29±1.83	94.98 ±0.12	30.07±2.41	94.25±0.17	27.64 ±1.72	94.87±0.17	45.23±1.52	91.87±0.15
MirrorBERT (Liu et al., 2021a)	97.72±0.42	27.11 ±1.36	94.90±0.16	31.11±2.44	94.11±0.11	27.13±0.93	95.05 ±0.13	42.54±0.60	92.36±0.05
Li et al. (2021)	97.59±0.33	27.72±1.24	95.15 ±0.14	27.40±2.30	94.56±0.10	26.61 ±0.07	95.15±0.09	38.16±1.89	92.99±0.06
Zhou et al. (2021)	95.96±2.10	32.05 ±3.26	92.80 ±0.49	34.34±0.84	92.66±0.70	32.97±1.89	92.65±0.87	36.13±0.68	92.38±0.40
Zeng et al. (2021)	95.09±2.92	44.71 ±6.22	89.93 ±2.38	-	-	47.28±9.02	89.24±3.02	-	-
LaCL (ours)	98.03±0.00	23.45 ±2.05	95.50 ±0.05	29.59±2.43	94.22±0.20	35.92±2.67	93.56±0.48	48.44±2.32	90.56±1.06
IND : Banking split (75%) → OOD : Banking split (25%)									
Baseline	92.69±0.58	42.63±3.54	91.63±0.13	41.32 ±2.79	92.48 ±0.39	41.78±1.21	91.62±0.18	49.74±0.37	90.04±0.04
DOC (Shu et al., 2017)	91.83±0.21	51.05±3.54	90.29±1.34	42.90 ±5.58	92.07 ±0.93	51.06±3.91	89.97±1.27	49.87±5.40	90.48±1.16
ConSERT (Yan et al., 2021)	92.78±0.34	42.64±0.37	92.31±0.16	40.27±0.74	92.73 ±0.12	39.41 ±0.47	92.46±0.10	46.98±1.86	90.69±0.30
SimCSE (Gao et al., 2021b)	92.70±0.33	43.55±1.30	92.39±0.02	42.11 ±1.12	92.62 ±0.05	43.23±0.28	92.29±0.18	47.17±2.14	90.36±0.18
MirrorBERT (Liu et al., 2021a)	93.69±0.33	42.11±0.56	92.12±0.40	40.00 ±0.75	92.33 ±0.27	42.37±0.93	92.01±0.32	47.96±0.83	90.26±0.42
Li et al. (2021)	93.69±0.46	41.32±3.34	92.53±0.23	40.92 ±2.60	92.63 ±0.29	41.05±2.23	92.47±0.41	46.12±2.33	90.78±0.56
Zhou et al. (2021)	92.14±0.46	43.36±0.65	92.28±0.06	41.58 ±2.05	92.42 ±0.08	45.53±0.74	91.94±0.23	50.00±5.59	90.26±1.23
Zeng et al. (2021)	92.10±0.33	48.16 ±3.01	88.04 ±0.78	-	-	52.89±3.89	86.69±0.77	-	-
LaCL (ours)	94.55±0.33	35.14 ±2.79	93.12 ±0.04	38.42±1.48	91.84±0.19	41.84±1.30	92.40±0.75	60.79±0.75	85.73±0.10

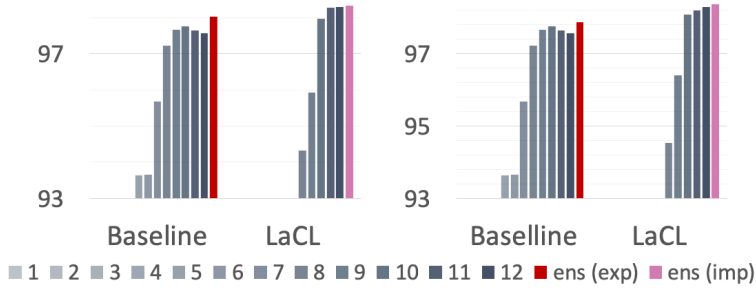
Table 4.5: IND / OOD performance of each model 3 different settings on Far-OOD setting. The best performance in each method is indicated in **bold** and the global best is underlined. LaCL outperforms other methods constantly in both IND and OOD metric.

noticing that LaCL shows good compatibility with cosine evaluation than the Mahalanobis evaluation since the Mahalanobis evaluation assumes that the extracted representations follow a Gaussian distribution. The following condition holds when the model is trained with cross-entropy loss, as they can be viewed as a generative classifier (Lee et al., 2018b). However, LaCL does not utilize cross-entropy loss, and the mentioned assumption is hardly met. Lastly, cosine ensemble evaluation tends to perform better than the Mahalanobis ensemble (Shen et al., 2021) counterpart in general. We conjecture that aggregating each



(a) Close-OOD.

Banking (ratio 50%), CLINC (ratio 50%)



(b) Far-OOD.

CLINC \rightarrow CLINC (OOD), CLINC \rightarrow Banking

Figure 4.3: Layer-wise AUROC score of baseline and our model with cosine scoring function. Explicit ensemble (baseline) tends to work well in relatively easy setting (far-OOD), while it yields worse performance than best performing single representation in harsh conditions (close-OOD). Implicit ensemble representation from LaCL outperforms other layers consistently.

result into a single one is more difficult in the Mahalanobis ensemble, as the Mahalanobis distance is not a normalized score (ranging $-\infty$ to ∞) while cosine is normalized (ranging -1 to 1). To conclude, we demonstrate that our model can extract elaborate ensemble representation, which yields the highest performance in various scenarios without multiple computations of the scoring function.

4.5 Analysis

In this section, we conduct supplementary experiments on LaCL to analyze our framework in-depth to elucidate its behavior.

4.5.1 Layer-wise Performance

Although our model outperforms other methods, it is unclear whether LaCL can well-assemble the information in the intermediate representations, analogous to our initial intuition. In an attempt to give answer this question, we scrutinize the layer-wise performance of LaCL and the baseline model. Fig. 4.3 summarizes the layer-wise AUROC score of LaCL and baseline in far-OOD and close-OOD settings. While higher layers tend to exhibit better performance, it is not always the case. Speaking otherwise, the last layer does not always guarantee the best performance among the upper layers. In this situation, the explicit ensemble of the baseline model conditionally shows performance gain. Namely, in a far-OOD setting (Fig. 4.3b), the ensemble representation displays substantial performance gain. In contrast, in a close-OOD setting (Fig. 4.3a), the ensemble representation often yields worse performance than the best-performing single layer. On the other hand, LaCL displays the best performances among other layers unconditionally, proving the capability of LaCL to absorb layer-specialized information of the entire layers properly.

4.5.2 Ablation study

We present ablations on LaCL to give intuition behind its behavior and justify our design choices.

Module ablations. We alter our model in several ways by removing some components in LaCL to test their independent impact. Tab. 4.6 summarizes component-wise ablations of our model in Banking 50% split setting, which is

Components	Acc \uparrow (IND)	Cosine	
		AUROC \uparrow	FPR@95 \downarrow
Baseline	95.07	88.53	55.39
+ SCL	95.65	91.68	39.55
+ GCL	95.72	92.23	37.62
+ CR (LaCL)	95.66	92.84	35.13
LaCL (variant 1)	95.72	92.51	37.34
LaCL (variant 2)	95.52	92.55	38.71

Table 4.6: Ablation study on LaCL components in Banking split setting .

the harshest condition (lowest performance) from Tab. 4.4, 4.3. While our layer agnostic training (GCL) or regularization term (CR loss) does not statistically contribute to the accuracy compared to applying SCL alone, they substantially improve OOD performance.

LaCL variants. From previous experiments (Sec. 4.5.1), we verified that the higher layers tend to yield better performance than the lower layers. So it is a reasonable conjecture that assembling only the upper layers may render better performance, assuming there is no meaningful information in the lower layers. Founded on this observation, we introduce two variants of LaCL: First variant (variant 1 in Tab. 4.6) utilize upper half layers \mathbf{z}^* in the *inference*:

$$\mathbf{z}^* = [\mathbf{c}^{(|L|/2)} \oplus \mathbf{c}^{(|L|/2+1)} \oplus \dots \oplus \mathbf{c}^{|L|}] \quad (4.8)$$

Furthermore, the second variant (variant 2) also utilizes the upper half layers \mathbf{z}^* ; however, they disconnect the lower half layers with GCL when they train the model. To our surprise, we verified that LaCL outperforms the other two variants, indicating that the features from lower layers retain considerable meaningful information, regardless of their performance.

Data Augmentation Selection. This section provides in-depth explanations about the various data augmentation methods along with their performance in OOD detection. We investigate the effectiveness of each augmentation method

Method	Instance 1	Instance 2	AUROC	FPR-95	ACC
Baseline	-	-	88.75	57.18	95.07
Shuffle	Raw + DO	Raw + DO + Shuffle	89.05	51.54	94.41
Dropout	Raw + DO	Raw + DO	90.35	49.23	95.2
Token-Cutoff	Raw + DO	Raw + DO + TC	90.55	50.71	95.2
BT	Raw + DO	BT + DO	91.1	48.27	95
RSM	Raw + DO	Raw + DO + RSM	90.91	48.85	95.59
RSM (pair)	Raw + DO + RSM	Raw + DO + RSM	91.75	43.89	95.26
RSM (pair) + BT	Raw + DO + RSM	BT + DO + RSM	91.94	42.89	95.26

Table 4.7: Data augmentaion results.

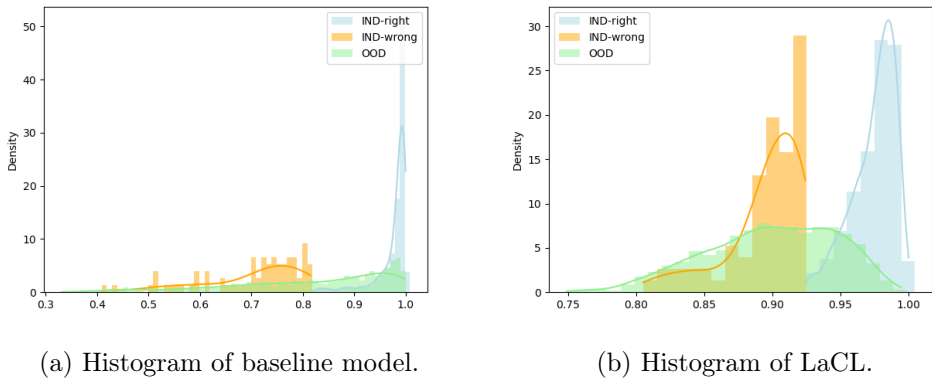


Figure 4.4: Histogram of LaCL and Baseline model trained on Banking split 50% setting.

in OOD detection to select our final data augmentation combination. Tab. 4.7 summarizes the results in Banking split 50% setting. For our final data augmentation, we greedily combined two best performing augmentations, i.e., BT and RSM, which showed best performance in OOD metrics.

4.5.3 Distribution Visualization

In this section, we plot a histogram of our model and baseline model to visualize how each model forms the IND and OOD distribution. Fig. 4.4 illustrates the histogram with the cosine scoring function of LaCL and the baseline model trained on Banking split 50% setting. We regard inputs as OOD when the input

Error Type	Input Text	GT	Prediction
Keyword over reliance	Check battery health on this device	OOD	jump_start
	Read text	OOD	text
	Who invented the internet	OOD	who_made_you
	Where can i find cheap rental skis nearby	OOD	car_rental
	Search up someone who plays in a movie	OOD	play_music
	What oil is best for chicken	OOD	oil_change_how
Mislabeled	What is harry’s real name	OOD	change_user_name
	Give me the weather forecast for today	OOD	weather
	I need you to order a new pair of eyeglasses for me	OOD	order
	Tell me something about linkin park	OOD	fun_fact
	What’s my current electric bill	OOD	bill_balance
	Order me a book of stamps and envelopes	OOD	order

Table 4.8: Examples of OOD test samples misclassified as IND. The keywords that cause over-reliance are in **bold**. GT stands for the ground-truth label.

Error Type	Input Text	GT	Prediction
Misspelling	I apprecaite the help from you	thank_you	OOD
	tell me how to spent frightened	spelling	OOD
	I appeci ate it	thank_you	OOD
	How much is alorie intake	calories	OOD
	Give me restaurant reccomendations	restaurant_suggestion	OOD
Nonstandard / Uncommon	10-4	yes	OOD
	Is it ok to use oil spray instead of canola oil	ingredient_substitution	OOD
	What’s your bday	how_old_are_you	OOD
	This charge is bs	report_fraud	OOD
	Ya	yes	OOD
Absence of keywords	Tell fred that i don’t have his guitar	text	OOD
	Did i stick to my dinner budget	spending_history	OOD
	Do i overspend when it comes to fast food	spending_history	OOD
	I want to tell susan that the meeting has been cancelled	text	OOD
	That’s all i need, i’m going now	goodbye	OOD

Table 4.9: Examples of IND test samples misclassified as OOD. Words related to their error type are highlighted in **bold**.

score is lower than the threshold δ , where δ is a preset threshold when TPR is at 95%, as stipulated in FPR-95%. To our surprise, both models have the ability to discriminate IND-wrong (yellow line) from IND-right answer (blue line), meaning can output high uncertainty for inputs that are likely to be wrong. On the other hand, LaCL forms a much clearer decision boundary and measures more precisely predictive uncertainty for OOD inputs (green line).

4.5.4 Case study

In this section we elaborate detailed case study on LaCL trained and tested on CLINC150 setting. Following previous experiment in the paper, we regard inputs as OOD when the the input cosine score is lower than the threshold δ , where δ is a preset threshold when TPR is at 95%, as stipulated in FPR-95%. To further investigate our error cases, we categorized the error cases into two classes: *OOD inputs, misclassified as IND*, and *IND inputs, misclassified as OOD*.

