

DEEP OPEN REPRESENTATIVE LEARNING FOR IMAGE AND TEXT
CLASSIFICATION

A Dissertation
IN
Computer Science
and
Telecommunications and Computer Networking

Presented to the Faculty of the University
of Missouri–Kansas City in partial fulfillment of
the requirements for the degree

DOCTOR OF PHILOSOPHY

by
MAYANKA CHANDRASHEKAR

M.Sc. Integrated Computer Science,
Women's Christian College, Chennai, India, 2014

Kansas City, Missouri
2020

© 2020

MAYANKA CHANDRASHEKAR

ALL RIGHTS RESERVED

DEEP OPEN REPRESENTATIVE LEARNING FOR IMAGE AND TEXT
CLASSIFICATION

Mayanka Chandrashekar, Candidate for the Doctor of Philosophy Degree
University of Missouri–Kansas City, 2020

ABSTRACT

An essential goal of artificial intelligence is to support the knowledge discovery process from data to the knowledge that is useful in decision making. The challenges in the knowledge discovery process are typically due to the following reasons: First, the real-world data are typically noise, sparse, or derived from heterogeneous sources. Second, it is neither easy to build robust predictive models nor to validate them with such real-world data. Third, the ‘black-box’ approach to deep learning models makes it hard to interpret what they produce. It is essential to bridge the gap between the models and their support in decisions with something potentially understandable and interpretable. To address the gap, we focus on designing critical representatives of the discovery process from data to the knowledge that can be used to perform reasoning.

In this dissertation, a novel model named Class Representative Learning (CRL) is proposed, a class-based classifier designed with the following unique contributions in machine learning, specifically for image and text classification, i) The unique design of a latent feature vector, i.e., class representative, represents the abstract embedding space

projects with the features extracted from a deep neural network learned from either images or text, ii) Parallel ZSL algorithms with class representative learning; iii) A novel projection-based inferencing method uses the vector space model to reconcile the dominant difference between the seen classes and unseen classes; iv) The relationships between CRs (Class Representatives) are represented as a CR Graph where a node represents a CR, and an edge represents the similarity between two CRs.

Furthermore, we designed the CR-Graph model that aims to make the models explainable that is crucial for decision-making. Although this CR-Graph does not have full reasoning capability, it is equipped with the class representatives and their inter-dependent network formed through similar neighboring classes. Additionally, semantic information and external information are added to CR-Graph to make the decision more capable of dealing with real-world data. The automated semantic information's ability to the graph is illustrated with a case study of biomedical research through the ontology generation from text and ontology-to-ontology mapping.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of the School of Graduate Studies, have examined a dissertation titled “Deep Open Representative Learning for Image and Text Classification,” presented by Mayanka Chandrashekar, candidate for the Doctor of Philosophy degree, and hereby certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Yugyung Lee, Ph.D., Committee Chair
Department of Computer Science Electrical Engineering

Deepankar Medhi, Ph.D.
Department of Computer Science Electrical Engineering

Praveen Rao, Ph.D.
Department of Computer Science Electrical Engineering

Zhu Li, Ph.D.
Department of Computer Science Electrical Engineering

Jeff Rydberg-Cox, Ph.D.
Department of English Language & Literature

Ye Wang, Ph.D.
Department of Communications Studies

CONTENTS

ABSTRACT	iii
ILLUSTRATIONS	x
TABLES	xiv
ACKNOWLEDGEMENTS	xvii
Chapter	
1 INTRODUCTION	1
1.1 Class Representative Learning for Image	3
1.2 Class Representative Learning for Text	7
2 CRL: Class Representative Learning for Image Classification	12
2.1 Related Work	16
2.2 Dynamic Model Building	23
2.3 Class Representative Learning Model	24
2.4 CR Graph: Relationships between Class Representatives	34
2.5 Implementation and Experimental Design	38
2.6 Discussion	73
2.7 Conclusion	75
3 Class Representatives for Zero-shot Learning using Purely Visual Data	76
3.1 Introduction	76
3.2 Related Work	79

3.3	Class Representative Learning Model	87
3.4	Class Representative Inference	96
3.5	Experiments	105
3.6	Results and Discussion	110
3.7	Discussion	122
3.8	Conclusion	123
4	MCDD: Multi-class Distribution Model for Large Scale Classification	125
4.1	Related Work	128
4.2	Multi-Class Discriminative Distribution Model	131
4.3	Experimental Results	140
4.4	Conclusion	147
5	Zero Shot Learning for Text Classification using Class Representative Learning	149
5.1	Introduction	149
5.2	Related Work	151
5.3	Text-based Class Representative Learning	152
5.4	CRL Experiments and Evaluations	160
5.5	Conclusions	170
6	Visual Context Learning with Big Data Analytics	172
6.1	Introduction	172
6.2	Related Work	174
6.3	Visual Context Model	176
6.4	VisContext Framework: Image Context	178

6.5	Results & Evaluation	185
6.6	Conclusion	195
7	Transformation from Publications to Ontology using Topic-based Assertion Discovery	196
7.1	Introduction	196
7.2	Related Work	199
7.3	Assertion Discovery Framework	201
7.4	Results and Evaluation	213
7.5	Diabetes Publication Ontology Application	218
7.6	Conclusion	221
8	Ontology Mapping Framework with Feature Extraction and Semantic Embeddings	223
8.1	Introduction	223
8.2	Related Work	226
8.3	Ontology Mapping Framework	228
8.4	Results and Evaluation	234
8.5	Conclusion	248
9	Conclusion	250
9.1	Class Representative Learning for Image	250
9.2	Class Representative Learning for Text	251
10	Future Work	253
	Appendix	
A	A Comparative Evaluation with Different Similarity Measures	255

BIBLIOGRAPHY	257
VITA	289

ILLUSTRATIONS

Figure		Page
1	Class Representative Learning for Image	4
2	Class Representative Learning for Text	8
3	Types of Semantic Space and Zero-shot Learning Methods	23
4	Class Representative Learning Architecture	24
5	CR-Graph in CIFAR-100	35
6	Inception-V3 Activation Visualization Layerwise	41
7	t-SNE Visualization of Class Representatives[82]	43
8	Benchmark Datasets: CalTech-101 & CalTech-256 & CIFAR-100 & ImageNet-1K	44
9	Accuracy Distribution and Cosine Similarity Distribution	49
10	Dynamic Models: CalTech-101 & CalTech-256	54
11	Dynamic Models: CIFAR-100 and ImageNet-1K	55
12	Dynamic Model: Mixed 4 Datasets - ImageNet, CalTech-101, CalTech-256, CIFAR-100	56
13	Accuracy with Increasing Class# in Dynamic Models	57
14	Reduction using Average Pooling	57
15	Class Representative Graphs	73
16	Zero-shot Learning and Semantic Spaces [4]	82

17	Label Embedding Based Zero-Shot Learning	89
18	Class Representative Based Zero-Shot Learning	90
19	Source Domain - Cosine Similarity Distribution	102
20	t-SNE Visualization for Class Representatives of Benchmark Datasets [82]	107
21	t-SNE Visualization of Class Representatives [82]	112
22	Cosine Similarity Distribution of 8 Benchmark Datasets	113
23	Accuracy Distribution of 8 Benchmark Datasets	114
24	Inference Performance for Two Best and Two Worst Cases	115
25	Accuracy on CRL Model based on Increasing Instances in Setting-2 ($\mathbb{S} \Rightarrow$ \mathbb{S} and $\mathbb{T} \Rightarrow \mathbb{T}$)	117
26	MCDD Deep Learning Networks	127
27	Example of the MCDD Distribution and Classification	130
28	ImageNet Communities 1 - 6: Superstars	137
29	ImageNet Communities 1 - 6: Trouble Makers	138
30	ImageNet Community Size: Confusion Graph vs. MCDDNet	140
31	ImageNet Community and Classification Accuracy	142
32	ImageNet Communities: 3 Best and 3 Worst Communities	145
33	ImageNet Communities: Accuracy Distribution	145
34	Class Representative - Text Classification	151
35	Overview of Universal Sentence Encoder using Deep Averaging Network	155
36	Universal Sentence Encoder based Deep Averaging Network	155
37	Two-layer feed-forward Network	156

38	Class Representative Generation	158
39	T-SNE Visualization for 20 Newsgroup Instances (USE+DNN+CRL) . . .	163
40	CNN/DNN Layer-based CRL Performance for ZSL with 20NG/DBP Datasets	165
41	VisContext Framework on Apache Spark	174
42	Evaluation: Effectiveness of NLP with RMSE	187
43	Evaluation: Root Mean Square Error(RMSE) vs. K Value	189
44	Evaluation: Feature Vector Size vs Precision	189
45	Evaluation: Feature Vector Size vs Recall	190
46	Evaluation: Feature Vector Size vs F-Measure	190
47	Evaluation: Average Log Likelihood of LDA Models	191
48	Comparison between EM and K-Means	192
49	Classification Algorithms: (a) Precision (b) Recall (c) F-Measure	193
50	20 Context Clusters of IAPR Dataset	194
51	20 Context Clusters of Flickr30K Dataset	194
52	Image Context Clusters of IAPR Dataset	194
53	Image Context Clusters of Flickr30K Dataset	195
54	Assertion Discovery Framework	198
55	Transformation from Publications to Diabetes Publication Ontology	204
56	Inter-topic Distance Map for Diabetes Publications	208
57	Diabetes Publication Ontology (DPO): Topic Entity Hierarchy (Topics 1- 10) & Property Hierarchy	209

58	<i>Diabetes Mellitus</i> and Associated Assertions in (a) OGMD (b) OGDI (c) DIAB (d) DDO	211
59	<i>Diabetes Mellitus</i> in (a) DMTO, (b) DPO, (c) Integration of DPO and DMTO	211
60	Mapping between DMTO and DPO	217
61	Web Application: Publication Searching	219
62	Web Application: Ontology Visualization	219
63	Web Application: List of Discovered Subject, Predicate, and Object from PubMed Abstracts	220
64	Web Application: List of Filtered Entities (Subject and Object) using Annotator API [237]	220
65	Ontology Mapping Framework	225
66	Tokens from Hypertension Ontologies	240
67	Word2Vec Embeddings	241
68	Ontology Connectivity in Hypertension	245
69	Ontology Mapping in Hypertension (Original Mapping vs. OMF Mapping)	246
70	Ontology Mapping in Hypertension (Strong Degree of Connectivity: Original Mapping vs. OMF Mapping)	246
71	OMF with TF & TF-IDF Feature Extraction: Ontology Mapping in Hypertension	247