OOD inputs, misclassified as IND occurs when LaCL predicts a high confidence for OOD input. Example cases for this error are shown in Tab. 4.8. There are many controversial variation in this error in this case; however, they contain keywords or phrases that are highly relevant to the wrongly predicted IND intent, meaning they tend to learn some shortcuts (Geirhos et al., 2020) from the train set as mentioned in the paper. The fact that fine-tuned classifier learns some shortcuts from the train set is a well-known problem, and there are previous works (Moon et al., 2021). As a side note, few data were mislabeled, as can be seen in Tab. 4.8.

IND inputs, misclassified as OOD occurs when LaCL predicts a low confidence for IND input. Example cases for this error are summarized in Tab. 4.9. We sort the error cases into 3 groups: Misspelled words (typos), Nonstandard words (*e.g., acronyms, slangs*), and absence of keywords.

The examples in first error case occurs when the words heavily related to the intent are misspelled. Interesting part is that even though the model assigns low score to this type of errors, it predicts the true intent correctly. The second error type happens when nonstandard words appear. We believe that these errors are caused by the PLM not having semantics for those abnormal tokens.

Lastly, final error case arises when the intent-specific words are absent in the sentence. Namely, LaCL suffers when the input sentence comprises the words that are not commonly used, although their semantic is roughly the same. This phenomenon is another example of the prominence of keyword over reliance. However, learning shortcut is natural phenomenon surmising the following example: The train data in the '*text*' intent, the word *text* appears in 96 out of 100 sentences. As a side note, few data were mislabeled similar to previous error cases.

Summary of PART I

In this part of the dissertation, proposes two self-supervised learning based frameworks for two most common input types (i.e., images and sentences). In Chapter 3, we propose a novel training method called *masked contrastive learning* (MCL) and an inference method called *self-ensemble inference* (SEI). MCL can shape class-conditional clusters by inheriting advantages of CL and SEI fully leverages trained features from auxiliary self-supervised tasks in the inference phase. By combining our methods, our model reaches the new state-of-the-art performance in unsupervised image anomaly detection. And Chapter 4, we propose a novel framework called LaCL to improve OOD detection by leveraging intermediate representations. Our framework seeks a more calibrated output by combining layer-specialized representations from each layer via a layer-agnostic training scheme and novel regularization loss. Through extensive experiments and ablations, we have demonstrated the potential of intermediate representations in OOD detection and the effectiveness of our framework, which significantly outperforms other existing works.

While we confirmed the superiority of our frameworks among other compet-

ing methods in respective domains (i.e., CV and NLP), the proposed methodology can be used in more universal situations. For instance, MCL can substitute for contrastive learning when the data label is available regardless of the field, and LaCL can also be applied to every architecture that stacks identical neural modules hierarchically. In particular, CNN-based CV architectures (e.g., ResNet (He et al., 2016), Efficient-Net (Tan and Le, 2019)) are well known for capturing fine-grained features at the lower layers and coarse-grained fine-grained features at the higher ones, which have sufficient background reasons for the perfect compatibility of LaCL in CV. In future work, we plan to cross-apply our methods in other fields to investigate their potential and unify them to achieve better performance.

Part II: Exploiting Large Pre-trained Models for Anomaly Detection (Chapter 5 & 6)

Recently, we have witnessed a series of notable improvements from scaling the pre-trained models, where several cutting-edge models obtain intriguing extra functionalities in addition to the high performance on downstream tasks. Despite the aforementioned breakthrough, even very recent DAD studies (Cho et al., 2022; Shen et al., 2021) are still limited to relatively small pre-trained models, which naturally begs the question: *How do large-scale PLMs cope with outliers?* In this regard, **Part II** of this dissertation (**Chapter 5 & 6**) primarily focuses on exploiting the large-scale pre-trained model for DAD in natural language. To shed some light on our research objective, **Chapter 5** first analyzes the anomaly detection capability of the billion-scale pre-trained language models (PLMs) from various perspective and share some intriguing findings and their limitations. To address the limitations of the previous analyses, **Chapter 6** proposes a novel transferring method termed *prompt-augmented linear probing* (**PALP**) that can leverage large PLMs without any internal access or adaptation of PLMs. Before getting into the details, we introduce some recently proposed novel transfer learning methods frequently used in this part of

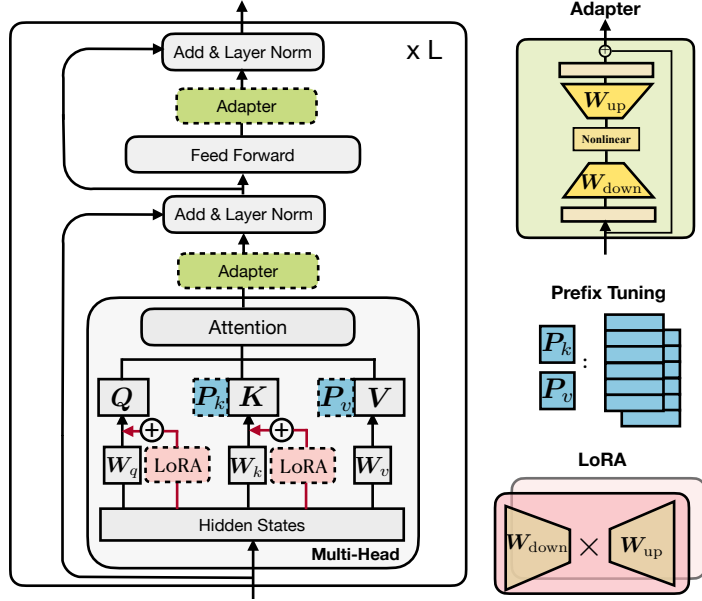


Figure 4.5: Illustration of various parameter efficient transfer learning methods. The image is from He et al. (2022)

the thesis as a preliminary.

Preliminaries.

Parameter-Efficient Transfer Learning

Parameter-Efficient Transfer Learning (PELT) is a cost-efficient transferring method that injects and updates a lightweight module between transformer layers while keeping the original network remain fixed. This section introduces several PETLs, i.e., Adapter, LoRA, and Prefix-tuning (Fig. 4.5 illustrates these methods):

Adapter inserts small trainable adapter modules between transformer layers while keeping most parameters of the original network frozen. The adapter module employs a bottleneck architecture that consists of 3 consecutive oper-

ations: down-projection, nonlinear activation, and up-projection. In this work, we attach adapter modules in two places, i.e., after the projection following multi-head attention and after the two feed-forward layers, following original implementation in (Houlsby et al., 2019). Also, we use relu as a nonlinear function, and layer normalization (Ba et al., 2016).

LoRA LoRA injects trainable low-rank matrices into transformer layers to approximate the weight updates. For a pre-trained weight matrix $W \in \mathbb{R}^{h \times k}$, LoRA decompose $\Delta W = W_{down}W_{up}$ where $W_{down} \in \mathbb{R}^{h \times r}$, $W_{up} \in \mathbb{R}^{r \times k}$ are trainable parameters. Specifically, we attached LoRA to query and key vector following the original implementation.

Prefix-Tuning Prefix tuning prepends l tunable prefix vectors to the keys and values of the multi-head attention at every layer. Following the original implementation, we reparametrize the prefix matrix of dimension h by a smaller matrix of dimension r composed with a large feed-forward neural network with \tanh as a nonlinear function.

In-Context Learning

In-context learning (ICL) is a brand new, training-free paradigm that attempts to make the most use of the nature of language models to conduct a target task. ICL promotes a language model to generate the desired output by guiding the model with a few examples of the target task (i.e., demonstrations) plus a set of templates tailored for the task. The figure 4.6 illustrates the underlying mechanism of ICL.

In detail, ICL consists of two steps (Liu et al., 2022b): First, the **input pre-processing** step combines the input of interest \mathbf{x} with k -shot samples (i.e., demonstrations) $(\mathbf{x}_i, y_i)_{i=1}^k$ from the training set. Then, a template function $f_{\text{template}}(\cdot)$ attach pre-defined descriptions to the input $f_{\text{template}}(\mathbf{x})$ or addition-

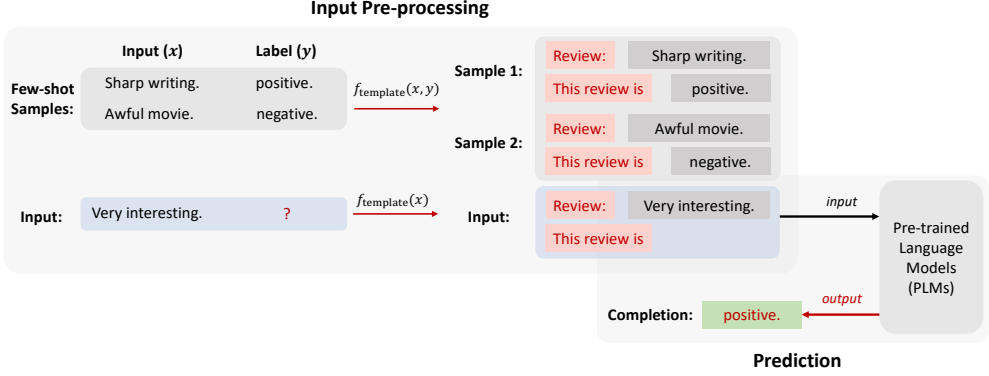


Figure 4.6: Illustration of ICL.

ally attach the corresponding natural language label to the *templified* input $f_{\text{template}}(\mathbf{x}, y)$, commonly referred to as a demonstration. (See Figure 4.6 for a graphical explanation.) For instance, the function attaches a prefix or postfix to the original \mathbf{x} , or it transforms y_i into the form of natural language rather than numeric numbers. In consequence, the final input for ICL, denoted as $\hat{\mathbf{x}}$, becomes a concatenation of all the pre-processed \mathbf{x} and $(\mathbf{x}_i, y_i)_{i=1}^k$:

$$\hat{\mathbf{x}} = D_1 \oplus D_2 \oplus \dots \oplus D_k \oplus f_{\text{template}}(\mathbf{x}), \quad (4.9)$$

where $D_i = f_{\text{template}}(\mathbf{x}_i, y_i)$ and \oplus refers to the concatenation operation.

Second, in the **prediction** phase, ICL leverages PLMs to compute the feature $\hat{\mathbf{h}} = e(\hat{\mathbf{x}})$, followed by a *verbalizer* $\mathcal{V} : \mathcal{H} \rightarrow \mathcal{Y}$ that is a reformulation of the language model head for task-specific adaptation. It is often assumed that the verbalizer only considers single tokens as its output candidates, which correspond to each item in the label space \mathcal{Y} .

Chapter 5

Investigating the Potential of Large Pre-trained Models as Anomaly Detectors

5.1 Introduction

Pre-trained language models (PLM), which are pre-trained on large-scale corpora using transformer-based architectures (Vaswani et al., 2017), have achieved groundbreaking success on sundry benchmarks (Wang et al., 2019b; Rajpurkar et al., 2016; Wang et al., 2019a), establishing themselves as the standard neural model in countless applications. Moreover, language models pre-trained with larger parameters on a rich volume of corpora tend to exhibit more intriguing potentials, such as the ability to capture world knowledge (Petroni et al., 2019), generate codes (Poesia et al., 2022), and even solve mathematical problems (Henighan et al., 2020), on top of understanding linguistic knowledge (e.g., semantic or syntactic). To explore the apex of pre-trained language models (PLMs), the size of PLMs is growing exponentially and has reached billions

to a trillion (Brown et al., 2020; Chowdhery et al., 2022; Fedus et al., 2022; Hoffmann et al., 2022).

Under these circumstances, the conventional method for transferring PLMs to a target task (i.e., fine-tuning) is now infeasible as it entails prohibitive costs to train and store the entire parameters of large PLMs for every desired task. To mitigate this issue, several recent parameter-efficient transfer learning (PETL) methods have been proposed to improve task scalability. For instance, adapter-based (Houlsby et al., 2019; Hu et al., 2022) approaches insert small neural modules into each layer of the PLM and update those lightweight modules in the training phase. Inspired by the recent success of textual prompts (Brown et al., 2020), prompt-based methods (Li and Liang, 2021; Lester et al., 2021; Shin et al., 2020) concatenate extra tunable tokens to the front of the input or hidden layers and update prepended soft prompts in the training phase.

Despite these breakthroughs in NLP, even very recent anomaly detection studies (Cho et al., 2022; Shen et al., 2021) are still limited to relatively small bi-directional PLMs (e.g., BERT, RoBERTa). Thus, *how large-scale PLMs or auto-regressive PLMs cope with outliers* is uncharted territory, naturally begging the following questions:

- **Q1:** Does increasing model size improve OOD detection performance without model parameters?
- **Q2:** If so, does scaling the size of PLM makes the model robust enough to utilize them without any additional process?
- **Q3:** Do fine-tuning and various PETL methodologies display differences in OOD detection performance according to the size of PLMs?
- **Q4:** Can the OOD detection methods from previous works (usually for the bi-directional PLMs) be transferred to auto-regressive PLMs (GPT)?