TABLES

Tables		Page
1	Formal Symbol and Notations in the CRL model	27
2	Class Representative Relationships in CIFAR-100	35
3	Benchmark Datasets	43
4	Source Domain and Target Domain: Cosine Similarity Median, Accuracy, Kolmogorov-Smirnov Test Scores	45
5	Class Representative Distribution Statistics	45
6	CRL Models based on Deep Learning Networks (Inception-V3 vs. ResNet- 101)	50
7	Inception-V3 Layerwise Accuracy on Similar Domain Datasets	52
8	Feature Pooling and Top Accuracy	58
9	Image Classification Accuracy for an Increasing Image#	59
10	Dynamic Model Accuracy (Random Generation)	60
11	Transfer Learning Performance Analysis: CRL vs. Inception-V3 [80] Pretrained Model: Inception-V3 with ImageNet-1K	62
12	Comparison of Class Representative with Pre-trained Models([33], [87]) .	66
13	CR-Inception-V3 Accuracy with Increasing Target Instances (#Ins.) . . .	68
14	Accuracy for CR-Inception-V3 Zero-Shot Learning Tasks	69
15	CR Graphs for ImageNet-1K, CalTech-101, CalTech-256, CIFAR-100 . .	71

16	CR Graphs for ImageNet-1K, CalTech-101, CalTech-256, CIFAR-100 . . .	72
17	Related Work: Zero-Shot Learning Methods	83
18	Formal Symbol and Notations in the CRL model	91
19	Benchmark Dataset: Seen and Unseen Classes	106
20	Source Domain and Target Domain: Accuracy, Kolmogorov-Smirnov Test Scores, \mathcal{A} -distance	110
21	Accuracy for Zero-Shot Learning Tasks (Setting-2)	116
22	Comparison between the CRL model with VGG-19 Model for IN360 (Setting-1)	118
23	Accuracy of Generalized Zero-Shot Learning Algorithms with Setting-1 .	119
24	Transfer Learning Performance Analysis: CRL vs. Inception-V3 [80] Pre-trained Model: Inception-V3 with ImageNet-1K	122
25	Example of Confusion Matrix	134
26	Example of Confusion Factor Matrix	134
27	Example of Normalized Confusion Factor Matrix	135
28	Community-Based Classification: Confusion Graph [72] vs. MCDDNet on AlexNet with ImageNet	143
29	Comparison: SplitNet vs. MCDDNet	146
30	MCDD Hierarchical Classification Model Results on AlexNet	146
31	Performance with CIFAR-100 Image# on AlexNet	148
32	Notations for Text-based Zero-Shot Learning	153
33	Class Representative Text Classification Setting	167

34	Class Representative Learning Text Classification- Testing Accuracy . . .	168
35	Dataset for Zero-Shot Learning Evaluation	168
36	State-of-Art Zero-Shot Learning Model	168
37	Comparison of Zero-Shot Learning Models	169
38	Class-wise Accuracy for Class Representative Learning in Zero-shot Learning setting	170
39	Dataset Description	186
40	Diabetes Ontologies	203
41	DPO Dataset and Assertions	213
42	DPO Ontology: Entities/Annotated Entities and Assertions per Topic . . .	215
43	Mapping between Publications and Ontologies	218
44	Ontology Dataset in Hypertension Domain	236
45	Reuses in Other Ontologies	237
46	Ontology Formats of the NCBI BioPortal [244]	237
47	Ontology Properties for Extracting Ontological Features	238
48	Ontology Feature Match Example: Medication	239
49	Semantic Embeddings: Integration of TF-IDF and Word2Vec	242
50	Ontology Mapping	243
51	Ontology Mapping and Connectivity in Hypertension	244
52	Ontology Mapping (Source/Target) in Hypertension	245
53	Comparison between Similarity Measures in CRL Image Classification Task	256

ACKNOWLEDGEMENTS

Firstly, I wish to express my sincere appreciation to my advisor *Dr. Yugyung Lee*, who convincingly guided and encouraged me to become the best researcher I can and to do the right thing even when the road was tough. Without her persistent mentoring, this dissertation would not have been realized.

I want to thank my committee members for enhancing my research experience. I had the opportunity to work with my co-discipline advisor *Dr. Deep Medhi*, during my early Ph.D. years. Some of the fundamentals and research techniques I learned through him have stayed with me. *Dr. Praveen Rao* and *Dr. Zhu Li* are members closest to my area of research. Every discussion within planned or just down the hall always provided deeper and interesting perspectives I could incorporate into my research. I started collaborating with *Dr. Jeff Rydberg-Cox* and *Dr. Ye Wang* in my last years of Ph.D., provided the perfect platform to learn the perfect blend of inter-disciplinary research. This collaborating gave me the experience of active academic collaboration.

I want to thank the School of Computing and Engineering for providing me with an excellent education. Special thanks to my department chair, *Dr. Ghulam Chaudhry* for constant support and guidance.

I want to thank all the UMKC Distributed Intelligent Computing Association (UDICA) members for being present. Over the six years, thank you for creating some of the best projects I worked on. A few special mentions, *Feichen Shen* and *Pradyumna Doddala*, for helping me establish my research work; *Vijaya Yeruva*, *Rohith Nagulapati*,

and *Saria Goudarzvand* couldn't have asked better lab mates.

My deep and sincere gratitude to my family for their unparalleled love, help, and support. I am grateful to my mom *Sudha*, for helping me stand up and doing salsa with me to continue my research regardless of the obstacles. I am thankful to my dad *Chandra Shekar*, for having the vision of dissertation with me and constant reminder of my goals by asking “what happened to the journal paper!” . I am grateful to my husband *Amit*, for constant support and the much-needed sanity checks. I would like to thank my brother *Vivaswan* for making me humble and grounded, also for the reality checks. I would like to thank my friends *Monica* and *Divya*, for being my second family.

I would like to thank my therapist, counselor, doctors, and nurses for bringing back to life and giving me a second chance. I want to express gratitude to my friends and family, who were by my side during unfavorable times.

I express my gratitude to all the funding agencies; without the financial support completing my dissertation would not be possible. Below is a comprehensive list of funding I have received during of six years of Ph.D..

- UMKC School of Computing and Engineering Teaching Assistant
- UMKC School of Graduate Studies Travel Grants
- UMKC School of Graduate Studies Research Grants
- UMKC School of Graduate Studies Opportunity Fellowship
- UMKC Inclusive Excellence Grant

- UMKC Women's Council Graduate Assistance Fund
- UMKC Opportunity Scholarship
- Grace Hopper Celebration Scholarship
- Google Lime Scholarship
- NSF Center for Big Learning
- Numerous Travel Grants from UMKC SGS, UMKC OSI, IEEE & NSF

I would like to dedicate this dissertation to my grandparents, *Kamala & Venkataraman* and *Manorama & Sitaraman*. Their willpower and drive for life have been my constant inspiration.

CHAPTER 1

INTRODUCTION

The era of artificial intelligence (AI) and machine learning has brought sophisticated AI-systems in all walks of human life (as in medicine, defense, or education). With movement in AI towards deep learning, inherently encourages black-box machine learning. The increase in the use of black-box machine learning and deep learning models in critical context demands transparency of these decisions [1]. The need for transparency introduces the particular interest in the need for machine learning systems that can explain their rationale, characterize their strengths and weaknesses, and also convey an understanding of how they will behave in the future [2]. In the last few years, the need for explainable AI has increased drastically, and commendable research has been done towards this big goal. Explainable AI can be achieved through deep explanation, interpretable models, or model induction. This dissertation focuses on using data representations and graphs to aid explainable machine learning models.

The performance of machine learning is heavily dependent on the choice of data representation. For that reason, much of the actual effort in deploying machine learning algorithms go into the design of pre-processing pipelines and data transformations that result in a representation of the data that can support effective machine learning. A good representation is often one that captures the posterior distribution of the underlying explanatory factors of the observed input [3]. Good data representation gives us explanatory

information about the data it represents, making representations an essential piece towards explainable AI. The data representations of a given data provide an intricate knowledge of the structure of the data and the valuable information from a large amount of data. Data representations have been present for a long time since dimensionality reduction, feature extraction techniques; from there, it has come to the new world with intensive and beautifully designed deep learning techniques. The data representations have feature-level information about the data. They need to be coupled with the auxiliary data like semantics, attributes, and textual information to improve the reasoning. This information is typically inter-dependent and intertwined and needs a good knowledge representation. Graphs have become an essential part of developing useful data exploration tools, especially knowledge graphs.

Most available knowledge graphs are synthetic, seeding real-world properties into generation routines. Others are laboriously curated from source materials or manually created by domain experts. This process is time-consuming. So this dissertation presents a path: dynamically deriving the graph from real-world data sources. This dynamic graph generation process is used for creating data representation graphs. A larger corpus can be automatically processed with greater confidence by adequately grounding the tools in a small collection of curated datasets. The end system leverages the machine learning model's knowledge and reasoning, reducing the amount of time for human intervention through automation.

This dissertation's objectives can be split into two parts: class representative learning for images and class representative learning for text.

1.1 Class Representative Learning for Image

The Class Representative Learning for Image is done through three different goals. The first goal aims at designing class representatives for visual data for image classification settings, i.e., seen data. The next step is creating class representatives for unseen data with no learning, also known as the zero-shot image classification setting. The third goal is to create a discriminant distribution of classes based on the misclassification score. The discriminant distribution essentially groups heterogeneous classes or non-confusing classes to increase overall accuracy. Figure 1 shows the gist of each goal represented as a chapter and how chapters are interlinked in the dissertation. As discussed earlier, the whole architecture em-composes the steps involved in the transformation of data into knowledge. We take the data to create data representations (also known as class representations) and create knowledge graphs.

A brief abstract of each chapter is as follows:

Chapter 2: CRL: Class Representative Learning for Image Classification

Recent advances in deep learning (DL) have improved the state-of-the-art results of the data-driven approaches and applications in a wide range of domains. However, building robust and real-time classifiers with diverse datasets is one of the most significant challenges to deep learning researchers. It is because there is a considerable gap between a model built with training (seen) data and real (unseen) data in applications [4, 5, 6]. The current deep learning research assumes firm boundaries between data, between data and models, and between models in deep learning. There is no attempt to deal with this problem of breaking the boundaries or dynamically building a model. Consequently, the

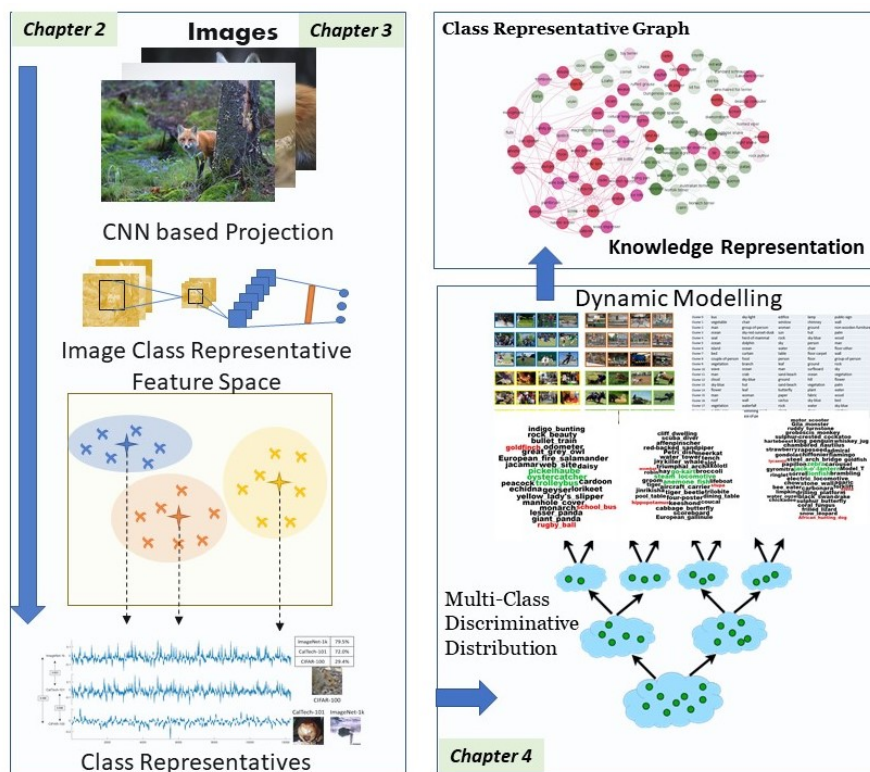


Figure 1: Class Representative Learning for Image

new paradigm focusing on the universal representation of diverse datasets and dynamic modeling depending upon users' contexts appears to be of great importance. There has been increasing attention on Zero-Shot Learning [4] and one-shot/few-shot learning [7, 8]. These efforts aim to build the ability to learn from a few examples or even without seeing them. Alternatively, it is required to represent and match new instances on a semantic space, minimizing training efforts and maximizing learning outcomes. They focus on active transfer learning by fully leveraging information from pre-trained models. The seamless integration of unlabeled data from the seen/unseen classes is possible through the artistic representations of multi-model embeddings, including semantic, word, visual embeddings.

Chapter 3: Class Representatives for Zero-shot Learning using Purely Visual Data

The building of robust classifiers with high precision is an important goal. In reality, it is quite challenging to achieve such a goal with the data that are typically noise, sparse, or derived from heterogeneous sources. Thus, a considerable gap exists between a model built with training (seen) data and testing (unseen) data in applications. Recent works, including zero-shot learning (ZSL) and generalized zero-shot learning (G-ZSL), have attempted to overcome the apparent gap through transfer learning. However, most of these works are required to build a model using visual input with associated data like semantics, attributes, and textual information. Furthermore, models are made with all of the training data. Thus, these models apply to more generic contexts but do not apply to the specific settings that will eventually be required for real-world applications. In this

chapter, we propose a novel model named Class Representative Learning (CRL), a class-based classifier designed with the following unique contributions in machine learning: i) The unique design of a latent feature vector, i.e., class representative, represents the abstract embedding space projects with the features extracted from a deep neural network learned only from input images. ii) Parallel ZSL algorithms with class representative learning; iii) A novel projection-based inferencing method uses the vector space model to reconcile the dominant difference between the seen classes and unseen classes. This study demonstrates the benefit of using the class-based approach with CRs for ZSL and G-ZSL on eight benchmark datasets. Extensive experimental results suggest that our proposed CRL model significantly outperforms the state-of-the-art methods in ZSL/G-ZSL based image classification.

Chapter 4: MCDD - Multi-class Distribution Model for Large Scale Classification

We need a parallel and distributed machine learning framework to deal with a large amount of data. We have seen unsatisfactory classification performance, especially with an increasing number of classes. In this chapter, we propose a distributed deep learning framework, called Multi-Class Discriminative Distribution (MCDD) that aims to distribute classes while improving the accuracy of deep learning with large-scale datasets. The MCDD framework is based on an evidence-based learning model for the optimal distribution of classes by computing a misclassification cost (i.e., confusion factor). These observations about learning attempts have been used to extend a classifier into a classification model hierarchy by learning an optimal distribution of classes. As a result, a distributed deep neural network model with multi-class classifiers (MCDDNet) was built

to optimize the learning process's accuracy and performance. The MCDD model has been implemented in parallel environments, Apache Spark, and Tensor Flow using large real-world datasets (Caltech-101, CIFAR-100, ImageNet-1K). MCDD can build a class distribution model with higher accuracy compared to existing models.

Chapter 5: Zero-shot Learning for Text Classification using CRL

Zero-Shot Learning (ZSL) has been a very active research area over recent years. However, the ZSL algorithms for text classification remains very limited despite considerable research efforts in NLP. In this chapter, we proposed a novel Class Representative Learning (CRL) framework for ZSL-based text classification that was designed using sentence-level word embeddings and deep neural network features. The experiments show that CRL achieved the highest overall accuracy compared with the state-of-the-art research in Zero-Shot Learning and Generalized Zero-Shot Learning (GZSL).

1.2 Class Representative Learning for Text

Figure 2 shows the gist of each goal for text data are represented as chapters and how text-based chapters are interlinked in the dissertation. Each chapter in the dissertation describes a specific problem that contributes to the entire architecture. As discussed earlier, the whole architecture em-composes the steps involved in the transformation of data into knowledge. We take the data to create data representations (also known as class representations) and create ontologies.

A brief abstract of each chapter is as follows:

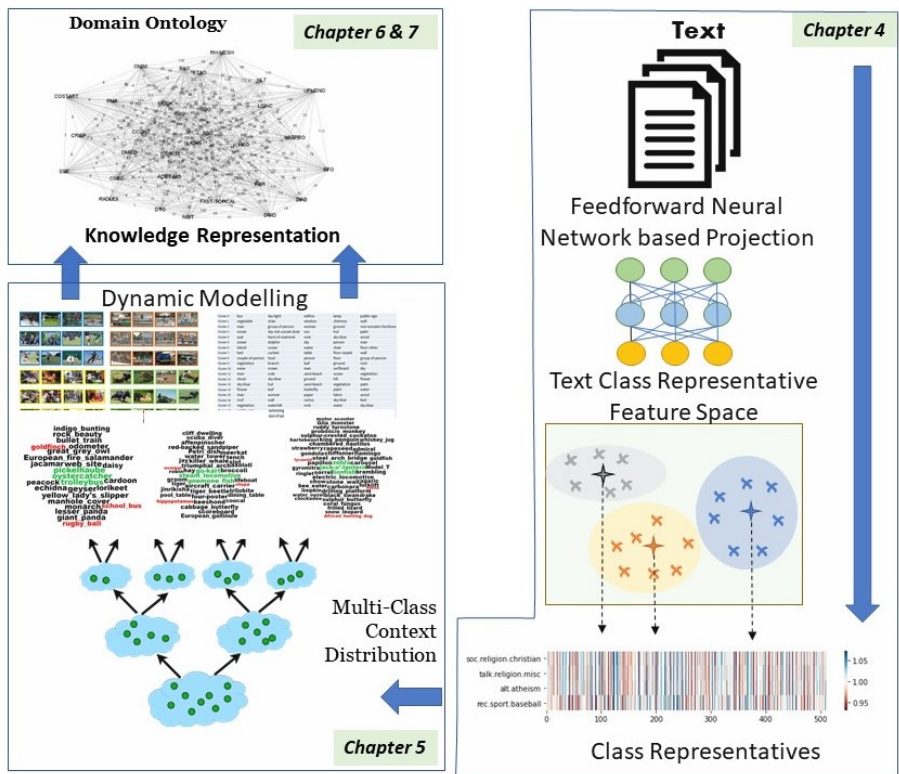


Figure 2: Class Representative Learning for Text

Chapter 5: Zero-Shot Learning for Text Classification using Class Representative Learning

With significant advances in supervised machine learning and enormous benefits from deep learning for a range of diverse applications. Despite the success of deep learning, in reality, very few works have shown progress in text classification. Transfer learning, known as the zero-shot learning (ZSL) or generalized zero-shot learning (G-ZSL), is receiving much attention due to its ability to transfer knowledge learned from a known domain to unknown domains. Nevertheless, most of the ZSL works are relying on large training corpus and external semantic knowledge. Thus, there are very few studies that have investigated the improvement of text classification performance in solely text-based ZSL/G-ZSL. In this chapter, we propose a class-based framework for zero-shot classification effectively that is based on a three-step approach consisting of: (1) sentence-based embeddings, (2) deep neural networks, and (3) class-based representative classifiers. Experimental results show that the proposed framework achieves the best classification results in text-based ZSL/G-ZSL compared with the state-of-the-art approaches investigated with three text benchmark datasets.

Chapter 6: Visual Context Learning with Big Data Analytics

Understanding contextual information composed of both text and images is beneficial for multimedia information processing. However, Capturing such contexts is not trivial, especially while dealing with real datasets. Existing solutions such as using ontologies (e.g., WordNet) are mainly interested in individual terms. Still, they do not support identifying a group of words that describe a specific context available at runtime. There are minimal

solutions regarding the integration of contextual information from both images and text within our knowledge. Furthermore, existing solutions are not scalable due to the computationally intensive tasks and are prone to data sparsity. In this chapter, we propose a semantic framework, called VisContext, based on a contextual model combined with images and text. The VisContext framework is based on the scalable pipeline that is composed of the primary components as follows: (i) Natural Language Processing (NLP); (ii) Feature extraction using Term Frequency-Inverse Document Frequency (TF-IDF); (iii) Feature association using unsupervised learning algorithms including K-Means clustering (KM) and Expectation-Maximization (EM) algorithms; iv) Validation of visual context models using supervised learning algorithms (Naïve Bayes, Decision Trees, Random Forests). The proposed VisContext framework has been implemented with the Spark MLlib and CoreNLP. We have evaluated the effectiveness of the framework in visual understanding with three large datasets (IAPR, Flick3k, SBU) containing more than one million images and their annotations. The results are reported in the discovery of the contextual association of terms and images, image context visualization, and image classification based on contexts.