To resolve these questions, this paper investigates the capability of large

PLMs as outlier detectors from various perspectives. Specifically, we compare the robustness to outliers with various transfer learning techniques on several OOD benchmarks: Full fine-tuning, LoRA (Hu et al., 2022), Adapter (Houlsby et al., 2019), and prefix-tuning (Li and Liang, 2021) on various auto-regressive PLMs with different sizes, i.e., GPT2-S, M, L, XL (Radford et al., 2019), GPT-Neo (Black et al., 2021) and GPT-J (Wang and Komatsuzaki, 2021). From in-depth investigations, we share several intriguing observations: (1) As the size of the PLM increases, the performance improves without any update of model parameters. However, it is still challenging to use it without supervision since their performances still lag far behind compared to the fine-tuned small PLM (i.e., BERT-base). (2) PETLs outperform fine-tuning with sufficiently large PLMs in both IND and OOD metrics. (3) Lastly, leveraging the information of the last hidden representation, which is the most prevailing method for bi-directional PLM in recent OOD detection, does not transfer well in auto-regressive PLM, requiring a novel representation extracting technique. We believe that these findings will help future anomaly detection studies.

5.2 Probing OOD Robustness

5.2.1 Backbones and Models

To investigate the trend of OOD performance under varying scales of PLM, we consider three factors during backbone selection. They should be (1) publicly available, (2) reasonably large, and (3) share identical structures to eliminate factors other than size. Since recent large PLMs utilize auto-regressive objectives due to their computational complexity, we adopt six auto-regressive PLMs as the backbone of our experiments accordingly: **GPT2 (S,M,L,XL)**, **GPT-Neo**, and **GPT-J**.

For the parameter-efficient transfer methods, we selected two methods: two adapter-based and one prompt engineering-based. Namely, **Adapter** (Houlsby et al., 2019), **LoRA** (Hu et al., 2022), and **Prefix-tuning** (Li and Liang, 2021) are selected for the adapter approach, which is compatible with classification tasks, for the prompt approach. We also report the performance of linear evaluation, i.e., single layer perceptron (SLP) on top of PLMs, and fine-tuning, which act like a lower-bound and upper-bound, respectively.

5.2.2 Dataset and Metrics

Dataset. We evaluate our model on two datasets, CLINC150 and Banking77, widely used in OOD detection. CLINC150 dataset (Larson et al., 2019) contains 150 class labels (15 intents for 10 domains), while Banking77 dataset (Casanueva et al., 2020) consists of fine-grained 77 bank-related intents. Following the experimental settings from previous works (Cho et al., 2022; Zhang et al., 2022a; Shu et al., 2017; Fei and Liu, 2016; Lin and Xu, 2019), we validate our models in two different scenarios: far-OOD setting and close-OOD setting. For CLINC dataset, we train our model with the whole training dataset and test with an independent OOD test split from CLINC dataset, which does not overlap with 150 classes in the training dataset. Outliers in CLINC OOD split are distributionally far from the training distribution (Zhang et al., 2022a), so it is relatively easy to discern. For Banking77, we partition the dataset into 2 disjoint datasets (i.e., IND / OOD dataset) based on the class label. Since both IND and OOD datasets originated from the equivalent dataset, they share similar distributions and properties, making the task more demanding. Thus, we refer to a CLINC OOD setting as far-OOD and split settings in Banking as close-OOD settings, respectively.

Metrics. To evaluate IND performance, we measured the classification accu-

racy. And for OOD performance, we adopt two metrics commonly used in recent OOD detection literature:

- **FPR@95.** The false-positive rate at the true-positive rate of 95% (FPR@95) measures the probability of classifying OOD input as IND input when the true-positive rate is 95%.
- **AUROC.** The area under the receiver operating characteristic curve (AUROC) is a threshold-free metric that indicates the ability of the model to discriminate outliers from IND samples.

5.2.3 OOD Evaluation Methods

Evaluation in OOD detection is done via a scoring function, which outputs the appropriateness of the input into a single scalar value (p). Then we compare p with the pre-set threshold δ to determine whether the input is an outlier or not:

$$I_{\delta}(\mathbf{x}) = \begin{cases} \text{IND} & p(\mathbf{x}) \geq \delta \\ \text{OOD} & p(\mathbf{x}) < \delta, \end{cases} \quad (5.1)$$

In this paper, we evaluate the performance of our method in 4 different evaluation methods, which can be categorized into 2 higher branches: representation-based and logit-based.

Logit-based approaches exploit the PLM’s prediction result extracted from the classification layer as their primary information to discern outliers. Logit-based approaches are simple and have their own dominance in computational cost since it pursues OOD detection and general classification nigh simultaneously.

- **MSP** is a baseline method in this branch that employs the maximum softmax probability to score the appropriateness of the given input, based on the idea that the model will output more certain output (higher probability) to a normal

sample (Hendrycks and Gimpel, 2017):

$$p(\mathbf{x}) = \frac{e^{f_i(\mathbf{x})}}{\sum_{j=1}^N e^{f_j(\mathbf{x})}}, \quad (5.2)$$

where $f_i(\mathbf{x})$ refer to as max value from the classification layer (max logit value).

• **Energy** is a variant of MSP, which calibrates logit value based on energy function (Liu et al., 2020):

$$p(\mathbf{x}) = -E(\mathbf{x}; f) = T \cdot \log \sum_i^N e^{f(\mathbf{x})/T}. \quad (5.3)$$

Representation-based approaches, on the other hand, employ the hidden representation from PLM as their primary source. Since the size of the hidden representation is larger and inheres more copious information, they generally yield a more precise decision than logit-based approaches. However, they require more inference time to derive a final score. We employed Mahalanobis distance-based and cosine similarity-based methods in this branch.

• **Mahalanobis distance** refers to the distance between the specific distribution and the input. In OOD detection, we estimate the gaussian distribution of the training dataset and utilize the minimum Mahalanobis distance to score the input suitability (Lee et al., 2018b):

$$p(\mathbf{x}) = (\mathbf{h} - \boldsymbol{\mu}_k)^\top \Sigma^{-1} (\mathbf{h} - \boldsymbol{\mu}_k), \quad (5.4)$$

where training distribution is $(\mathcal{N}(\boldsymbol{\mu}_i, \Sigma))$ for $i \in i = \{1, 2, \dots, |C|\}$, and k refers to a index of minimum mahalanobis distance.

• **Cosine Similarity** method utilizes the cosine distance between the representation of the given input ($z(\mathbf{x})$) and the nearest neighbor $z(\mathbf{x}_{nn})$ (Tack et al., 2020):

$$p(\mathbf{x}) = \text{sim}(z(\mathbf{x}), z(\mathbf{x}_{nn})) \quad (5.5)$$

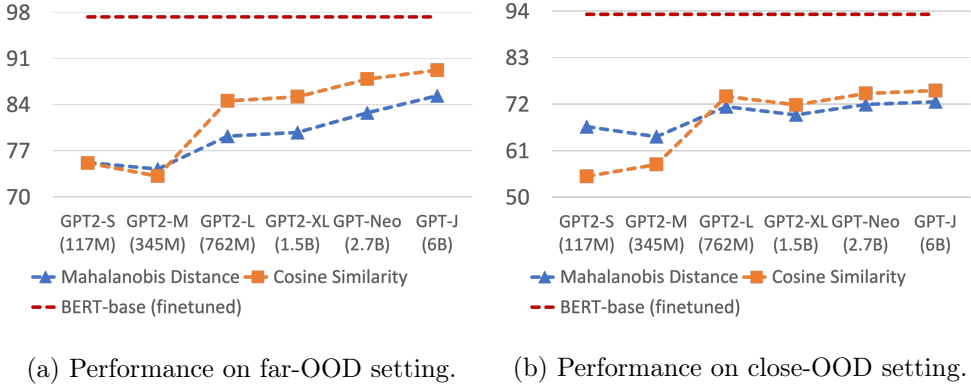


Figure 5.1: OOD detection performance of PLMs without updating the model parameters.

5.3 Analysis

In this section, we share several intriguing findings and insights from various settings.

5.3.1 OOD Robustness of PLMs without Supervision.

In this experiment, we investigate the OOD detection capability of PLMs without parameter tuning. Precisely, we extract the final layer representation from each frozen PLM and evaluate their performance via representation-based evaluation methods. (Logit-based evaluation methods are not used as they require additional training of the classification layer.) Figure 5.1 summarizes the results in two scenarios (i.e., far-OOD and close-OOD). We verified the correlation between the size of PLMs and their OOD detection ability, but utilizing them without parameter supervision is roughly impossible since they still lag far behind the small supervised methods (i.e., BERT-base with Mahalanobis evaluation) in a barebone setting. Moreover, performance improvement from

Setting	Backbone	Evaluation Method			
		MSP	Energy	Mahal.	Cosine
CLINC Setting	GPT2-S	93.22	95.79	77.63	76.34
	GPT2-M	95.41	97.63	82.42	79.82
	GPT2-L	96.21	97.77	96.93	97.57
	GPT2-XL	96.48	97.99	97.28	97.66
	GPT-Neo	96.04	97.72	96.59	97.64
	GPT-J	97.34	98.50	97.91	98.20
Banking Split 25%	GPT2-S	90.12	91.32	75.32	73.11
	GPT2-M	91.74	92.78	78.03	76.56
	GPT2-L	93.02	93.45	92.44	93.41
	GPT2-XL	94.29	94.95	93.24	94.10
	GPT-Neo	93.83	94.85	92.79	93.88
	GPT-J	94.11	95.10	93.66	94.80

Table 5.1: AUROC of each PLMs trained with LoRA adapter. Energy function outperforms other evaluation methods consistently.

the scaling saturates in a more harsh setting (i.e., close-OOD), displaying an unbridgeable gap with the fine-tuned model.

5.3.2 Evaluation methods for auto-regressive PLMs.

Many recent OOD works (Zhou et al., 2021; Shen et al., 2021) leverage hidden representation-based evaluation, as they generally surpass logit-based evaluations (Podolskiy et al., 2021). The reasonable conjecture behind their success is that hidden representations have more copious information than the logit value. However, in auto-regressive PLMs, logit-based evaluations (i.e., MSP and Energy) outperform representation-based methods (i.e., Mahalanobis distance and cosine similarity), as shown in Table 5.1. The reasonable conjecture

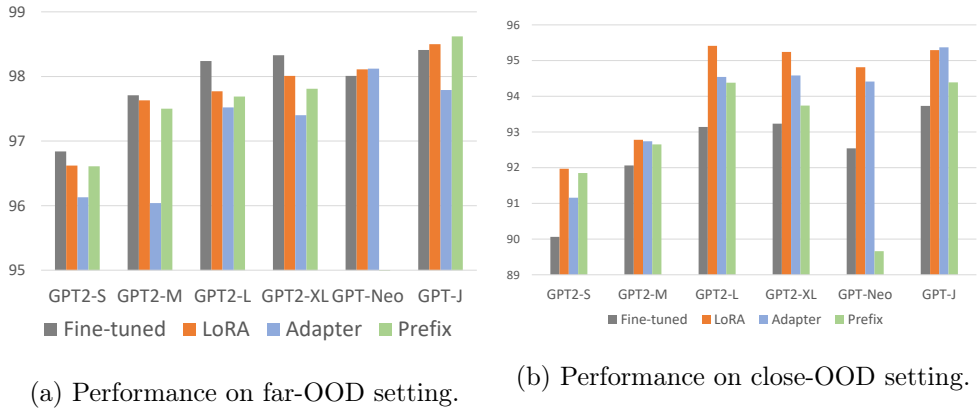


Figure 5.2: OOD detection performance of PLMs without updating the model parameters.

for this phenomenon is due to the characteristic of the language model. Unlike bi-directional models (e.g., BERT, RoBERTa, DeBERTa), decoder models (e.g., GPT and its variants) do not have [CLS] embedding, which assembles the token embeddings to capture holistic information (Devlin et al., 2019; Kim et al., 2021). Therefore, auto-regressive PLMs generally utilize the last token embedding as a final feature embedding replacing [CLS] embedding of encoder-based models. While the last token of GPT is befitted for predicting the next token, however, it cannot extract the holistic semantics of the sentence suitably, unlike [CLS] embedding. We believe extracting a better representation through various pooling (Wang and Kuo, 2020) methods might be a possible avenue for auto-regressive models to improve the OOD robustness further.

5.3.3 PETLs VS. Fine-tuning

In this experiment, we investigate the performance gap between various PETL methods (i.e., Adapter, LoRA, prefix-tuning) and model fine-tuning. To compare the performance of each method under similar circumstances, we set every

Setting	Method	# Params.	Backbone					
			GPT2 (S)	GPT2 (M)	GPT2 (L)	GPT2 (XL)	GPT Neo	GPT-J
CLINC (far-ood)	Linear (SLP)	0%	83.03	87.39	88.47	89.55	89.44	91.94
	Fine-tuning	100%	96.84	97.71	98.24	98.33	98.01	98.41
	LoRA	0.1%	95.00	96.54	97.66	97.72	98.14	97.79
		0.5%	96.41	96.04	97.52	97.45	98.12	97.89
		1%	96.13	95.89	97.61	97.40	98.11	98.50
	Adapter	0.1%	96.62	97.52	97.74	97.71	97.81	96.80
		0.5%	95.64	97.07	97.86	96.94	97.98	98.37
		1%	95.79	97.63	97.77	97.99	98.12	98.50
	Prefix	0.1%	95.53	96.93	96.38	97.88	90.25	98.55
		0.5%	96.91	96.96	97.78	97.88	89.81	97.92
		1%	96.97	97.50	97.69	97.81	88.98	98.62
Banking split 25% (close-ood)	Linear (SLP)	0%	72.97	75.17	80.46	77.59	86.55	89.12
	Fine-tuning	100%	90.06	92.06	93.14	93.23	92.54	93.73
	LoRA	0.1%	91.18	91.74	94.65	94.58	94.29	95.82
		0.5%	91.16	92.98	94.54	94.04	94.55	94.65
		1%	91.39	92.39	93.45	93.59	94.81	95.29
	Adapter	0.1%	91.97	93.24	94.90	94.69	93.26	95.59
		0.5%	92.90	92.63	95.18	95.24	93.61	95.83
		1%	91.32	92.78	95.41	94.95	94.41	95.37
	Prefix	0.1%	91.22	91.92	93.96	93.48	81.9	94.93
		0.5%	91.85	92.55	93.84	93.34	80.82	93.99
		1%	92.09	92.65	94.38	93.74	89.66	94.39

Table 5.2: AUROC of various PETL methods with various number of parameters. Performance was achieved by the energy function.