Chapter 7: Transformation from Publications to Diabetes Ontology using Topic-based Assertion Discovery

During the last decade, we have seen explosive growth in the number of biomedical publications. In this chapter, we present an Assertion Discovery framework that aims to transform from PubMed publications (diabetes domain) to an ontology, called Diabetes

Publication Ontology (DPO). The assertions in the DPO ontology were mapped and integrated with ones in existing diabetes ontologies. The Assertion Discovery framework consists of three main components: (i) Assertion Discovery, (ii) Assertion Alignment, and (iii) Assertion Integration. The proposed approach for ontology generation was based on Stanford CoreNLP for Natural Language Processing, OpenIE (Open Information Extraction) for relation extraction, LDA (Latent Dirichlet Allocation) for topic modeling, and OWL API for ontology generation on the Spark parallel engine. We presented a web-based application for searching diabetes publications and retrieving the assertions from the diabetes publications through the DPO ontology.

Chapter 8: Ontology Mapping Framework with Feature Extraction and Semantic Embeddings

During the last decade, we have seen an increasing interest in biomedical ontologies. In this chapter, we proposed a semantic framework for an automatic ontology mapping through ontology search, feature extraction, and word embeddings. This framework is a new way to discover semantic mapping between the concepts across multiple ontologies. The ontologies were mapped to semantic features extracted from various ontologies selected from the NCBO BioPortal. We confirmed that the OMF framework effectively enhances the existing ontologies mapping and discovers new relations across ontologies beyond the boundary of ontologies.

CHAPTER 2

CRL: CLASS REPRESENTATIVE LEARNING FOR IMAGE CLASSIFICATION

Recent advances in deep learning (DL) have improved the state-of-the-art results of the data-driven approaches and applications in a wide range of domains. However, building robust and real-time classifiers with diverse datasets is one of the most significant challenges to deep learning researchers. It is because there is a considerable gap between a model built with training (seen) data and real (unseen) data in applications [4, 5, 6]. The current deep learning research assumes strong boundaries between data, between data and models, and between models in deep learning. There is no attempt made to deal with this problem of breaking the boundaries or dynamically building a model. Consequently, the new paradigm focusing on the universal representation of diverse datasets and dynamic modeling depending upon the user's context appear of great importance.

There has been increasing attention on Zero-Shot Learning [4] and one-shot/few-shot learning [7, 8]. These efforts aim to build the ability to learn from a few examples or even without seeing them. Alternatively, it is required to represent and match new instances on a semantic space, which results in minimizing training efforts and maximizing learning outcomes. They focus on active transfer learning by fully leveraging information from pre-trained models. The seamless integration of unlabeled data from the seen/unseen classes is possible through the expressive representations of multi-model embeddings, including semantic, word, visual embeddings. However, there are the notable limitations of

these Zero/Few-Shot Learning works: Many of them are relying on semantic embeddings in a common semantic space having a generative model [9, 10, 11].

The current works of Zero-Shot Learning demonstrated their effectiveness in transferred from prior experiences to new classes, which is a form of transfer learning. The most used semantic space in the Zero-Shot Learning model is supported by a joint embedding framework called Label-Embedding Space [4, 12]. This semantic space contains a combination of visual embeddings and word embeddings. The other popular semantic space is Engineering Semantic Space called Attribute Space, which uses attribute annotations for the ZSL model [13]. In contrast to prior work, we mainly extract the deep neural network features learned from visual inputs of seen classes creating image representatives, and we do not rely on any other features such as attribute annotations or word embeddings.

Similar to our approach in the feature extraction, there are active efforts [14, 15] for extracting important features from Convolutional Neural Networks such as Inception or ResNet. Mahendran et al. [14] analyzed the preserved deep features through inverting the fully-connected layers. Zhou et al. [15] built the class activation map using CNN features for the localization of the objects in the images for the discriminative image regions. Unlike [14] and [15], we are interested in generating class-specific markers using CNN features.

Our goal is to propose an innovate model, called *Class Representative Learning (CRL)*, for image classification for seen and unseen data. In this model, the focus is on

creating a universal representation called the class representatives using the source environment, which is typically pre-trained deep learning models. Given this goal, architectural improvements are not our purpose; instead, we explore the potential and impact the universal representation can make. It is desired to enable the universal representation to be trained from any existing architectures or datasets with reduced efforts and resources. The minimum requirement for the CRL model is to secure a suitable *source* (pre-trained) environment for a given dataset.

The CRL model can be classified as transfer learning, called meta-learning [16, 17]. The basic idea behind transfer learning is to use previously learned knowledge on different domains or tasks. The CRL model is based on the transductive approach that aims to project the target data onto a source environment for the extraction of features by mapping to unify the input spaces. The transductive property in transfer learning is to derive the values of the unknown function for points of interest (class-based or instance-based) from the given data (source environment or source domain) [4, 18].

The CRL model poses the property of being selective during inferencing. In other words, the CRL model can classify an input image to either source labels or target labels or both. Due to this property, the CRL model can behave like a traditional classification model. The Convolution Neural Network-based Classification models tend to be high in parameter requirements to achieve state-of-the-art accuracy [19, 20]. To show the superiority of the CRL model developed in this study, we have compared our CRL model against other state-of-the-art deep learning models.

The contributions of this chapter can be summarized as follows:

- The key contribution of our work is an effective way of building zero-shot classifiers. Classifiers are built by dynamically aggregating Class Representatives (CRs) depending upon the context in which it appears.
- The proposed modeling approach is also flexible enough to allow breaking down boundaries between datasets and building a model across domains through effective transferring knowledge from one domain to another.
- The unique contribution is that we can dynamically generate classifiers for image classification problems. The dynamic model in the CRL model was possible due to its ability to generate the Class Representatives (CRs) one by one through the aggregation of CNN activation features.
- Furthermore, we can represent the relationships of the CRs as a graph form. The CR Graph can be used for interpreting a model performance by identifying the accuracy of CRs as well as the similarity between CRs.
- A comprehensive evaluation of the proposed model has been conducted using the four benchmark datasets. The CRL model outperforms state-of-the-art Zero-Shot Learning (ZSL) in terms of learning time and accuracy. CRL also shows improved performance compared to the existing MobileNet models [21, 21, 22] for image classification.

2.1 Related Work

2.1.1 Dynamic Modeling for Classifiers

Chang et al. [23] presented a dynamic composition approach for discovering the optimal weights of different concept classifiers by aggregating the individual concept prediction scores. This technique is inspired by zero-shot learning frameworks [6, 24]. Aydin et al. [25] proposed dynamic Bayesian networks by sparsifying their parameters for protein secondary structure prediction. Some machine learning approaches were improved their accuracy by aggregating complementary features of data such as coding [26, 27] and dynamic pooling [28]. Among dynamic pooling works in deep learning for image analysis, [29] is limited due to their assumption of fixed regions while dynamic pooling with content-based regions [28]. Li and Vasconcelos [28] proposed a binary dynamic system for characterizing the action attributes, representation of human activity in attribute space [30], and a bag of words for attribute dynamics ([31]). These works were mainly concentrated on the attributes or features that would be used for classifiers. However, none of these works consider building a classifier dynamically for a specific context.

2.1.2 Transfer Learning

Recent studies have indicated the importance of transfer learning (TL) [32, 33] that aims to maximize the learning outcome by transferring a model developed for a task for building a model on another task. NASNet [22] explored the possibility of transferring from what learned from a small dataset (e.g., CIFAR-10) to a larger dataset (ImageNet-1K) through searching and utilizing a core architectural building block from the small

dataset.

He et al. [34] have shown that pre-trained models with an extensive data set like ImageNet-1K or with a small dataset like a subset of MS COCO have incredible influence in computer vision. Initialization with pre-trained models or evaluating with pre-trained features (e.g., unsupervised learning [35]) can reduce efforts and produce better results in Deep Learning (DL). It is possible because pre-training models are widely available, and learning from the models is faster than building from scratch.

The DL community has extensively studied transfer Learning [32, 33]. The transfer learning from ImageNet-1K in Decaf [36] showed substantial improvements compared to learning from image features. Ravi et al. [16] also presented a meta-learner model that supported the quick convergence of training with a new task using few-shot learning.

Pan et al. [37] defined an inductive transfer learning as cross-domain learning where the target task is different from the source task. The data in the target domain are required to induce a predictive model that can be transferred from the source domain to the target domain. Our model is similar to the Feature-representation-transfer defined by Pan et al. However, the difference is that our model was encoded based on the aggregation of the high-level features extracted from Convolutional Neural Networks.

In our chapter, we used a pre-trained model only for feature extraction, but training is not required with new data. After the fully connected layers are removed from the entire network, the rest will be mainly used for feature extraction for new data. Thus, the use of the pre-trained model in our study is different from others since we only use it as a reference model for extracting features for new data.

2.1.3 Universal Representation

Ubernet [38] is a *universal* CNN that allows solving multiple tasks in a unified architecture efficiently. It is through the end-to-end network training with a single training set for diverse datasets and low memory complexity. Universal representations [39, 40] perform well for visual domains in a uniform manner and have proven to be efficient for multiple domain learning in relatively small neural networks.

Rebuff et al. [41] demonstrated that universal parametric families of networks could share parameters among multiple domains using parallel residual adapter modules. Similar to our work, all these works presented universal representations for multiple domains or multiple tasks. However, unlike CRL, none of them focus on dynamically generating a model for multiple domains.

In this chapter, we define *Source Environment* for providing a basis for feature selection as well as a uniform representation of a set of heterogeneous data sources for effective deep learning. Feature selection is a crucial step in machine learning since it directly influences the performance of machine learning. (e.g., as the right choice of features drives the classifier to perform well). However, Kapoor et al. [42] observed that finding useful features for multi-class classification is not trivial due to the volume in the high-dimensional feature space as well as the sparseness over the search space.

Dictionary learning [43] was presented to determine the subspaces and build dictionaries by efficiently reducing dimensionality for efficient representations of classes of images. The critical contribution of the work is the reduction of sparsity constraints and the improvement of accuracy by identification of the essential components of the observed

data. From the extracted set of relevant features from images and quantizing them with these bags of visual words, we will further build up a visual CR vector for each class by combining these primitive features. The visual CRs will be used for efficient learning as well as recognition with large scale multi-class datasets.

2.1.4 Lightweight Deep Learning

Recently, there has been an increasing demand for mobile applications for small networks or dynamic networks in deep learning. There have been several deep neural architectures to strike an optimal balance between accuracy and performance. The lightweight deep learning was achieved using three types of layer compression techniques, namely: weight compression, convolution compression, and adding a single layer [44]. Weight compression is the primitive technique used to create a lightweight model. MobileNet-V1 [45] and Shufflenet [46] used a convolution compression technique in its architecture, specifically depth-wise separable convolutions and point-wise group convolution, respectively.

As an extension of the previous works, MobileNet-v2 [21] added a new layer, namely an inverted residual layer, with a narrow bottleneck to create a lightweight model. In NasNet Mobile [22], a new paradigm, called Neural Architecture Search (NAS), was proposed with reinforcement learning for knowledge transfer. In general, architectural changes are typically considered to achieve a lightweight model. In the CRL model, we have obtained a lightweight model through the flexibility of representation concerning the class.

2.1.5 Matching Networks

Few-shot classification [7, 8] is to label new classes which are not seen in training, but through matching with only a few examples of each of these classes. The matching networks are similar to a weighted nearest-neighbor classifier in an embedding space. The embedding in the matching networks was built as a form of sampled mini-batches, called episodes, during training. Notably, matching networks [8] is similar to our work in terms of mapping an attention-based embedding to a query set for predicting classes. However, our model is different from these works since we can build a dynamic model for multiple classes by assembling a set of a single class classifier, called Class Representative, built one by one independently.

A meta-learning approach [16] aims to build a custom model for each episode based on LSTM, unlike others building each episode over multiple episodes. The prototypical networks built a class prototype by computing the mean of the training set in the embedding space. The inferring step was achieved by finding the nearest class prototype for a query set. This approach is very similar to our work in terms of building the class's prototype as a class abstraction in shared embedding space [47]. Recently, Wang et al. [48] extended the performance of Zero-Shot Learning and Few-Shot Learning using latent-space distributions of discriminative feature representations. Similar to our approach, they used only the feature extractor of the CNN model. However, they are different from our work in terms of the following aspects: (1) they used variational autoencoder (VAE) while we are using a vector space model with a cosine similarity measurement, (2) they built a model for all classes in any given dataset while we are building

a model class by class. They focused on learning an embedding of the meta-data into a shared space. However, in our work, we build CRs class by class. Thus, once the CR is generated, there is no dependence between CRs. Due to the independence between CRs, we can build multi-class models dynamically.

2.1.6 Zero-Shot Learning

Zero-Shot Learning (ZSL) defined a semantic encoding for predicting new classes by using a standard feature set derived from a semantic knowledge base [24]. All well-known works have worked on learning and understanding explicit and external attributes. Much of the early work adapted from the original definition of the semantic knowledge base in Zero-Shot Learning [24], focused on attributes solely based on visual feature learning. Some of the works in the feature learning include boosting techniques [49], object detection [50], chopping algorithm [51], feature adaption [52], and linear classifiers [53].

The recent works of Zero-Shot Learning can be categorized into Engineered Semantic Spaces (ESS) and Learned Semantic Spaces (LSS) according to the semantic space type and ZSL methods in Wang et al. [4]. ESS can be further sub-categorized into Attribute Space, Lexical Space, and Text-Keyword Space; and LSS into Label-Embedding Space, Text-Embedding Space, and Image-Representation Space. The Zero-Shot Learning method is classified into Classifier-based Methods and Instance-Based Methods. According to this categorization, the CRL model can be classified as Image Representation

Semantic Space and Instance-Based Method, specifically Projection Method (see Figure 3). The Projection method provides insights for labeled instances from an unseen class by projecting both the instance's feature space and the semantic space prototype to a shared space [4].

Most of the recent work includes two kinds of semantic spaces, namely Label-Embedding Spaces [4, 12] and Attribute Spaces [4] (also known as Probability Prediction Strategy [12]). Label-Embedding Spaces focuses on learning a projection strategy that connects image semantic features to labels, in which labels are represented a high dimensional embedding using Word2Vec [54] or Glove [55]. Image Features in Label-Embedding Spaces are typically learned from Convolutional Neural Networks [13, 56, 57, 58, 59, 60]. Attribution Spaces or Probability Prediction initially focuses on pre-training attribute classifiers based on source data [12], where an attribute is defined as a list of terms describing various properties of given a class [4]. Each attribute forms the dimensions of class; value is typical given if a class contains the attribute or not [61, 62, 17].

Pure Image Representation Space-based ZSL is rarely observed, one of the very first works was to use Image Deep Representation and Fisher Vector for the ZSL Project method [63] and an extension of this approach was used to create unsupervised domain adaptation [64]. Zhu et al. used the partial image representation method to achieve a universal representation for action recognition [65].

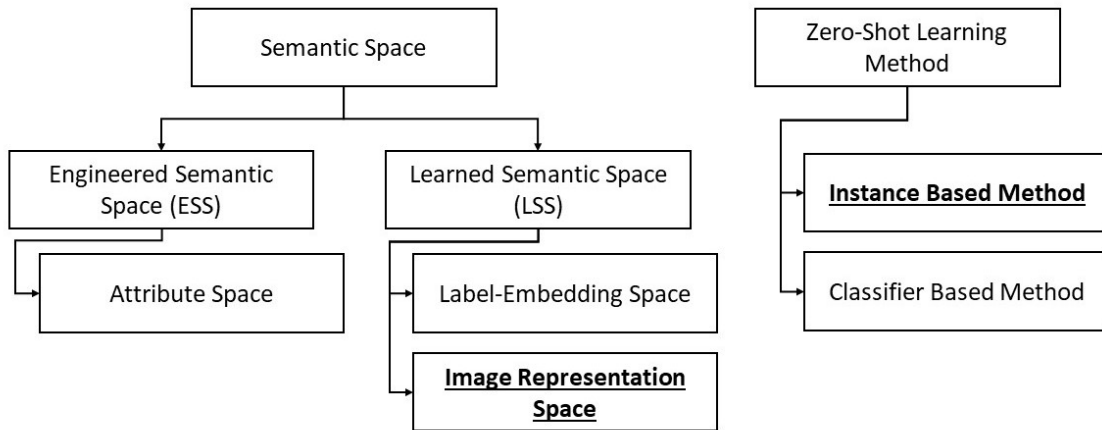


Figure 3: Types of Semantic Space and Zero-shot Learning Methods

* The CRL model belongs to the highlighted types.

2.2 Dynamic Model Building

The CRL framework supports dynamic modeling that can generate a deep learning model on the fly. The models generated by the deep learning community are mostly static. Our dynamic modeling is different from existing dynamic modeling supported by online learning researchers. We typically define that a static model is trained offline and no more updated. In contrast, a dynamic model is trained online with updated continuously by incorporating new data into the model. The dynamic model is created by selecting the Class Representatives of a set of classes. The subset of classes can be done through any given criteria, namely context-based selection criteria, random selection criteria, or association based. As the Class Representatives are independent of each other, the creation of a dynamic model is done by mere selection, so there is no additional cost associated with the model generation. The dynamic model generation is technique is similar to the ensemble model with no learning component and no additional cost with a changing set

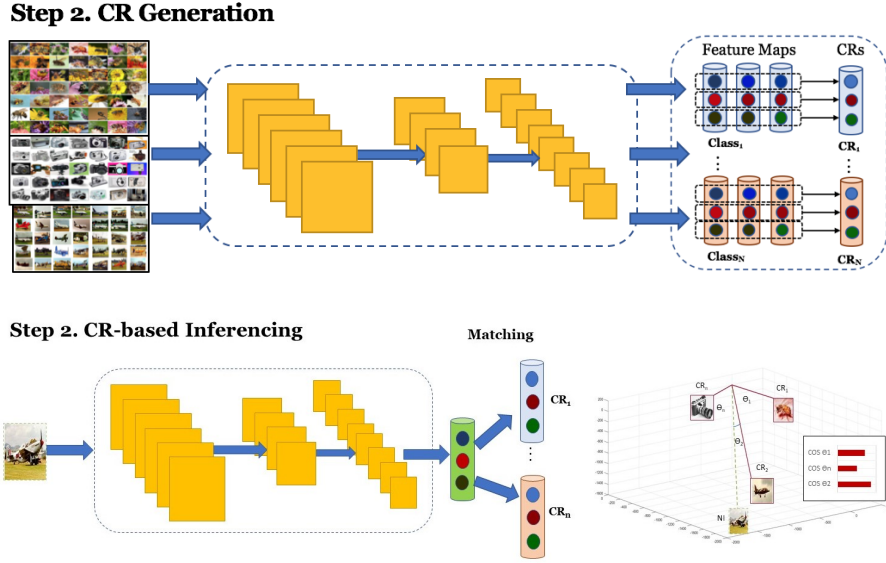


Figure 4: Class Representative Learning Architecture

of chosen classes. Source \mathbb{S} provides the environment to understand the Target \mathbb{T}

2.3 Class Representative Learning Model

The significance of the Class Representative Learning (CRL) model is its competence to project the input data to a global space that is specified by the activation of neurons in the pre-trained model such as CNN. The space of the CRL model is similar to the universal representation proposed by Tamaazousti et al. [66], where visual elements in the configuration (e.g., scale, context) can be encoded universally for transfer learning. The fundamental concept of the CRL model is its ability to represent the representatives of class in parallel and independently without depending on other classes. Also, universal representation allows us to integrate heterogeneous data from diverse domains or multiple modalities to generate a model dynamically. Thus, the CRs are a basis for generating a

new model by integrating data from different sources, platforms, and modalities.

As shown in Figure 4, the CRL model is composed of two primary components, namely: CR Generation and CR-based Inferencing. The model used to evaluate the CRL model is the Inception-V3 model that was pre-trained with ImageNet-1K [67]. The pre-trained model is the *Source* environment for the CRL model where no learning is happening, but the *Source* environment was mainly used as a reference standard for producing a feature vector of the input data in space. Figure 4a shows the *Source* environment (i.e., Pre-trained model), and Figure 4b shows the inferencing process with CRs on how a new image is projected on the *Source* environment and is mapped it on to the CRs for classification.

2.3.1 Problem Setup

Assume the given source data $D_s = \{x_n, y_n\}_{n=1}^{m_s}$ of m labeled points with a label from the source class $1, \dots, S$, where $x_n \in \mathbb{R}$ is the feature of the n^{th} image in the source data and $y_n \in \mathbb{S}$ where $\mathbb{S} = \{1, \dots, C_s\}$ is the number of source classes. The target data $D_t = \{x_n, y_n\}_{n=1}^{m_t}$ classes where $y_n \in \mathbb{T}$. The target classes \mathbb{T} are represented as $\{C_s + 1, C_s + 2, \dots, C_t\}$ where C total number of classes is $C = \mathbb{S} \cup \mathbb{T}$. For each class $c \in \mathbb{S} \cup \mathbb{T}$, has a Class Representative $CR(c)$, which is the semantic representative of class c . Furthermore, the source label \mathbb{S} and the target label \mathbb{T} are considered such that $\mathbb{S} \cap \mathbb{T} = \emptyset$. [Note: For simplicity, the source and target datasets have overlapped labels, but these overlapped classes are considered distinct.] In the CRL model, the source data are considered as *seen*, and the target data are *unseen*. In other words, the target data are

not used in the learning process. Table 1 summarizes all the symbols and notations used in the CRL model.