PETL method to utilize a similar number of parameters sufficient enough to reach maximum accuracy. Moreover, we utilized the energy function to evaluate each method as they displayed the best performance among other evaluation methods, i.e., cosine, Mahalanobis, and MSP, in the previous experiments. Table 5.2 summarizes the results.

From this experiment, we observed that PETL methods are more robust than fine-tuning with reasonably large PLMs (i.e., GPT-J). Specifically, most PELT methods on GPT-J outperform fine-tuning with proper tunable parameters. Nevertheless, size is not the ultimate answer. While it is clear that the scale of a model is an essential factor in OOD robustness, larger models are still vulnerable to close-OOD inputs. The capability to detect far-OOD inputs (far from the training distribution) improves proportionally as the size grows, while the ability to identify close-OOD input improves rather trivially. PLM’s vulnerability to close-OOD has already been reported in other studies (Zhang et al., 2022a), and this may be related to shortcut learning (Geirhos et al., 2020) that predicts with high probability by looking at specific words. Generating OOD data with particular keywords or utilizing another pretext task, such as (Moon et al., 2021), can be worthy approaches to alleviate such phenomena. A suitable OOD approach is necessary to alleviate the aforementioned issue, as it can further boost the robustness. We conduct additional experiments with PETLs on three different numbers of tunable parameters: 0.1%, 0.5%, and 1% of the PLM parameters. Figure 5.2 summarizes the results. With sufficient parameters to reach maximum performance, there is no meaningful difference or improvement within each methodology. Also, empirically, we confirmed that LoRA is the most stable during learning and that prefix-tuning fluctuates severely according to learning.

5.4 Related Work

Parameter-Efficient Transfer Learning is drawing considerable attention lately, emerging as an alternative strategy to fine-tuning. Compared to fine-tuning, parameter-efficient transfer methods show superiority in the number

of trainable parameter usage while achieving performance analogous to fine-tuning. Depending on the characteristics of the methods, parameter-efficient transfer methods can be categorized into *Adapter-based* and *Prompt-Engineering* approaches.

Adapter (Houlsby et al., 2019; Pfeiffer et al., 2021) refers to a lightweight neural module injected within each layer of PLM. The structure of the adapter generally consists of a bottleneck layer (down-projection and up-projection), a nonlinear function, a normalization layer, and a residual connection. The adapter has many different variants due to numerous design choices, such as the order or specifics of each component (e.g., which normalization technique will be used) and where the adapter will be attached. For example, LoRA (Hu et al., 2022) inserts low-rank decomposition matrices in each weight in self-attention (Vaswani et al., 2017) (i.e., query, key, and value).

Another line of work, *prompt engineering*, casts the existing task as a text generation problem to fully leverage the capability of PLMs to predict the appropriate word in the given sentence. This approach requires an empirical endeavor of optimizing the prompt to maximize a PLM’s performance. Earlier works exploit handcrafted manual prompts (Schick and Schütze, 2021a; Jiang et al., 2020) or by providing demonstrations to PLM¹ (Brown et al., 2020; Raffel et al., 2020; Gao et al., 2021a; Zhao et al., 2021). More recent work replaces the manual prompt with a soft prompt (Li and Liang, 2021; Lester et al., 2021; Shin et al., 2020; Liu et al., 2021b), a machine trainable continuous vector. The soft prompt is a more modular and versatile method that evades additional latency in the inference phase because it detaches the additionally trained parameters and solely employs the final output of the trained parameters as the prompt.

While former parameter-efficient transfer methods showed noticeable achieve-

¹also termed as in-context learning.

ments, their evaluations generally assume the train and test distributions are identical (i.e., i.i.d. assumption); however, this condition is rarely satisfied in real-world scenarios due to the diversity and volatility of user input. Consequently, if the model can not correctly handle distribution-shifted malicious input and misconceives it as an in-distribution (IND) example, it may lead to fatal accidents.

Despite its practical importance, how large PLMs or parameter-efficient transfer learning cope with unknown input is poorly understood. This work aims to understand language models’ capabilities to detect outliers through parameter-efficient transfer learning methods.

Chapter 6

Enriching the Anomaly Detection Ability of Large Pre-trained Models via Prompting

6.1 Introduction

Since the emergence of Transformer-based (Vaswani et al., 2017) language models, we have witnessed notable improvements in the natural language processing literature, even attaining human-level performance on several benchmarks. In addition, as it becomes evident that scaling laws work for such models (Kaplan et al., 2020), there has been a significant amount of investment in the field to enhance them in terms of the number of their parameters—from millions to billions—and the volume of the data they consume during training (Brown et al., 2020; Chowdhery et al., 2022; Fedus et al., 2022; Hoffmann et al., 2022). As a result, some cutting-edge language models become possible to obtain intriguing extra functionalities, such as the ability to capture world knowledge (Petrone et al., 2019), generate codes (Poesia et al., 2022), or solve mathematical

problems (Henighan et al., 2020), in addition to being proficient in recognizing linguistic patterns. These anecdotes, which demonstrate the general power of large language models, naturally raise researchers’ expectation that language models can act as a universal, off-the-shelf solution for a range of downstream tasks while minimizing the cost required for adapting them to a specific job at the same time.

However, there is no free lunch; the effectiveness and generalizability of large language models achieved by scaling come at the cost of physically serving such gigantic neural architectures. Thus, the institutions that distribute large models such as GPT-3 (Brown et al., 2020) usually pursue the strategy of providing commercial APIs which only allow limited access to the models. In other words, it is often the case that users cannot receive the information about the inner workings of the models, such as gradients concerning the models’ parameters which are crucial for fine-tuning the models for a particular purpose. Therefore, there has been a growing interest of adapting language models in this restricted setting, dubbed as *black-box tuning* (Sun et al., 2022; Diao et al., 2022).

In this scenario, especially focusing on classification, the two mainstream paradigms are linear models and in-context learning (ICL). The former derives the final representation from a language model and trains a simple classifier, e.g., Support Vector Machine (SVM), Single-Layer Perceptron (SLP), and k Nearest Neighbors (k -NN), on top of the extracted features. On the other hand, ICL is a training-free mechanism that makes the most use of the nature of language models. Specifically, ICL promotes a language model to generate the desired output by guiding the model with a few examples of the target task (i.e., demonstrations) plus a set of templates tailored for the task, where many works have suggested that exploiting such a mechanism improved various NLP tasks (Shwartz et al., 2020; Ben-David et al., 2022).

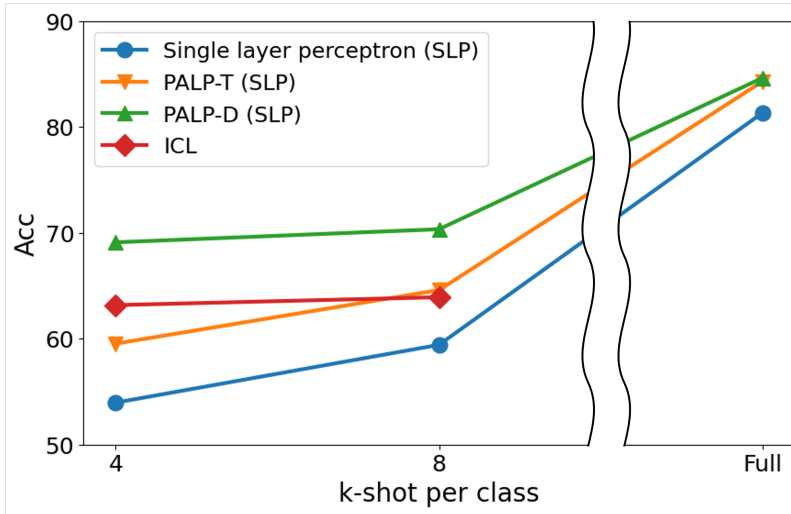


Figure 6.1: Average accuracy (12 classification tasks) of various *black-box* transferring methods trained on GPT-J. ICL can not leverage the full train dataset due to the length limit, and their performance saturates quickly. Our method (PALP) is scalable with available training samples and minimizes the performance gap between ICL in a few-shot setting. The performance of the respective task is summarized in Table 6.3, 6.4.

Yet, it is worth noting that according to the quantity of data instances available for training, the two aforementioned approaches have clear strengths and shortcomings. Namely, ICL shows a remarkable generalization ability in a few-shot setting. However, the ICL performance does not scale well with the number of available training samples (See Figure 6.1 for details). This is because ICL relies on the innate capability of language models, i.e., being able to predict the next word given context, and a verbalizer, which is a function that maps the probabilities of some tokens to the probability of each label rather than constructing a task-specific decision maker. By doing so, ICL is equipped with

the invaluable benefit of being free from explicit fine-tuning, but it is also a double-edged sword that induces a scalability issue. In particular, an explicit upper bound exists on the performance of ICL, which is determined by the maximum length of the utilized language model.

In this paper, we show that the combination of linear classifiers and the techniques introduced for ICL can cover the weaknesses of each method. We train diverse linear classifiers whose representations are extracted from input pre-processing strategies invented for facilitating ICL. Specifically, we augment training data instances with the templates or prepend additional demonstrations in front of the input of interest for better contextualization.

We validate our method with various datasets, demonstrating that it is consistently superior to baselines in both the low-data and full-data settings. From empirical experiments, we observe that exploiting templates that provide hints about the target task or concatenating demonstrations can significantly enhance the extracted representations from PLM, improving the classifiers’ performance in various scenarios and reducing the gap between ICL and fine-tuning. Intriguingly, we also discover that importing the techniques directly from ICL without care may cause the inheritance of the disadvantages of ICL, such as a substantial performance variance depending on the appended demonstrations or high sensitivity to the format of templates.

6.2 Preliminary

6.2.1 Problem Formulation

The classification task is to repeatedly make a sound forecast in new situations on the basis of currently available information (Michie et al., 1994). In this work, we consider classifying input sequences based on *black-box tuning* (Sun et al.,

2022; Diao et al., 2022), where the parameters of pre-trained language models (PLMs) are inaccessible. That is, PLMs only serve as an encoder function e that delivers n -dimensional latent features \mathbf{h} from input \mathbf{x} . Formally, for input $\mathbf{x} \in \mathcal{X}$, let the n -dimensional continuous latent features extracted from PLMs be $\mathbf{h} = e(\mathbf{x})$, where $\mathbf{h} \in \mathcal{H}$. In addition, let $\mathcal{Y} = \{0, 1, \dots, |C|\}$ be a label space, where $|C|$ is the cardinality of the space. Then, a classifier $p(y|\mathbf{h}; \boldsymbol{\theta}) : \mathcal{H} \rightarrow \mathcal{Y}$ maps \mathbf{h} to a class label $y \in \mathcal{Y}$, estimating the probability of input \mathbf{x} belonging to a certain label.

6.2.2 Linear Classifiers

Linear models, such as single layer perceptron (SLP), SVM, or logistic regression, are trained by solving optimization problems concerning their parameters. First-order methods such as gradient descent shown below are a general choice for parameter estimation:

$$\theta_{t+1} = \theta_t - \eta \nabla \mathcal{L}(\theta(t), \mathcal{H}_{batch}), \quad (6.1)$$

where, \mathcal{L} , η , \mathcal{H}_{batch} refer to a loss function (e.g., the cross-entropy loss, hinge loss in SVM, and MSE loss in regression), learning rate, and a mini-batch sampled from the training dataset. In this paper, we evaluate 5 different linear-probing classification methods: k -NN, logistic regression (LR), SVM, gaussian discriminative analysis (GDA), and SLP.

- **k -Nearest Neighbors (k -NN)** : the k -NN is a non-parametric supervised learning method that outputs a class membership by assigning the most common class label among its k nearest neighbors. Specifically, we set k to 3 and leverage euclidean distance to select nearest samples.
- **Support Vector Machine (SVM)** : The goal of SVM is to find the

hyperplane that separates the group of data points x_i with *maximum-margin*:

$$w^T h - b = 0,$$

where w is the normal vector to the hyperplane. We trained SVM with linear kernel (linear) and used squared hinge loss.

- **Logistic Regression (LR)** : LR is a statistical model that models the probability of each class by having the combination of log-odds from independent features:

$$p(y|h) = Ber(y|\sigma(w^T h + b)),$$

where σ is the sigmoid activation, w and b refers to weights and bias. Then probability of input h belonging to class $y = 1$ can be written as follows:

$$p(y = 1|h) = \sigma(a),$$

where Ber is bernoulli distribution, $a = w^T h + b$ usually termed as *logit* which is combination of log odds from variables.

- **Single layer perceptron (SLP)** : SLP is a single-layer version of neural network, which we commonly use in machine learning literature:

$$p(y|x) = softmax(w\sigma(h) + b),$$

where σ refers to a activation function. To train SLP, we used ReLU activation, cross entropy loss, Adam optimizer with learning rate 15e-5. For the training batch size, we set it to 2 for few-shot setting, and 16 for full-training dataset.