The goal of the CRL model is that given a new test data x^* , the model will classify it into one of the classes y^* where $y^* \in C$. The CRL model defines a universal problem for a classification approach as well as a traditional Zero-Shot Learning approach as follows:

- Classification (CL) Approach : $y^* \in \mathbb{S}$
- Zero-Shot Learning (ZSL) Approach: $y^* \in \mathbb{S} \cup \mathbb{T}$

In both models, there are no dependencies among CRs. The difference between these two models is in the properties of the inference. If the CR of the target set was introduced, then it would be *CL*, and if both CRs of the source set and the target set are introduced, then it would be *ZSL*.

2.3.2 CR Definition and Property

Definition 1: *Activation Feature*

Activation Feature is a feature value generated from a source model kernel for a given input data, as shown in Equation 2.1.

$$\begin{aligned}
 a(j)_{j=1}^n &= (I * K)(i) \\
 &= \sum_d I(i - d)K(d)
 \end{aligned}
 \tag{2.1}$$

The feature value $a(j)$ is generated from the convolution of a two-dimensional

Table 1: Formal Symbol and Notations in the CRL model

Notation	Description
$D_s \& D_t$	Source and Target Domain
$m_s \& m_t$	#Data Points from Source and Target, respectively
$C_s \& C_t$	#Classes from Source and Target, respectively
$\mathbb{S} \& \mathbb{T}$	Source and Target Label Set
C	#Classes
x	Feature Vector of Labeled Data Point
y	Label of Data Point
j	#Neurons in a Given Activation Layer
AFM_s	Activation Feature Map $\{b_1, b_2, \dots, b_j\}$
$CR(c)$	Class Representative of Class c where $c \in C$
$CR(c)$	Class Representative Set $\{CR_c^1, CR_c^2, \dots, CR_c^n\}$
x^*	Unlabeled Data Point
CRC	Class Representative Classifier
$CRFS^n$	Class Representative Feature Space
n	Dimensions of the $CRFS$

input image I with dimensions d and two-dimensional kernel K as shown in Equation 2.1 [68]. The Activation Feature is the element (also known as a single neuron) within the resultant matrix generated from Equation 2.1. The kernel K is the combination of weights and biases, where each matrix element in the two-dimensional has a weight w and bias b . The activation feature value j in n dimensions can be represented as a vector like the Activation Feature Map.

Definition 2: *Activation Feature Map*

Activation Feature Map (AFM) is a vector of features extracted from a base model that will be defined by a pre-trained model for any given instance. For a given input, AFM represents the features that are defined by the activation of neurons in the base model. The AFM dimensionality is the number of neurons in the selected layer of the base model. In

other words, it is the number of distinct neuron activating neurons occurring in the corpus. The n dimensional AFM forms the basis for the Semantic Space that is defined in Zero-Shot Learning (ZSL).

Definition 3: *Class Representative*

Class Representative (CR) is a representative of K instances in a single class. The Activation Feature Map of the CR is a unique characteristic pattern of visual expression that occurs as a result of the deep learning process in Convolutional Neural Networks (CNN). Thus, CR is an abstraction of instances of a class by computing an aggregation of the average mean vectors of the AFM for the K instances. The CR characterizes a class and differentiates one class against another. The Class Representatives $CR(c)$ for Class c is represented as $\{CR_c^1, CR_c^2, \dots, CR_c^n\}$ with n dimensions. Each dimension corresponds to a separate feature. If a feature occurs in CR, its value in the vector is non-zero.

Definition 4: *Class Representative Classifier*

A Class Representative Classifier $CRC : I^d \rightarrow C$ maps an input image space I^d of the dimension d to the Class Representative Feature Space $CRFS^n$ of the dimension n to classify it to Class C . (CRFS is defined in Definition 5). CRC is defined as a composition of two functions as shown in Equation 2.2.

$$\begin{aligned}
 CRC &= L(S(.)) \\
 S : I^d &\rightarrow CRFS^n \\
 L : CRFS^n &\rightarrow C
 \end{aligned}
 \tag{2.2}$$

The CRC model first maps the Input Space I^d to Class Representative Feature Space $CRFS^n$ with n dimensions. $CRFS^n$ further maps into Class C . The mapping function S represents the source environment, which aids the transformation of the input data into the Feature Space. (The source environment is further discussed in Section 2.3.2). For example, the input of a *dog* image with dimensions [299x299x3] is mapped into Class Representative Space (as defined in Definition 4). Then, the CRFS will be labeled with the class *dog* through the classification process [24, 69, 70].

Definition 5: *Class Representative Feature Space*

Class Representative Feature Space (CRFS) is a n dimensional semantic feature map in which each of the n dimensions represents the value of a semantic property. These properties may be categorical and contain real-valued data or models from deep learning methods [24]. The Class Representative Feature Space represents n dimensional representative features as a form of the Activation Feature Map (AFM).

The design of the CRFS is based on the equations defines in [71]. Given an data point $\{x^{(1)}, \dots, x^{(k)}\}$ with each $x^{(i)} \in \mathbb{R}$ where C is the number of classes. A set of the mean of the base features is defined as $y^{(i)} \in C = \mathbb{S} \cup \mathbb{T}$. The labeled target data points can be defined as $D_t = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m_t)}, y^{(m_t)})\}$ as in Equation 2.4 where m_t is the number of data points. Note that the data points can also be defined in D_s , that will be used in CRFS to understand both source and target domains (refer to Section 2.3.4).

$$D_t = \{x^{(i)}, y^{(i)}\}_{i=1}^{m_t} \quad (2.3)$$

$$\hat{a}(x^{(i)}) = \operatorname{argmin}_{a^{(i)}} \|x^{(i)} - \sum_j a_j^{(i)} b_j\|_2^2 + \beta \|a^{(i)}\|_1$$

$$\hat{D}_t = \{(\hat{a}(x^{(i)}), y^{(i)})\}_{i=1}^{m_t} \quad (2.4)$$

CRFS is created based on the base vectors $b = \{b_1, b_2, \dots, b_s\}$ with each $b_j \in R_n$. The base vector b is generated in the source environment using D_s . The activation $a = \{a^{(1)}, \dots, a^{(k)}\}$ with each $a^{(i)} \in R_s$ forms the semantic property of CRFS. Each dimension of an activation vector $a_j^{(i)}$ is the transformation of input $x_u^{(i)}$ using the base b_j . The number of bases s can be much larger than the input dimension n . The transformed target data points \hat{D}_t will be input for the Class Representative Generation.

Since each pair of $(x^{(i)}, y^{(i)})$ is independent of each other, our algorithm was designed with the CRCW (Concurrent Read Concurrent Write) model which allows parallel computing, including I/O, with the shared memory and processors. The CRL's operation time is proportional to the number of the selected CRs across all processors. The CR Generation will be proportional to the input set per CR independent of the number of classes in a given dataset. The CR-based inferencing will be proportional to the number of CRs in a given model.

2.3.3 CR Generation

Class Representatives (CR) are generated using the nearest prototype strategy by aggregating feature vectors. The nearest mean feature vector with instances of the given class, i.e., Class Representatives, is computed for each class. Specifically, the average

feature mean operation was used to summarize the instances of classes. For each class, the instances of each feature in the feature maps (e.g., 12K for a CNN layer) are aggregated into a simple mean feature in order to create its CR. The CR is an aggregated vector of the mean features for all the features in the feature maps.

For the Class Representative Generation, we considered the transformed Target Dataset \hat{D}_t as the input (as shown in Equation 2.3). As we emphasis on the parallelism and independence, we considered the individual activation vector $\hat{a}(x^{(i)})$ such that $y^i = c$, that will be used in formulating the CR as shown in Equation 2.5.

$$CR(c) = \{CR^1, CR^2, \dots, CR^n\}$$

$$CR^j = \frac{1}{N_c} \sum_{i=1}^{N_c} \hat{a}(x^{(i)}) \quad (2.5)$$

where j ranges from 1 to n representing the feature dimensions, c is the class of the input image, and N_c is the number of data points for the class c . Class Representative of the given class c is represented as the group of CR features values CR^j where j ranges from 1 to n feature dimensions. CR^j is generated from the mean of AFM (refer to Equation 2.3) of every input image in a given class c as shown in Equation 2.5. The CR Generation can be conducted in parallel so that each CR of class independently can be generated. The parallel algorithm with CRCW (as explained in Section 2.3.2) was implemented with Spark's broadcast variables for the CR Generation.

2.3.4 CR Feature Space: Source and Target Mapping

The Class Representative (CR) mapping is a variation of Multi-Discriminative Problem network [66]. This method is attempting to universalize a method that combines different but complementary features learned on different problems. The source domain problem is defined as the DP^s in the class when the Convolution Neural Network assigns to the input image the label corresponding to the classes provided by the source domain D . Similar to what is described in Pan et al. [37], we define our source and target domain based two aspects, namely Class Representative Feature Space (CRFS) and Marginal Distribution.

$$\begin{aligned} D_s &= CRFS_s, P(CR_s) \\ D_t &= CRFS_t, P(CR_t) \end{aligned} \tag{2.6}$$

As shown in Equation 2.6, the CR source domain D_s is a two-element tuple consisting of Source CR Feature Space $CRFS_s$ and Probability Distribution of CR $P(CR_s)$, where CR_s is Class Representatives from the source domain. The CR target domain D_t is also defined as a two-element tuple consisting of Target CR Feature Space $CRFS_t$ and

Probability Distribution of CR, $P(CR_t)$, where CR_t Class Representatives were generated from the target domain.

$$\begin{aligned}
 F_s(x) &= P[CR_s] \forall CR_s \in CRFS_s \\
 F_t(x) &= P[CR_t] \forall CR_t \in CRFS_t \\
 D^* &= \max_x |F_s(x) - F_t(x)|
 \end{aligned}
 \tag{2.7}$$

As shown in Equation 2.7, $F_s(x)$ and $F_t(x)$ are the distribution functions based on the probability distribution ($P(CR_s)$ and $P(CR_t)$) for the source and target domain CRs respectively. D^* shows the Kolmogorov-Smirnov distance between the source CR distribution and target CR distribution, i.e., it is computed as the max of distance between $F_s(x)$ and $F_t(x)$.

We use Two-Sample Kolmogorov-Smirnov Test (KS-Test) as a simple test for measuring the differences of the distributions of two sets, such as a sample and a reference probability distributions. Equation 2.7 computes the distance D^* between the medians of the Source Distribution $F_s(x)$ and Target Distribution $F_t(x)$. The distance D^* is the indicator that would be used to measure the CR similarity between $F_s(x)$ and $F_t(x)$. A larger distance D^* yields less accurate transfer learning for the target domain.

2.3.5 CR-based Inferencing

The CR-based inferencing is a mapping between the input and Class Representatives (CRs) and label it with a class. The CR-based inferencing can be done in parallel since the CRs are independent of each other.

$$\begin{aligned}
\cos(CR(c), NI) &= \frac{CR(c) \cdot NI}{\|CR(c)\| \|NI\|} \\
&= \frac{\sum_{i=1}^N x_{i,c} x_{i,ni}}{\sqrt{\sum_{i=1}^N x_{i,c}^2} \sqrt{\sum_{i=1}^N x_{i,ni}^2}}
\end{aligned} \tag{2.8}$$

Here are the steps for the CR-based inferencing. Given a new input is vectorized in the source environment to the Class Representative Feature Space (CRFS), $NI = \hat{a}(x^{(i)})$, as shown in Equation 2.3. The cosine similarity between the new input (NI) and Class Representatives for class c ($CR(c)$), where $c \in C$ can be computed using Equation 2.8. The CRL Model assigns the new input with the label associated with Class c that has the highest cosine similarity score. The higher cosine similarity score indicates the closeness between the Class Representative $CR(c)$ and the new input (NI) in the Class Representative Feature Space (CRFS).

$$\hat{c} = \operatorname{argmax}_{c \in C} \{\cos(CR(c), NI)\} \tag{2.9}$$

As shown in Equation 2.9, the label for the new input from CRL Model \hat{c} is predicted by selecting the class from all classes C that has the highest cosine similarity to the new input. The CRL model will conduct inferencing by matching the new input against the available CRs and label it with a class having the highest cosine similarity score.

2.4 CR Graph: Relationships between Class Representatives

The Class Representative Graph is a Graphical Representation of the Domain-based on CR-to-CR Similarity. The CR Graph is introduced to demonstrate the ability of

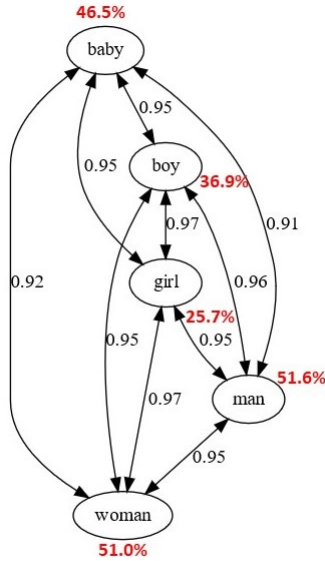


Figure 5: CR-Graph in CIFAR-100

Table 2: Class Representative Relationships in CIFAR-100

Baby	Boy	Girl	Man	Woman	CS_g	A_c	
Baby	1	0.95	0.95	0.91	0.92	0.918	46.5%
Boy	0.95	1	0.98	0.96	0.95	0.938	36.9%
Girl	0.95	0.98	1	0.95	0.97	0.94	25.7%
Man	0.91	0.96	0.95	1	0.95	0.754	51.6%
Woman	0.92	0.95	0.97	0.95	1	0.758	51%

the CRL model to understand the Target Domain’s Semantics. The CR Graph showcases excellent performing classes as well as wrong performing classes. In recent years, graph representation for image data has been used to identify mislabeled data [72], to create a community to improve classification [73] or to create a state-based ZSL Model [58]. Visual Trees are also used for hierarchical classification [74]. In this work, the goal of CR-Graph brings insights within domain-based individually as well as a group.

Definition 6: CR Graph

Class Representative Graph is defined as a graph *CR Graph* formed with nodes are classes (hosting Class Representatives with them), and the association between the Classes form the edges. A CR Graph G_c is defined as a set of vertex V and edges E , i.e., $G_c(V, E)$. The vertex V in G_c represents a class that is highly similar to other classes (thus, it is called *confusion class*), and the accuracy of the confusion class is lower than the overall accuracy of the model. The percentage of P in red is the classification accuracy of each class at G_c . The edge E represents the cosine similarity CS between two classes (C_i, C_j) .

Definition 7: CR Community

Class Representative Community is defined as the neighborhood $CRCom$ in *CR Graph* of a given class c . $CRCom$ is a set of CRs containing the members (i.e. classes) that are similar to each other forming a community. The community can be represented in the CR Graph that was introduced in our previous work [75]. In addition, the measurement *degree of similarity* is extended to find the relationships between CRs within the community or across the communities and estimate the accuracy performance of the CRL model.

The cosine similarity CS_i of all edges E in the CR Graph G_c is bigger than the threshold (δ), i.e., $CS(C_i, C_j) > \delta$. Figure 5 shows the CR Graph G_c with the threshold $\delta = 0.85$. Each edge E is bi-directed since the cosine similarity between any two confusion classes is symmetric, $CS(C_i, C_j) = CS(C_j, C_i)$. Figure 5 shows a CR Graph, which represents a CR Community including *baby, boy, girl, man, and woman* that are confusion classes randomly selected from the CIFAR-100 dataset.

The group cosine similarity of class C is computed as follows:

$$GCS(C) = \frac{\sum_i CS(C_i)}{K} \quad (2.10)$$

where K is the number of the confusion classes excluding self in the CR Graph G_c . In Figure 5, the total number of confusion classes is 6 and the number of the neighborhood classes for *girl* $K = 6 - 1 = 5$. The group cosine similarity of *girl* is $CS_g(\text{girl}) = \frac{(0.95+0.98+0.95+0.97+0.85)}{5} = 0.94$. The accuracy of the girl class ($A_{\text{girl}} = 25.7\%$). The $CS_g(\text{girl})$ is the highest similarity while its accuracy A_{girl} is the lowest accuracy. The group cosine similarity of *baby* is $CS_g(\text{baby}) = \frac{(0.95+0.95+0.91+0.92+0.86)}{5} = 0.91846$ while the accuracy is $A_{\text{baby}} = 46.5\%$.

Definition 8: *Class Representative Superstar*

Class Representative Superstar (CR-SS) is a class with the CR having a high individual accuracy as well as being less confused with other classes. CR-SS contributes to high performance in classification; unlike *Troublemaker*, especially they are less similar to other CRs in the CR Graph.

$$CR - SS(c_i) = CS_g(c_i) \leq \alpha_g \ \& \ A(c_i) \geq \delta_d \quad (2.11)$$

Definition 9: *Class Representative Troublemaker*

Class Representative Troublemaker (CR-TM) is a class with the CR having a low individual accuracy as well as being highly confused with other classes. Troublemaker causes low performance in classification, especially due to the high similarity with other CRs in

the CR Graph. Let us define $c_i \in C$, α_g is the threshold of cosine similarity of the group (g), and δ_d is an average accuracy of the domain (d).

$$CR - TM(c_i) = CS_g(c_i) > \alpha_g \ \& \ A(c_i) < \delta_d \quad (2.12)$$

The accuracy of these classes in this CR Graph is lower than the overall accuracy of the dataset (for example, CIFAR-100, 58%). The group cosine similarity of class $CS_g(C)$ is inversely related to its accuracy (A_c). As shown in Table 2, the candidates for the confusion classes include *boy*, *girl*, *man*, and *woman*. They meet the condition for confusion classes. Thus, the confusion classes of *baby* include *boy*, *girl*, *man*, and *woman*. The CS threshold (α_g) is set as α_g (for example, α_g is defined 0.9 through experiments).

2.5 Implementation and Experimental Design

The experiments on the Class Representative Learning (CRL) model have been conducted on ImageNet-1K as a source domain and CIFAR-100, CalTech-101, and CalTech-256 as a target domain. The *source* environment, i.e. the pre-trained model from the source dataset (ImageNet-1K), with three different deep learning networks, such as Inception-V3 [76], ResNet-101 [77], and VGG-19 [78].

2.5.1 Implementation

2.5.1.1 System and Library Specifications

The Feature extraction was implemented on a single GPU, which is Nvidia GeForce GTX 1080 (with 12GB GDDR5X RAM) on MATLAB 2018b version. The CR Generation and CR-based Inferencing were implemented using Spark 2.4.3 version [79]. The parallel and batch process was conducted through RDD based parallelism on a single CPU with 4GHz Intel Core i7-6700K (quad-core, 8MB cache, up to 4.2GHz with Turbo Boost) and 32GB DDR4 RAM (2,133MHz) (i.e., local parallelism of 4 cores).

2.5.1.2 Models Specification

As described in Section 2.3.2, the *source* environment provides the feature space for the CRL model. The Inception-V3 model was predominantly used as the source environment (pre-trained with ImageNet-1K) for the CRL experiments. The Inception-V3 model was obtained from MATLAB's Pre-trained Deep Neural Networks [80]. The layer information of the Inception-V3 model is shown in Figure 6. The feature extraction has been conducted class by class as a form of parallel processing to build a CR for each class. For some experiments, ResNet-101 and VGG-19 extracted from MATLAB were also used as our source environments. The last convolution layer from the source environments was considered for Feature Extraction. The CR Generation was implemented in parallel with Spark's Resilient Distributed Datasets (RDDs), which is a collection of features partitioned across the nodes of the cluster. The batch in this context was defined while keeping CR independence of each class for the CR Generation and CR-based

Inference.

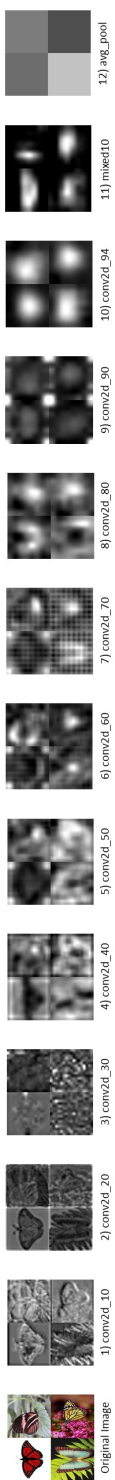
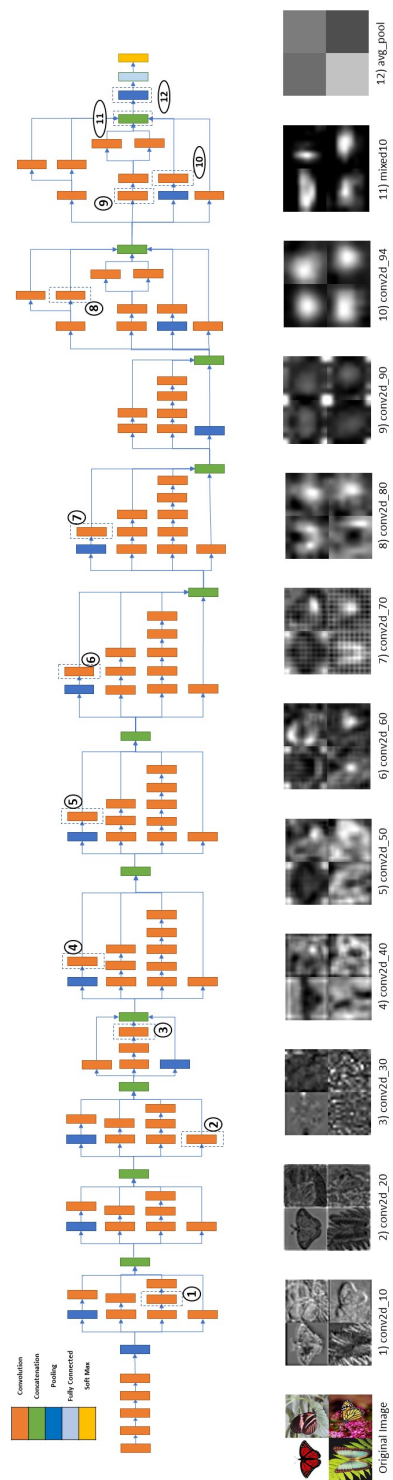


Figure 6: Inception-V3 Activation Visualization Layerwise

2.5.2 Datasets

We have conducted the experiments with the CRL model using four datasets according to the three transfer learning types defined in Day et al. [81], i.e., Homogeneous Transfer Learning (HOTL), Heterogeneous Transfer Learning (HETL), and Negative Transfer Learning (NTL). The four datasets include ImageNet-1K, CalTech-101, CalTech-256, and CIFAR-100 (as shown in Table 3). Figure 7 shows the four datasets that were used for the CRL’s transfer learning [82].