- **Gaussian discriminative analysis (GDA)** : GDA is a generative classification methods, unlike other probing methods mentioned above. Generative classifiers classify the given input via Bayes rule, while discriminative classifier directly models the class posterior:

$$p(y = c|\mathbf{h}; \boldsymbol{\theta}) = \frac{p(\mathbf{h}|y = c; \boldsymbol{\theta})p(y = c; \boldsymbol{\theta})}{\sum_{c^* \in |C|} p(\mathbf{h}|y = c^*; \boldsymbol{\theta})p(y = c^*; \boldsymbol{\theta})}. \quad (6.2)$$

Formally, they estimate posterior from the class likelihood $p(\mathbf{h}|y = c; \boldsymbol{\theta})$ and prior distribution $p(y = c; \boldsymbol{\theta})$. Then GDA compute the most probable class label as follows :

$$\begin{aligned} \hat{y}(h) &= \underset{i^*}{\operatorname{argmax}} \log p(y = i^*|h). \\ &= \underset{i^*}{\operatorname{argmin}} (h - \mu_c)^T \Sigma^{-1} (h - \mu_c) \end{aligned} \quad (6.3)$$

Meaning GDA classifies new input with the label i^* which has closest Mahalanobis distance (i.e., $(h - \mu_c)^T \Sigma^{-1} (h - \mu_c)$). We estimate tied (shared) covariance Σ of GDA with Ledoit-wolf shrinkage method (Ledoit and Wolf, 2004).

6.3 Prompt-Augmented Linear Probing

6.3.1 Motivation

The primary intuition behind our method borrows from the in-context learning ability exhibited by language models. Specifically, in-context learners benefit from more elaborate and longer prompts (Reynolds and McDonell, 2021), allowing them to carry out deeper reasoning through longer input sequences and corresponding hidden layers.

We posit that providing appropriate guidance to the language model via prompts (input pre-processing step in ICL) benefits not only the usual causal

language modeling (i.e., predicting the next token) ability but also enhances the quality of the representation for the input text. The primary goal of our method is to extract a more distinctive representation from PLMs via crafting a raw dataset into a more understandable form and training linear probers on the top of the extracted representations. Specifically, we transform the dataset in two ways:

1. We utilize a simple template that gives a brief description of input and the objective of the task.
2. On top of *templified* dataset, we concatenate a single class-wise demonstration to an inferring input to give important cues (i.e., input-label mapping, label space, or distribution of the input text) (Min et al., 2022b; Kim et al., 2022) regarding the target task.

In the following subsection, we explain the dataset reconstruction strategies for each method.

6.3.2 Template-based Training Samples (PALP-T)

Applying a template to the input is the most straightforward and intuitive way to enforce PLM to follow user requirements. Accordingly, we attempt to extract more task-specific features by attaching a fixed prefix, infix (for sentence pair tasks), or postfix, which describes a raw input into a more understandable form.

For instance, we transform sentiment analysis instance ‘*very interesting.*’ into ‘**Review:** *very interesting.* **Sentiment:**’ to provide the language model additional indications that the input is the form of review, and the user expects sentiment as an answer. Formally, for the training dataset $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, y_i) | i \in m\}$, we convert this into $\mathcal{D}_{\text{train}}^{\text{template}} = \{(f_{\text{template}}(\mathbf{x}_i), y_i) | i \in m\}$, where $f_{\text{template}}(\cdot)$

is a template function. Then we train linear classifiers with transformed dataset $\mathcal{D}_{\text{train}}^{\text{template}}$ in the exact same way as in Eq. 6.1.

In the inference time, we also have to apply the same template function to inferring input $f_{\text{template}}(\mathbf{x}_{\text{test}})$ in order to match the format.

6.3.3 Demonstration-based Training Samples (PALP-D)

On top of the previous *templified* dataset (PALP-T), PALP-D additionally concatenates some demonstrations to the front to maximize the capability of PLM to learn in-context. However, unlike ICL, which attaches all available samples, our method extracts a single representative demonstration per class (i.e., total $|C|$ demonstrations) and leverages them as demonstrations. Formally, let there exists k accessible training samples per each class label:

$$\mathcal{D}_{\text{train}} = \{(\mathbf{x}_j^i, y_j^i) | j \in k, i \in |C|\}, \quad (6.4)$$

where $|C|$ is the cardinality. Then, we choose one instance from each class and concatenate them into one prefix τ :

$$\hat{\mathbf{x}} = \tau \oplus f_{\text{template}}(\mathbf{x}), \text{ where} \quad (6.5)$$

$$\tau = D_1 \oplus \dots \oplus D_{|C|}. \quad (6.6)$$

In this way, we can avoid atrociously lengthy input, minimizing the computational cost and enhancing the method’s scalability. Specifically, while the length of the input in ICL is normally proportional to the total available number of samples, PALP-D is proportional to the number of labels of the task.

Given that PALP-D does not inject complete available training samples, the core is selecting a single meaningful demonstration that can distill PLM sufficient knowledge to comprehend essential signs of the task, such as input-label mapping, label space, or distribution of the input text. We hypothesize

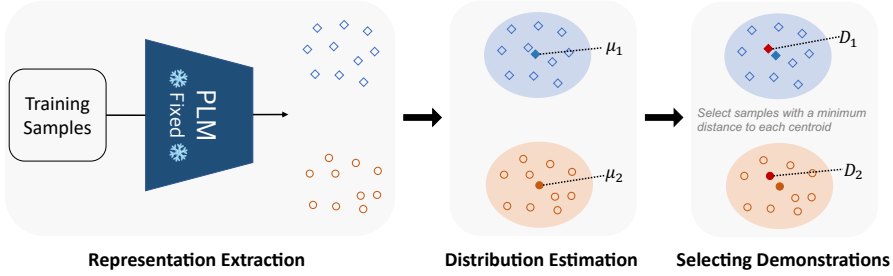


Figure 6.2: Illustration of selecting demonstrations in PALP-D in binary classification task. We estimate the normal distribution for each class from available training samples and select the closest sample respectively.

that the inputs closest to the centroid of each class label can capture the most representative information for respective classes and extract them as a set of demonstrations $P = \{D_1, D_2, \dots, D_{|C|}\}$.

In order to do so, we first estimate the normal distribution for each class label $\mathcal{N}(\mu_{i \in |C|}, \Sigma_{i \in |C|})$ from the available training inputs and measure the distance between entire training samples and the estimated centroid through Mahalanobis distance:

$$dist_{\text{Mahal}}(\mathbf{x}, \mu_i; \Sigma_i) = (\mathbf{x} - \mu_i)^\top \Sigma_i^{-1} (\mathbf{x} - \mu_i). \quad (6.7)$$

Finally, we select samples closest to each class centroid and utilize them as demonstrations $D_i = \min_{j \in k} (x_j^i, u_i; \Sigma_i)$. (Figure 6.2 illustrates the selection of demonstration from available training samples.)

In a few-shot scenario, we randomly permuted the order of available demonstrations P to construct multiple prefixes $\hat{\tau}$ in the training phase, where $\hat{\tau} \sim \sigma(P)$ and $\sigma(\cdot)$ refers to a random permutation operator. By doing so, we can generate $|C|!$ different prefixes that can give the effect of data augmentation and alleviate the data-scarcity problem. And in the inference phase or data-

Task type	Dataset	# Class	# Train	# Test
Single Sentence	SST2	2	67,349	872
	Rotten_tomatoes	2	8,530	1,066
	Offensive	2	19,916	860
	CoLA	2	8,551	1,043
	Stance_atheism	3	461	220
	Emotion	4	3,257	1,421
	AG_news	4	120,000	7,600
	TREC	6	5,452	500
	Banking77	77	10,003	3,080
	CLINC	150	15,000	4,500
Sentence pair	MNLI	3	392,702	9,832
	MRPC	2	3,668	408
	RTE	2	2,490	277
	BoolQ	2	9,427	3,270
	CB	3	250	56

Table 6.1: Statistics of 15 different datasets used in our experiments.

abundant setting, we attach a unified prefix τ (without random permutation) to the front of the test input to match the format with the training samples.

6.4 Experiments

6.4.1 Experimental Setup.

Backbone. In the following experiments, we adopt GPT-J (Wang and Komatsuzaki, 2021) as a main backbone of our experiments, with additional experiments using GPT-2 (Radford et al., 2019).

Datasets. To investigate the performance of each method in many different scenarios, we utilized 15 different datasets ranging from binary tasks to multi-class classification tasks in single sentence task and sentence pair tasks in various

domains as follows: SST2, CoLA, MRPC, RTE, and MNLI datasets from GLUE benchmark (Wang et al., 2019b), BoolQ, and CB datasets from SuperGLUE (Wang et al., 2019a), stance_atheism, and emotion datasets from tweet eval (Barbieri et al., 2020), and AGnews (Zhang et al., 2015), Rotten tomatoes (Pang and Lee, 2005), TREC (Li and Roth, 2002), CLINC (Larson et al., 2019), and Banking77 (Casanueva et al., 2020) datasets.

Setting & reporting methods. Our experiments cover both a few-shot setting and full training data setting. As a reporting method, we select currently prevailing linear probing methodologies (i.e., SVM, SLP, LR, GDA, and k -NN) and their application of PALP. In a few-shot setting, we take a closer look at how the performance of each method changes when the training input is converted into ICL style and compare them with ICL, which is known to yield superior performance in the data-hungry setting. In the full-shot setting, we investigate the performance gap between our method and the *white-box* training method (i.e., accessible to model parameters), such as Adapter (Houlsby et al., 2019) or full fine-tuning, which can be considered upper bound.

Other details. We optimized the hyper-parameters of each classification method on SST2 dataset with 4 Tesla V100 SXM2 32GB GPUs and universally utilized them in different settings. Additionally, we found a manual template for each task where ICL exhibited sound performance and utilized them universally in our methods. (All templates for each task are stipulated in Table 6.2.) For stable evaluation, we report the average of 5 different seeds (13, 27, 250, 583, 915) as a model performance.

Task type	Dataset	Template
Single	SST2	Sentence 1: [sent1]\nSentiment: [positive / negative]
	Rotten_Tomatoes	Sentence 1: [sent1]\nSentiment: [positive / negative]
	Offensive	Sentence 1: [sent1]\nSentiment: [non-offensive / offensive]
	CoLA	Sentence 1: [sent1]\nSentiment: [correct / wrong]
	Stance_atheism	Sentence 1: [sent1] Label: [none / against / favor]
	Emotion	Sentence 1: [sent1]\nSentiment: [anger / joy / optimism / sadness]
	AGnews	Sentence 1: [sent1]\nSentiment: [World / Sports / Business / Technology]
	TREC	Sentence 1: [sent1]\nLabel: [Description / Entity / Expression / Human / Number / Location]
	Banking 77	Sentence 1: [sent1] Label: []
	CLINC 150	Sentence 1: [sent1]\nLabel: []
	mnli	Sentence 1: [sent1]\nSentence 2: [sent2]\nLabel: [True / Neither / False]
	MRPC	Sentence 1: [sent1]\nSentence 2: [sent2]\nLabel: [True / False]
Pair	RTE	Premise: [sent1]\nHypothesis: [sent2]\nLabel: [True / False]
	BoolQ	Premise: [sent1]\nHypothesis: [sent2]\nLabel: []
	CB	Premise: [sent1]\nHypothesis: [sent2]\nLabel: []

Table 6.2: A list of template and verbalizer for each dataset.

GPT-J 4-shot per class													
Method		AG	SST2	Rotten	Stance	Emotion	TREC	CoLA	Offensive	MNLI	RTE	MRPC	AVG
B	k-NN	54.66	51.33	58.67	54.73	33.92	48.84	42.36	49.37	35.09	48.74	64.66	49.31
	LR	66.48	50.25	65.08	58.64	36.89	64.48	50.68	51.72	36.46	49.39	59.02	53.55
	SVM	66.20	50.41	65.91	59.27	36.59	65.92	48.90	48.72	36.02	49.31	61.96	53.56
	SLP	67.88	50.23	65.68	59.45	37.13	66.80	49.13	49.95	36.36	49.24	61.76	53.96
	GDA	66.32	50.41	65.93	58.64	36.88	66.44	48.88	48.72	36.03	49.24	61.96	53.59
T	k-NN	61.88	58.30	65.10	35.27	45.18	51.04	53.02	59.98	36.40	53.14	61.47	52.80
	LR	71.84	62.00	71.73	58.09	50.56	65.56	49.86	59.00	38.73	50.54	60.44	58.03
	SVM	72.08	63.67	71.33	56.27	50.27	66.36	53.08	62.02	37.93	49.60	61.18	58.53
	SLP	73.79	63.58	72.18	57.45	52.71	<u>68.44</u>	<u>58.81</u>	63.07	38.25	49.96	61.57	59.53
	GDA	72.82	64.20	71.33	56.27	52.86	66.00	53.08	62.02	38.13	49.60	61.18	58.86
D	k-NN	75.99	70.16	71.01	49.45	56.95	46.80	55.67	54.56	38.85	51.09	63.86	57.67
	LR	77.92	70.62	80.58	<u>61.27</u>	68.97	65.96	53.83	67.63	40.30	49.00	62.27	63.49
	SVM	77.96	75.46	79.42	55.82	68.91	66.24	55.30	63.75	<u>40.12</u>	51.54	64.76	<u>63.57</u>
	SLP	<u>80.69</u>	<u>77.41</u>	<u>81.01</u>	71.36	<u>70.38</u>	71.92	69.13	<u>72.35</u>	39.41	<u>54.66</u>	71.73	69.10
	GDA	76.08	70.16	68.05	55.91	65.66	63.76	54.42	69.18	39.32	50.53	64.15	61.57
ICL		81.74	91.77	90.45	23.09	72.76	64.00	37.89	73.19	36.04	55.60	<u>68.38</u>	63.17

Table 6.3: Experimental results on GPT-J in 4-shot per class settings. B, T, and D refers to a baseline, template, and demonstration individually. For each dataset, the **best method** is in bold and the second best method and is underlined.

6.4.2 Few-shot Results.

In the few-shot setting, we conducted an experiment based on the number of accessible samples for each task class. For instance, a 4-shot setting in the Sentiment Analysis task with 2 classes (positive and negative) means that a total of 8 samples are accessible, which is analogous to a balanced 8-shot setting in ICL. Tab. 6.3, 6.4 summarizes the performance of ICL, baseline linear probing methods, and their application of PALP (T and D) in the 4,8-shot setting. Baseline refers to utilizing raw input without modification, which is a conventional supervised learning paradigm. Template (PALP-T) and demonstration (PALP-D) refer to template-based training samples and demonstration-based training samples from our method individually. We now refer to a T for PALP-T and D

GPT-J 8 shots per class													
Method		AG	SST2	Rotten	Stance	Emotion	TREC	CoLA	Offensive	MNLI	RTE	MRPC	AVG
B	k-NN	63.05	50.55	65.42	58.09	36.05	60.16	52.41	52.26	35.69	51.70	59.17	53.14
	LR	74.56	57.50	74.15	65.73	41.90	79.08	52.79	58.47	37.43	51.34	58.97	59.27
	SVM	73.26	57.16	75.87	64.27	42.66	80.12	53.98	57.79	37.43	52.06	59.56	59.47
	SLP	74.72	57.91	75.25	63.82	42.87	80.92	54.34	55.93	37.37	51.05	59.46	59.42
	GDA	74.12	57.16	75.78	63.00	44.31	81.84	53.98	57.79	37.60	52.06	59.56	59.75
T	k-NN	69.65	61.93	65.46	56.45	49.87	62.44	45.29	56.35	38.13	50.90	56.42	55.72
	LR	78.52	69.33	77.04	66.00	59.93	79.04	51.51	59.30	41.24	52.56	63.48	63.45
	SVM	78.39	75.44	77.97	63.36	60.53	81.12	51.98	65.07	40.45	53.07	61.13	64.41
	SLP	79.61	72.80	78.03	66.09	62.36	80.12	52.10	64.37	40.87	52.49	61.67	64.59
	GDA	79.03	75.37	77.90	62.55	62.15	<u>81.68</u>	51.98	65.05	40.90	53.07	61.18	64.62
D	k-NN	79.21	67.98	78.82	49.63	58.31	58.36	50.43	63.26	38.24	51.37	56.94	59.32
	LR	<u>84.12</u>	73.78	85.33	<u>66.55</u>	66.66	69.04	<u>57.20</u>	69.21	42.25	53.43	64.30	66.53
	SVM	83.96	77.10	85.27	64.36	68.32	71.20	55.55	<u>71.65</u>	43.00	<u>53.60</u>	60.82	<u>66.80</u>
	SLP	85.27	<u>78.37</u>	<u>86.75</u>	69.27	<u>69.97</u>	75.76	69.63	71.23	<u>42.30</u>	53.26	71.94	70.34
	GDA	83.05	72.55	83.83	61.18	64.32	68.08	55.02	71.05	42.10	51.42	61.92	64.96
ICL		83.26	91.72	89.72	27.27	73.12	71.60	34.28	73.02	36.62	54.08	<u>68.38</u>	63.92

Table 6.4: Experimental results on GPT-J in 8-shot per class settings.

for PALP-D for short.

While ICL displays sound performance in various tasks, the baseline linear probing method exhibits poor performance compared to ICL. In sentiment analysis tasks (SST2, rotten tomatoes), the performance of ICL is above 90% only with 4-shot samples per class, whereas the baseline linear probing methodology almost makes arbitrary random decisions in the same environment (around 50% \sim 60%). However, the performance of ICL quickly saturates and scales poorly with the number of available training samples. And even in some cases, ICL performs worse than arbitrary decisions without understanding the target task at all (e.g., stance, CoLA). Furthermore, if the input length exceeds a certain level, it is infeasible to utilize ICL in the usual way. On the other hand, linear probing methods are much more scalable with the number of available samples, revealing stable performance regardless of the dataset. Moreover, their application of PALP boosts performance by a substantial margin minimizing

GPT-J Full dataset													
Method		AG	SST2	Rotten	Stance	Emotion	TREC	CoLA	Offensive	MNLI	RTE	MRPC	AVG
B	k-NN	88.87	76.15	84.99	68.64	56.72	91.80	66.73	74.42	43.63	51.26	67.89	70.10
	LR	90.96	91.97	86.96	72.32	72.20	97.60	77.37	73.60	71.07	68.59	75.22	79.81
	SVM	90.34	90.48	88.18	72.27	71.26	96.80	76.13	72.91	67.32	69.68	75.98	79.21
	SLP	90.43	90.88	89.51	75.00	75.40	97.00	77.87	80.69	70.70	70.71	76.44	81.33
	GDA	92.18	92.40	89.49	72.27	72.56	97.00	78.52	74.07	71.07	68.59	74.27	80.22
T	k-NN	89.87	87.61	85.83	66.82	72.55	91.60	66.16	73.84	51.70	57.40	73.04	74.22
	LR	92.58	92.66	88.84	77.73	79.80	96.60	78.81	80.00	76.28	73.65	80.39	83.39
	SVM	90.58	89.45	85.83	75.91	79.52	97.40	75.10	73.84	72.58	71.48	80.39	81.10
	SLP	92.49	93.35	89.87	78.36	81.30	97.60	79.19	82.67	77.41	72.20	82.84	84.30
	GDA	92.36	<u>94.27</u>	90.81	75.46	81.07	97.00	78.91	82.21	75.91	71.84	79.17	83.55
D	k-NN	90.69	91.17	89.96	75.00	73.89	89.40	69.70	77.09	51.54	54.51	72.55	75.95
	LR	92.47	92.73	90.54	77.73	79.73	95.40	80.25	82.79	71.94	76.17	76.96	83.34
	SVM	90.78	91.97	88.37	77.27	76.92	95.60	80.06	75.47	51.38	76.53	76.23	80.05
	SLP	92.58	93.37	<u>91.33</u>	83.51	82.31	94.00	77.31	82.23	76.28	77.62	80.39	84.63
	GDA	92.86	93.00	90.81	73.18	81.66	96.60	79.10	81.16	75.26	77.26	77.70	83.51
Adapter		95.50	95.53	90.26	<u>81.53</u>	<u>83.54</u>	<u>97.41</u>	84.48	84.67	88.84	<u>82.80</u>	88.48	88.46
Fine-tuning		<u>94.80</u>	94.15	91.79	81.25	84.41	97.22	<u>82.34</u>	<u>84.02</u>	<u>87.47</u>	83.33	<u>86.51</u>	<u>87.94</u>

Table 6.5: Experimental results of 11 different datasets on GPT-J in full datasets settings. B, T, and D refers to a baseline, PALP-Template, and PALP-Demonstration respectively. For each dataset, the **best method** is in bold and the second best method and is underlined.

the gap between ICL, especially in most single-sentence tasks. We can obtain around 5% improvement in average from each ablation (appending template and demonstrations) in the 4-shot setting, and some linear probing methods outperform ICL. In general, the most high-performance results were obtained from the SLP among the various linear probing methodologies.

6.4.3 Full-data Results.

Table 6.5, 6.6 display the performance of different methodologies when the training data is fully available. Appending a simple template also displays a

GPT-J Full train dataset					
Method		CLINC	Banking	CB	BoolQ
B	k-NN	74.78	69.06	71.43	63.01
	LR	92.91	89.44	80.36	62.70
	SVM	91.20	89.55	80.36	63.77
	SLP	91.52	89.43	80.35	62.70
	GDA	93.78	89.54	80.36	63.00
T	k-NN	90.42	86.82	78.57	63.55
	LR	95.76	91.17	83.93	63.39
	SVM	96.49	92.52	83.93	64.50
	SLP	95.16	91.58	83.93	66.30
	GDA	96.16	92.79	82.14	64.50

Table 6.6: ICL cannot be applied to tasks with a large number of classes while various linear probing methods are capable of these tasks. For each dataset, the **best method** is in bold.

significant advantage even in a data-rich scenario, obtaining considerable improvements in accuracy regardless of the methods or the dataset. Notably, the performance of k -NN increases dramatically with the application of the template (PALP-T), which improved accuracy by 16% on the Emotion dataset.

However, the method of appending demonstration (PALP-D) has more cons than pros in a data-abundant scenario. First, while PALP-D often performs similarly to or better than PALP-T, they also sometimes yield worse scores than PALP-T, leading to a similar performance on average. Speaking otherwise, PALP-D only makes the input more lengthy and entails much higher inference costs compared to PALP-T in a data-abundant scenario. Moreover, PALP-D is infeasible to be applied to some tasks inheriting the limitations of ICL, as can be seen in Table 6.6: tasks with a large number of classes (i.e., CLINC, Banking), or tasks with long inputs (i.e., CB, BoolQ).

Setting	Method	AUROC	Accuracy
CLINC	SLP + Baseline	91.94	92.6
	SLP + PALP-T	96.85	96.01
	SLP + PALP-D	98.33	97.3
	Fine-tuning	98.41	96.85
Banking77	SLP+Baseline	89.12	96.32
	SLP + PALP-T	92.83	97.6
	SLP + PALP-D	94.55	98.4
	Fine-tuning	93.73	97.64

Table 6.7: OOD performance of PALP and other transfer learning methods.

Nevertheless, PALP greatly minimizes the performance gap between *white-box* tuning methods, such as Adapter or full fine-tuning, and *black-box* tuning methods, which is around 7% with baseline linear probing methods, while our approach narrows this gap to nearly 4%. In particular, our method outperforms or reaches statistically equivalent performance to *white-box tuning* methods in some tasks, such as rotten tomatoes, TREC, and CLINC.

6.4.4 Out-of-Distribution Detection Results.

In this section, we measure the out-of-distribution detection performance of PALP in two different scenarios. Following the experimental settings from previous works (Cho et al., 2022; Zhang et al., 2022a; Shu et al., 2017; Fei and Liu, 2016; Lin and Xu, 2019), we validate our models in two different scenarios: far-OOD setting and close-OOD setting. For CLINC dataset, we train our model with whole training dataset and test with independent OOD test split from clinc dataset, which does not overlap with 150 classes in the training dataset. Outliers in CLINC OOD split are distributionally far from the training distribution (Zhang et al., 2022a), so it is relatively easy to discern. For Banking77, we

partition the dataset into 2 disjoint datasets (i.e., IND / OOD dataset) based on the class label. Since the both IND and OOD datasets originated from the equivalent dataset, they share similar distributions and properties, making the task more demanding. Thus, we refer to a CLINC OOD setting as far-OOD and split settings in Banking as close-OOD settings respectively.

Tab. 6.7. displays the OOD performance, where we trained SLP on top of PALP extracted features from GPT-J. PALP not only improves task accuracy, but it also exhibits significant performance gain in OOD detection performance. Both template attachment and demonstration attachment enhances OOD robustness. It is noteworthy that PALP minimizes the gap between fine-tuning and even outperforms fine-tuning in both IND metric and OOD metric. Considering the fact that PALP trains a single layer on top of extracted representation without any adaptation of the model parameters while fine-tuning trains the whole model parameters, the result is quite encouraging.

6.5 Analysis & Ablations

6.5.1 Application to Small PLM.

In this subsection, we examine our method for relatively small PLM to verify whether our method is transferrable to small language models. Namely, we report the performance of 3 different tasks (sentiment analysis, natural language inference, and multiclass classification) on GPT-2 large (Radford et al., 2019) in a 4-shot per class setting. Table 6.8 summarizes the performance. Similar to previous experiments, our method mainly shows considerable performance gain on single sentence tasks and relatively small improvement on challenging tasks like sentence pair tasks. To summarize, PALP also benefits smaller language models, unlike ICL, which is known to yield poor performance or be unable to

GPT-2 (Large) 4 shots per class				
Method		SST2	AG-News	MNLI
Baseline	k-NN	50.34	26.59	33.86
	LR	51.81	46.06	33.22
	SVM	50.99	36.18	34.24
	SLP	49.77	43.56	35.05
	GDA	50.23	28.23	33.87
PALP-T	k-NN	53.53	42.49	34.35
	LR	53.23	52.36	32.8
	SVM	51.77	44.10	34.04
	SLP	52.27	43.23	34.97
	GDA	54.91	35.51	33.26
PALP-D	k-NN	56.54	63.44	35.93
	LR	58.11	69.87	37.44
	SVM	57.86	70.88	37.84
	SLP	55.02	70.58	37.09
	GDA	58.56	61.78	36.54

Table 6.8: Results on GPT-2 (Large) in 4-shot per class setting. B, T, and D refers to a baseline, template, and demonstration individually. Our method is transferable to smaller model.

apply them to relatively small language models. However, the performance improvement is less significant with smaller PLMs since our methodology depends solely on the capability of the language model.

6.5.2 Dataset Visualization.

In this experiment, we visualize the representation space when the differing input pre-processing method is applied to the input. Figure 6.3 is the result of the t -SNE visualized representation of the SST2 task on GPT-J. Demonstration-best is the representation obtained by attaching the demonstration that showed

the best performance, and Demonstration-worst is the opposite. Table 6.9 summarizes the precise example of demonstrations and accuracy of the aforementioned best and worst cases.

Consistent with the experimental results, we confirmed that PLM could extract more distinctive representations when appropriate templates or demonstrations are concatenated to the input of interest. The fact that a more meaningful representation can be drawn by applying a template is understandable and quite intuitive, as research has already shown that applying a template can benefit fine-tuning performance (Liu et al., 2021b; Schick and Schütze, 2021a,b). What is even more intriguing is that adding demonstrations to the front can promote language models to derive a more task-specific and form a more distinguishable representation cluster. While the degree of improvement varies significantly, depending on the selected demonstrations, even concatenating the poorest performing demonstration yields a more clustered representation than the baseline result indicating the language model’s capability to learn from the context of the input (Min et al., 2022b). Although we did not specifically identify which demonstrations were more helpful and which were not, we found that mislabeled demonstrations can hurt the overall quality of the representations. As can be seen in Table 6.9, the second demonstration of the worst demonstrations is wrongly labeled, where we conjecture is the cause of the poor performance as advocated in a recent study Kim et al. (2022) that the mapping of input sentence and ground-truth label space can be crucial.

6.6 Related Work

Large language models such as GPT-3 (Brown et al., 2020) and ERNIE 3.0 (Sun et al., 2021) are often released as black-box APIs due to commercial consider-

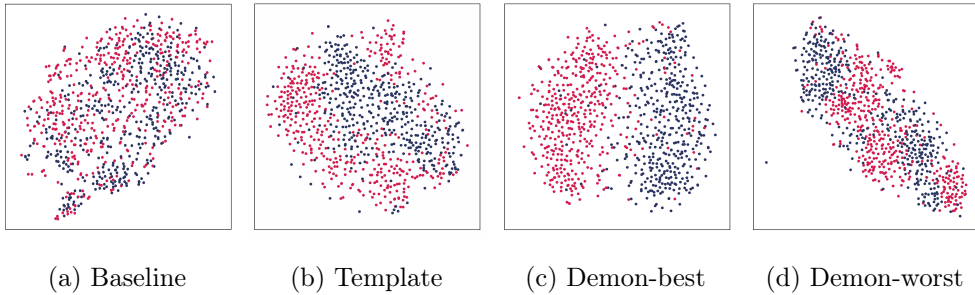


Figure 6.3: *t*-SNE visualization of SST2 representation from GPT-J. Adding understandable prompts to the input can reshape the representation into a more task-specialized form. Demon-best is obtained by attaching the demonstration that showed the best performance, and Demon-worst is the opposite.

Acc	Prefix (τ)	Visualization
Max (84.86)	Sentence 1: awful movie. $\backslash\backslash$ Sentiment: negative	Fig. 6.3c
	Sentence 1: soulful , scathing and joyous. $\backslash\backslash$ Sentiment: positive	
Min (54.62)	Sentence 1: without any passion. $\backslash\backslash$ Sentiment: negative	Fig. 6.3d
	Sentence 1: , incoherence and sub-sophomoric. $\backslash\backslash$ Sentiment: positive	

Table 6.9: Best and worst performing example prefixes from SST2.

ations and the potential risk of misuse. Thus, users are unable to train those pre-trained models with the traditional transferring paradigm (i.e., fine-tuning). Even in some cases where the weights of the pre-trained model are accessible (Zhang et al., 2022b; Scao et al., 2022), it may not be possible for many researchers to fine-tune those models due to the enormous resource they require. Several studies were proposed to circumvent the aforementioned problems:

6.6.1 Black-box Tuning.

Black-box tuning is a methodology that makes use of the target model without internal model access (e.g., gradient, middle layer representations, or weights).

As such, black-box tuning methods usually train a lightweight discriminator on top of pre-trained models or optimize input prompt in a derivative-free fashion since the gradient of the pre-trained language models is unavailable. Specifically, Diao et al. (2022) utilizes the natural evolution strategy (NES) to find better prompts for black-box models instead of using natural NES to fool the model as in black-box adversarial attacks. Sun et al. (2022) adopted a covariance matrix adaptation evolution strategy to perform optimization in a randomly generated small subspace, which is effective due to the low intrinsic dimensionality of large language models (Aghajanyan et al., 2021; Qin et al., 2021).

6.6.2 In-Context Learning.

ICL (Brown et al., 2020) is an alternative to the gradient-based tuning method. It is a novel transferring method that derives answers via conditioning appropriate prompts or often concatenating the training data. ICL is drawing explosive interest in the field of NLP due to their strong generalizability among many different tasks, from traditional natural language understanding tasks, including sentiment analysis and natural language inference (Wang et al., 2019b), to extreme ones, such as code generations (Poesia et al., 2022) or mathematical problems (Henighan et al., 2020).

As the underlying mechanism of ICL astonished the NLP field and has reminisced the capability of PLMs, a plethora of works has been proposed to utilize and understand ICL better. Studies include advanced ICL methods maximizing the downstream performance (Zhao et al., 2021; Min et al., 2022a; Holtzman et al., 2021), advanced methods of choosing example data (Liu et al., 2022a; Lu et al., 2022; Rubin et al., 2022), understanding the limitation of ICL (Liu et al., 2022a; Lu et al., 2022), and understanding the underlying mechanism of ICL (Xie et al., 2022; Reynolds and McDonell, 2021; Min et al.,

2022b; Razeghi et al., 2022; Kim et al., 2022).

Summary of PART II

In this study, we showed that the scale of the language model is an important factor in OOD robustness. Moreover, we also showed that various methodologies outperform fine-tuning when applied to sufficiently large PLM. Our follow-up work seeks to create a methodology that allows large PLMs to be more robust to OOD input. The performance improvement that can be achieved by the size of PLM and OOD technique is orthogonal. In line with the growing size of PLM, the OOD technique needs to be developed in a more parameter-efficient way. As such, developing a proper representation extraction technique or a novel scoring function compatible with the parameter-efficient transfer methods is our goal.

In this paper, we showed that providing task descriptions or demonstrations can enforce PLM to yield more robust representations without additional adaptation of the model weights, allowing them to be used for lightweight linear probing as an alternative to in-context learning. In light of this finding, we proposed prompt-augmented linear probing, where we augmented data representations with ICL-style crafted inputs. Our integrated approach is scalable with the available training data and the size of the language model. PALP obtains comparable results to ICL in the data-hungry scenario and comparable results to fine-tuning in the data-abundant scenario with little training overhead, potentially making PALP a strong alternative in various situations.

In our follow-up study, we will analyze how the additional prompt tokens (e.g., demonstrations or templates) affect the representation quality of the encapsulating input text. We are also interested in the effect of adopting self-supervised learning objectives, such as contrastive learning (Gao et al., 2021b),

to the shallow layers on top of the language model backbone, which might improve our method further.

Chapter 7

Conclusion

Anomaly detection is a long-standing problem that aims to discern outliers to secure strong reliability and maintain a high user experience of the software systems. This dissertation explored methods for enhancing robustness to outliers on deep neural network architectures in an unsupervised manner. Precisely, we have sought room for improvement in two orthogonal branches.

Part I (Chapter 3 & 4) of this dissertation attempts to enhance OOD detection capability via self-supervised learning which helps DNNs to learn more precise and distinctive representations by restraining them from learning unintended decision rules dubbed *shortcuts*. Accordingly, we propose self-supervised anomaly detection frameworks for images and sentences, representing sequential and non-sequential data types and the most frequently used inputs in human interaction.

In Chapter 3, we present **M**asked **C**ontrastive **L**earning (**MCL**) for image input, a task-specific variant of recent contrastive learning (Chen et al., 2020a). Among self-supervised learning tactics, *contrastive learning* is one spe-

cific framework validating their superiority in learning *task-agnostic* features without labels. MCL learns a more distinctive representation suited for anomaly detection by exploiting the label information in the training batch in addition to various image data augmentations.

In Chapter 4, we propose a novel framework dubbed **Layer-agnostic Contrastive Learning (LaCL)** for natural language, which encourages intermediate features of the language model to learn layer-specialized representations via a layer-agnostic training scheme and novel regularization loss. Then, LaCL *implicitly* assembles rich and diverse information in each layer into a single representation.

Part II (Chapter 5 & 6) of this dissertation primarily focuses on exploiting the large-scale pre-trained models as anomaly detectors. While large-scale pre-trained language models (PLMs) have achieved astonishing breakthroughs across a wide array of tasks, recent anomaly detection studies are still limited to relatively small bi-directional PLMs, which naturally begs the question: *How do large language models cope with outliers?*

In Chapter 5, to shed some light on the aforementioned unexplored research question, we first analyze the anomaly detection capability of the large PLMs from various perspectives and their combination with recent parameter-efficient transfer learning methods. From extensive experiments, we share some intriguing findings and limitations of large PLMs.

On top of the previous insights and limitations, Chapter 6 proposes a transferring method for large pre-trained models termed *prompt-augmented linear probing (PALP)* when the parameters of pre-trained language models (PLMs) are inaccessible. PALP exhibits robustness to anomalies as they can extract more distinctive representations from large PLMs by prepending extra prefix inputs inspired by the recent success of prompting methods.

The approaches proposed in this thesis improve the anomaly detection abil-

ity in diverse situations; nevertheless, several limitations remain. Firstly, although we confirmed the superiority of our frameworks among other competing methods in respective experimental settings, such setting covers limited scenarios. For instance, we need to investigate the compatibility of our method in various datasets, such as medical or autonomous-driving data, to verify the effectiveness of our methods in more practical applications. Moreover, despite the proposed methodology can be used in various situations (e.g., various data types or tasks), it has yet to be precisely analyzed whether it is a universal method that works well in universal scenarios or not. If not, we need to elucidate under what circumstances or differences are disrupting the framework’s underlying mechanism and degrading the model’s performance via in-depth analysis. In light of this limitation, our future work aims to expand the utility of the proposed methods to more general situations and adapt them into more appropriate forms considering the applying domain.

To conclude, we anticipate that the improvements proposed in this dissertation would compensate for the DNNs’ poor reliability in numerous real-world applications, as there is a growing concern for the lack of transparency and vulnerability of DNN-based applications in the wild.

Bibliography

- Aderemi O Adewumi and Andronicus A Akinyelu. 2017. A survey of machine-learning and nature-inspired based credit card fraud detection techniques. *International Journal of System Assurance Engineering and Management*, 8(2):937–953.
- Charu C Aggarwal. 2017. An introduction to outlier analysis. In *Outlier analysis*, pages 1–34.
- Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics, ACL*.
- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.
- Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. 2020. TweetEval: Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Proceedings of Findings of EMNLP*.
- Sara Beery, Grant Van Horn, and Pietro Perona. 2018. Recognition in terra

- incognita. In *Proceedings of the European conference on computer vision (ECCV)*, pages 456–473.
- Eyal Ben-David, Nadav Oved, and Roi Reichart. 2022. PADA: Example-based prompt learning for on-the-fly adaptation to unseen domains. *Transactions of the Association for Computational Linguistics*.
- Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.
- Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*.
- Azzedine Boukerche, Lining Zheng, and Omar Alfandi. 2020. Outlier detection: Methods, models, and classification. *ACM Computing Surveys (CSUR)*, 53(3):1–37.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2020. Semantic re-tuning with contrastive tension. In *International Conference on Learning Representations*.
- Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceed-*

ings of the 2nd Workshop on Natural Language Processing for Conversational AI.

Raghavendra Chalapathy and Sanjay Chawla. 2019. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58.

Derek Chen and Zhou Yu. 2021. GOLD: Improving out-of-scope detection in dialogues using data augmentation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. 2020b. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*.

Hyunsoo Cho, Hyuhng Joon Kim, Junyeob Kim, Sang-woo Lee, Sang-goo Lee, Kang Min Yoo, and Taeuk Kim. 2023a. Prompt-augmented linear probing: Scaling beyond the limit of few-shot in-context learners. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI*.

Hyunsoo Cho, Choonghyun Park, Jaewook Kang, Kang Min Yoo, Tae-uk Kim, and Sang-goo Lee. 2022. Enhancing out-of-distribution detection in natural language understanding via implicit layer ensemble. In *Findings of the Association for Computational Linguistics: EMNLP*.

- Hyunsoo Cho, Choonghyun Park, Junyeob Kim, Hyuhng Joon Kim, Kang Min Yoo, and Sang-goo Lee. 2023b. Probing out-of-distribution robustness of language models with parameter-efficient transfer learning. *arXiv preprint*.
- Hyunsoo Cho, Jinseok Seol, and Sang-goo Lee. 2021. Masked contrastive learning for anomaly detection. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI*.
- Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches.
- Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.
- Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT’s attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.
- Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009a. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009b. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, NAACL*.
- Akshay Raj Dhamija, Manuel Günther, and Terrance E. Boult. 2018. Reducing network agnostophobia. In *Advances in Neural Information Processing Systems*.
- Shizhe Diao, Xuechun Li, Yong Lin, Zhichao Huang, and Tong Zhang. 2022. Black-box prompt learning for pre-trained language models. *arXiv preprint arXiv:2201.08531*.
- William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.
- Geli Fei and Bing Liu. 2016. Breaking the closed world assumption in text classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics, NAACL*.
- Kunihiko Fukushima and Sei Miyake. 1982. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285.
- Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation:

- Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059.
- Tianyu Gao, Adam Fisch, and Danqi Chen. 2021a. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics, ACL*.
- Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021b. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP*.
- Mike Gashler, Christophe Giraud-Carrier, and Tony Martinez. 2008. Decision tree ensemble: Small heterogeneous is better than large homogeneous. In *2008 Seventh International Conference on Machine Learning and Applications*, pages 900–905. IEEE.
- Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.
- Izhak Golan and Ran El-Yaniv. 2018. Deep anomaly detection using geometric transformations. In *Advances in Neural Information Processing Systems*, pages 9758–9769.
- Yoav Goldberg. 2019. Assessing bert’s syntactic abilities. *arXiv preprint arXiv:1901.05287*.
- Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations*.

- Jean-Bastien Grill, Florian Strub, Florent Alth  , Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo   vila Pires, Zhao-han Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, R  mi Munos, and Michal Valko. 2020. Bootstrap your own latent - A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems, NeurIPS*.
- Douglas M Hawkins. 1980. *Identification of outliers*, volume 11. Springer.
- Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *The Tenth International Conference on Learning Representations, ICLR*.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. 2019. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 41–50.
- Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *5th International Conference on Learning Representations, ICLR*.
- Dan Hendrycks, Kimin Lee, and Mantas Mazeika. 2019a. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pages 2712–2721. PMLR.
- Dan Hendrycks, Mantas Mazeika, and Thomas G. Dietterich. 2019b. Deep

- anomaly detection with outlier exposure. In *7th International Conference on Learning Representations, ICLR 2019*.
- Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. 2020. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*.
- John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.
- Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface form competition: Why the highest probability answer isn’t always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP*.
- Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*.

- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR*.
- Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.
- Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know. *Transactions of the Association for Computational Linguistics*.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*.
- Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschiot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*.
- Junyeob Kim, Hyuhng Joon Kim, Hyunsoo Cho, Hwiyeol Jo, Sang-Woo Lee, Sang-goo Lee, Kang Min Yoo, and Taeuk Kim. 2022. Ground-truth labels matter: A deeper look into input-label demonstrations. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Taeuk Kim, Kang Min Yoo, and Sang-goo Lee. 2021. Self-guided contrastive learning for BERT sentence representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics ACL*.

- B Ravi Kiran, Dilip Mathew Thomas, and Ranjith Parakkal. 2018. An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. *Journal of Imaging*, 4(2):36.
- Clyde Kluckhohn. 1950. Human behavior and the principle of least effort.
- Edwin M Knox and Raymond T Ng. 1998. Algorithms for mining distancebased outliers in large datasets. In *Proceedings of the international conference on very large data bases*, pages 392–403. Citeseer.
- Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. *Master’s thesis, Department of Computer Science, University of Toronto*.
- Ludmila I Kuncheva and Christopher J Whitaker. 2003. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207.
- Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C Suh, Ikkyun Kim, and Kuinam J Kim. 2019. A survey of deep learning-based network anomaly detection. *Cluster Computing*, 22(1):949–961.
- Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing EMNLP*.
- Olivier Ledoit and Michael Wolf. 2004. Honey, i shrunk the sample covariance matrix. *The Journal of Portfolio Management*, 30(4):110–119.

- Haejun Lee, Drew A. Hudson, Kangwook Lee, and Christopher D. Manning. 2020. SLM: learning a discourse language representation with sentence unshuffling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*.
- Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. 2018a. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *6th International Conference on Learning Representations, ICLR 2018*.
- Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. 2018b. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 7167–7177.
- Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP*.
- Dan Li, Dacheng Chen, Jonathan Goh, and See-kiong Ng. 2018. Anomaly detection with generative adversarial networks for multivariate time series. *arXiv preprint arXiv:1809.04758*.
- Tian Li, Xiang Chen, Shanghang Zhang, Zhen Dong, and Kurt Keutzer. 2021. Cross-domain sentiment classification with contrastive learning and mutual information maximization. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8203–8207. IEEE.
- Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics, ACL*.

- Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.
- Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. 2017. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*.
- Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL*.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*.
- Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. 2017. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88.
- Fangyu Liu, Ivan Vulic, Anna Korhonen, and Nigel Collier. 2021a. Fast, effective, and self-supervised: Transforming masked language models into universal lexical and sentence encoders. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*.
- Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022a. What makes good in-context examples for gpt-3? In *Proceedings of Deep Learning Inside Out: The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, DeeLIO@ACL*.
- Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and

- Graham Neubig. 2022b. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*
- Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. 2020. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33:21464–21475.
- Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.
- Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR*.
- Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL*.
- Andrey Malinin and Mark J. F. Gales. 2018. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems 31, NeurIPS 2018*.
- Andrei Manolache, Florin Brad, and Elena Burceanu. 2021. DATE: detecting anomalies in text via self-supervision of transformers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021*.
- Donald Michie, David J. Spiegelhalter, and Charles C. Taylor. 1994. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood.
- Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022a. Noisy channel language model prompting for few-shot text classification. In

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL.

Sewon Min, Xinxu Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022b. Rethinking the role of demonstrations: What makes in-context learning work? *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Matthias Minderer, Olivier Bachem, Neil Houlsby, and Michael Tschannen. 2020. Automatic shortcut removal for self-supervised representation learning. In *International Conference on Machine Learning*, pages 6927–6937. PMLR.

Mehdi Mohammadi, Ala Al-Fuqaha, Sameh Sorour, and Mohsen Guizani. 2018. Deep learning for iot big data and streaming analytics: A survey. *IEEE Communications Surveys & Tutorials*, 20(4):2923–2960.

Seung Jun Moon, Sangwoo Mo, Kimin Lee, Jaeho Lee, and Jinwoo Shin. 2021. MASKER: masked keyword regularization for reliable text classification. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Poojan Oza and Vishal M Patel. 2019. C2ae: Class conditioned auto-encoder

- for open-set recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2307–2316.
- Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*.
- Aristotelis-Angelos Papadopoulos, Mohammad Reza Rajati, Nazim Shaikh, and Jiamian Wang. 2019. Outlier exposure with confidence control for out-of-distribution detection. *arXiv preprint arXiv:1906.03509*.
- Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, EMNLP*.
- Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, EACL*.
- Alexander Podolskiy, Dmitry Lipin, Andrey Bout, Ekaterina Artemova, and Irina Piontkovskaya. 2021. Revisiting mahalanobis distance for transformer-based out-of-domain detection. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*.
- Gabriel Poesia, Alex Polozov, Vu Le, Ashish Tiwari, Gustavo Soares, Christopher Meek, and Sumit Gulwani. 2022. Synchronesh: Reliable code generation from pre-trained language models. In *The Tenth International Conference on Learning Representations, ICLR*.

- Yujia Qin, Xiaozhi Wang, Yusheng Su, Yankai Lin, Ning Ding, Zhiyuan Liu, Juanzi Li, Lei Hou, Peng Li, Maosong Sun, et al. 2021. Exploring low-dimensional intrinsic task subspace via prompt tuning. *arXiv preprint arXiv:2110.07867*.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI blog*.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP*.
- Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*.
- Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–7.
- Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve

- prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics, NAACL*.
- David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *nature*, 323(6088):533–536.
- Chandramouli Shama Sastry and Sageev Oore. 2020. Detecting out-of-distribution examples with gram matrices. In *International Conference on Machine Learning*, pages 8491–8501. PMLR.
- Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.
- Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, EACL*.
- Timo Schick and Hinrich Schütze. 2021b. It’s not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics, NAACL*.
- Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. 2017. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pages 146–157. Springer.

- Vikash Sehwal, Mung Chiang, and Prateek Mittal. 2021. SSD: A unified framework for self-supervised outlier detection. In *9th International Conference on Learning Representations, ICLR 2021*.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016*.
- Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*.
- Yilin Shen, Yen-Chang Hsu, Avik Ray, and Hongxia Jin. 2021. Enhancing the generalization for intent classification and out-of-domain detection in SLU. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics ACL*.
- Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*.
- Lei Shu, Hu Xu, and Bing Liu. 2017. DOC: deep open classification of text documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP*.
- Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Unsupervised commonsense question answering with self-talk. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4615–4629.

- Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. In *International Conference on Machine Learning, ICML*.
- Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*.
- Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.
- Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. 2020. CSI: novelty detection via contrastive learning on distributionally shifted instances. In *Advances in Neural Information Processing Systems 33, NeurIPS 2020*.
- Mingxing Tan and Quoc V Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*.
- Mingxing Tan, Ruoming Pang, and Quoc V Le. 2020. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10781–10790.
- Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones,

- Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*.
- Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR*.
- Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. <https://github.com/kingoflolz/mesh-transformer-jax>.
- Bin Wang and C-C Jay Kuo. 2020. Sbert-wk: A sentence embedding method by dissecting bert-based word models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2146–2157.
- Jim Winkens, Rudy Bunel, Abhijit Guha Roy, Robert Stanforth, Vivek Natarajan, Joseph R Ledsam, Patricia MacWilliams, Pushmeet Kohli, Alan Karthikesalingam, Simon Kohl, et al. 2020. Contrastive training for improved out-of-distribution detection. *arXiv preprint arXiv:2007.05566*.
- Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *The Tenth International Conference on Learning Representations, ICLR*.
- Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran

- Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021*.
- Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. 2015. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*.
- Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. 2021. Barlow twins: Self-supervised learning via redundancy reduction. In *Proceedings of the 38th International Conference on Machine Learning, ICML*.
- Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Zijun Liu, Yanan Wu, Hong Xu, Huixing Jiang, and Weiran Xu. 2021. Modeling discriminative representations for out-of-domain detection with supervised contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics, ACL*.
- Li-Ming Zhan, Haowen Liang, Bo Liu, Lu Fan, Xiao-Ming Wu, and Albert Y. S. Lam. 2021. Out-of-scope intent detection with self-supervision and discriminative training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics ACL*.
- Hongjie Zhang, Ang Li, Jie Guo, and Yanwen Guo. 2020. Hybrid models for open set recognition. *arXiv preprint arXiv:2003.12506*.
- Jianguo Zhang, Kazuma Hashimoto, Yao Wan, Zhiwei Liu, Ye Liu, Caiming Xiong, and Philip Yu. 2022a. Are pre-trained transformers robust in intent classification? a missing ingredient in evaluation of out-of-scope intent detection. In *Proceedings of the 4th Workshop on NLP for Conversational AI*.

- Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuo-hui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022b. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.
- Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*.
- Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning, ICML*.
- Yinhe Zheng, Guanyi Chen, and Minlie Huang. 2020. Out-of-domain detection for natural language understanding in dialog systems. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1198–1209.
- Wenxuan Zhou, Fangyu Liu, and Muhao Chen. 2021. Contrastive out-of-distribution detection for pretrained transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP*.

초록

이상 탐지는 일반적으로 이상(異常)이라 불리는 일반적인 관찰을 벗어나는 데이터 입력을 식별하는 문제이다. 실제 상용되는 소프트웨어 시스템은 끊임없이 이상 입력을 마주하며, 이러한 이상 입력은 의료 진단이나 자율 주행 자동차와 같은 안전이 중요한 애플리케이션에서 치명적인 오류로 이어질 수 있다. 따라서 이상 징후를 식별하는 능력은 소프트웨어의 신뢰성을 확보하고 높은 사용자 경험을 유지하기 위한 초석이 되는 연구분야이다.

본 논문은 이상값에 대한 견고성을 강화하기 위해 심층 신경 모델에서 보다 분별력있는 표현을 추출하는 데 집중하여 다양한 비지도 학습 상황에서 이상 징후를 감지하기 위한 포괄적인 분석 및 새로운 방법을 제시한다. 이 논문의 첫 번째 파트는 인간 상호 작용에서 가장 많이 활용되는 입력의 형태인 이미지 및 문장에서 각각 두각을 드러내는 두 가지 자기 지도 학습 기반 프레임워크인 마스킹 대조 학습(Masked Contrastive Learning) 그리고 계층에 구애받지 않는 대조 학습론(Layer-agnostic Contrastive Learning)을 제안한다. 마스킹 대조 학습은 기존 대조 학습에 입력에서 레이블 정보를 활용해 반발 비율을 조정하는 마스크를 활용하는 방법론으로 클래스 별로 조금 더 밀집된 표현을 형성하도록 돕는다. 이와 동시에, 4방향으로 회전된 이미지를 따로 추론하여 얻은 결과를 취합하여 모델의 예측 결과를 더욱 증진시키는 방법을 활용하여 이상 탐지 능력을 높인다. 둘째로, 계층에 구애받지 않는 대조 학습론은 사전학습 언어 모델이 가진 풍부한 정보를 활용하기 위해 모든 층이 가진 정보가 서로 다르면서 유의미하도록 학습한 후, 이를 단일 표현으로 취합하는 네트워크를 제안한다.

본 논문의 두 번째 파트는 최근 기하급수적으로 커져가고 있는 사전 학습 언어 모델의 고유한 능력을 활용하여 이상탐지 능력을 극대화 하는 방법론을 찾는 것을 목표로 한다. 5장에서 우리는 먼저 다양한 관점에서 대규모 사전 훈련된 언

어 모델이 이상 검출기로서의 기능을 얼마나 갖추고 있는지를 확인하고, 이러한 큰 언어 모델이 최근에 제안된 매개변수 효율적인 전이 학습 방법과는 어떠한 연관성을 띄고 있는지를 분석하고 이로부터 얻은 몇 가지 흥미로운 발견과 한계를 공유한다. 6장에서는 5장에서 얻은 연구 결과를 바탕으로 프롬프트 증강 선형 프로빙(Prompt-Augmented Linear Probing)이라 불리는 대규모 사전학습 언어 모델로부터 더 의미있는 표현을 추출하는 방법론을 제안한다. 프롬프트 증강 선형 프로빙의 기본 메커니즘은 추가 접두사 입력을 앞에 추가하여 대형 PLM을 조작하는 프롬프트 방법의 최근 성공에서 영감을 받은 모델로, 모델의 매개변수에 대한 접근이나 업데이트 없이 대형 언어 모델으로부터 보다 분별력있는 정확한 표현을 추출할 수 있다.

본 학위 논문에서 제안된 개선 사항이 실제 상용되고 있는 다양한 심층 신경 모델 기반 응용 프로그램의 견고성을 향상시킬 것이라고 기대한다.

주요어: 자연어처리, 기계 및 심층 학습, 신경망, 문장 표현

학번: 2019-37513