The source environment is built on the ImageNet-1K dataset that is the Homogeneous Transfer Learning (HOTL) type (the same label set and the same attributes). The transfer learning with CalTech-101 and CalTech-256 are the Heterogeneous Transfer Learning (HETL) type, which was projected on the semantic space of the source domain with minimal distinction classes. The transfer learning with CIFAR-100 is the Negative Transfer Learning (NTL) type since the target domain data are projected on the semantic space that is quite distinct from the source domain. Although the CIFAR-100 is semantically relevant to other datasets, the CRL space of CIFAR-100 is divergent from the source space in terms of image modality, such as image quality and image size. The size of CIFAR-100 images is [32x32] while one of the source domain ImageNet-1K [400x400]. More specifically, the dimension of the source environment of Inception-V3 is [299x299]. In the experiment section, the details on the performance of these different transfer learning types will be discussed.

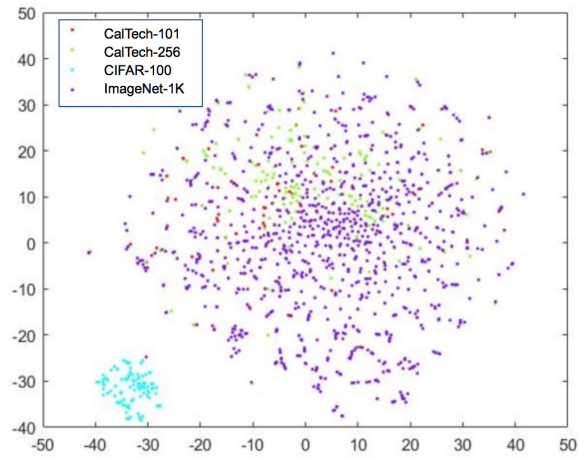


Figure 7: t-SNE Visualization of Class Representatives[82]

Table 3: Benchmark Datasets

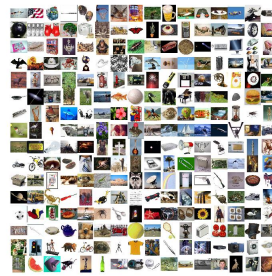
Domain	Dataset	#Class	#Image	Image Size
Source	ImageNet-1K	1000	1,281,167	400x400
Target	CalTech-101	101	8,677	300x300
	CalTech-256	256	30,608	300x300
	CIFAR-100	100	59,917	32x32

2.5.3 Experiments for Transfer Learning Performance

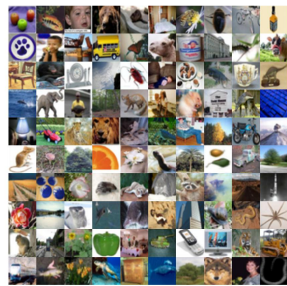
Transfer Learning Performance in terms of Accuracy and Time or Space Requirements: state-of-the-art Transfer Learning vs. CR-based Classification with different datasets (CalTech-101, CalTech-256, ImageNet-1K, ImageNet-20K, CIFAR-100).



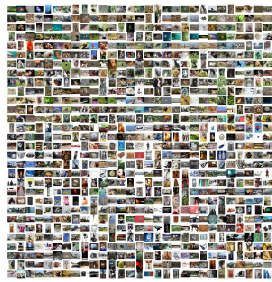
CalTech-101



CalTech-256



CIFAR-100



ImageNet-1K

Figure 8: Benchmark Datasets: CalTech-101 & CalTech-256 & CIFAR-100 & ImageNet-1K

Table 4: Source Domain and Target Domain: Cosine Similarity Median, Accuracy, Kolmogorov-Smirnov Test Scores

*CS: Cosine Similarity. A high CS Distance ($S_m - T_m$) and high KS Test Score indicate Negative Transfer Learning

Source Domain			Target Domain				Source & Target Comparison		
Dataset	CS Median (S_m)	Accuracy (S_a)	Dataset	CS Median (T_m)	Accuracy (T_a)	State-of-the-art Accuracy (S_{A_a})	CS Distance ($S_m - T_m$)	KS Score	Test Accuracy Change ($S_{A_a} - T_a$)
ImageNet-1K	0.434	73.9%	CalTech-101	0.398	93.9%	92.98%	0.036	0.1570	+1%
ImageNet-1K	0.434	73.9%	CalTech-256	0.454	77.9%	77.6%	0.02	0.0921	+0.3%
ImageNet-1K	0.434	73.9%	CIFAR-100	0.734	57.9%	76.2%	0.3	0.9125	-18.3%
ImageNet-1K	0.434	73.9%	ImageNet-1K	0.434	73.9%	78.8%	0	0	-4.9%

Table 5: Class Representative Distribution Statistics

	CIFAR-100		CalTech-101		CalTech-256		ImageNet-1K	
	Accuracy	GCS	Accuracy	GCS	Accuracy	GCS	Accuracy	GCS
Mean Accuracy	57.9%	0.74	88.8%	0.41	76.0%	0.46	73.8%	0.43
Std dev	15.2%	0.09	9.7%	0.13	15.6%	0.11	12.9%	0.10
Range	[17.7%, 86.4%]	[0.44, 0.98]	[55.6%, 100%]	[0.03, 0.92]	[29.1%, 100%]	[0.08, 0.96]	[28.6%, 96.5%]	[0, 0.99]

GCS: Group Cosine Similarity

- Case 1: Source Domain (\mathbb{S}): ImageNet-1K, Target Domain (\mathbb{T}): CalTech-101 or CalTech-256: The distance between the medians of both domains, SD and TD, is very small (i.e., $D^* \leq 0.05$). In this case, the classification with the class representatives (CV), which are generated from the pre-trained model in SD with small data in TD, are as effective as the state-of-the-art models.
- Case 2: Source Domain (\mathbb{S}): ImageNet-1K, Target Domain (\mathbb{T}): CIFAR-100: The distance between the medians of both domains, \mathbb{S} and TD, is very big (i.e., $D^* \geq 0.3$) as well as the size of the data in TD is small.

Two-Sample Kolmogorov-Smirnov Test is used to determine whether the instances of any given class are distributed within a class. The class distribution would also be applied to determine if there are any data issues such as data labeling errors or noise data. Thus, we could estimate the class accuracy using the class distribution model even before the training. Figure 9 and Table 4 shows the KS-Test results between the distribution of the source and target datasets. If the KS-Test value is high, then the source model may not be suitable for the target domain.

Figure 9 demonstrates the similarity distribution in the feature space of the source and target datasets as well as their accuracy distributions. Accuracy distribution represents the histogram of class accuracy in a given dataset while using CR based classification. Cosine similarity distribution represents the cosine similarity between a CR Pair. Higher cosine similarity means the similarity between CRs is high. The cosine similarity distribution is using to compare the source dataset to the target dataset. The comparison between CIFAR-100 and ImageNet-1K has the highest KS-Test value among the four

different datasets.

The CRL model is used to understand the distribution of datasets and their performance. It also showcased the overall group cosine similarity (refer to Section 2.4 and Equation 2.10). Table 5 shows the CR distribution statistics for the source and target domains in terms of their accuracy and group cosine similarity (GCS). The results based on the CR-Inception-V3 as seen from Table 5 are consistent with Figure 7, t-SNE Visualization shows that CalTech-101 and CalTech-256 are overlapped with ImageNet-1K (source domain), while CIFAR-100 is in a long distance from the source domain. The CalTech-101 shows the best mean accuracy and low cosine similarity. However, CIFAR-100 is limited by low mean accuracy and high cosine similarity. For the CIFAR-100 dataset, the accuracy of 57.9% is the least, and the group cosine similarity of 0.74 is the highest compared to the ones for all other datasets. The salient reason for the low accuracy of the CIFAR-100 dataset is mainly due to high cosine similarity and a huge distance from the source domain. In summary, among the four datasets, CIFAR-100 performs the worst, and CalTech-101 performs the best.

2.5.4 Classification Performance with Benchmark Datasets

In this chapter, the CRL model has been validated in terms of CR Feature Exaction and CR Generation as follows: i) Feature extraction in terms of CNN Network models (Inception-V3, ResNet-101) and CNN layers, ii) Feature representation and optimization such as (12K vs. 3K feature vector) and the number of training images (20, 30, 60, 100, and All). For most of the evaluation, CR-Inception-V3 version was considered.

2.5.4.1 Results for Architecture Selection in Feature Extraction

The CRs are mainly dependent on the quality of the features extracted from the pre-trained CNN model. The two popular pre-trained models such as Inception-V3 [76] and ResNet-101 [77] were used as the CR source environments and their performance was compared in Table 6. We also evaluated to select the most suitable layer in these pre-trained models. For both the pre-trained models, we compared CRL with the original accuracy, as shown in the state-of-the-art approaches [76, 77, 83, 84, 85].

The accuracy for the datasets reported here is for the Top-1 accuracy of the model. Comparing the CRL model to the original model, it was observed that CalTech-101 has an increase of 1.56% in both the models. There was a significant decrease in the accuracy of The CIFAR-100 for both the models. This is likely due to the greater distance in a semantic space between the source domain (ImageNet-1K) and the target domain (CIFAR-100), as discussed in Section 2.5.2. In the end, for the overall comparison of the models, the accuracies of the Homogeneous Transfer Learning (HTL) in Inception-V3 are better than the ones in ResNet-101. This comparison leads to the use of Inception-V3 as the source environment of the CRL model.

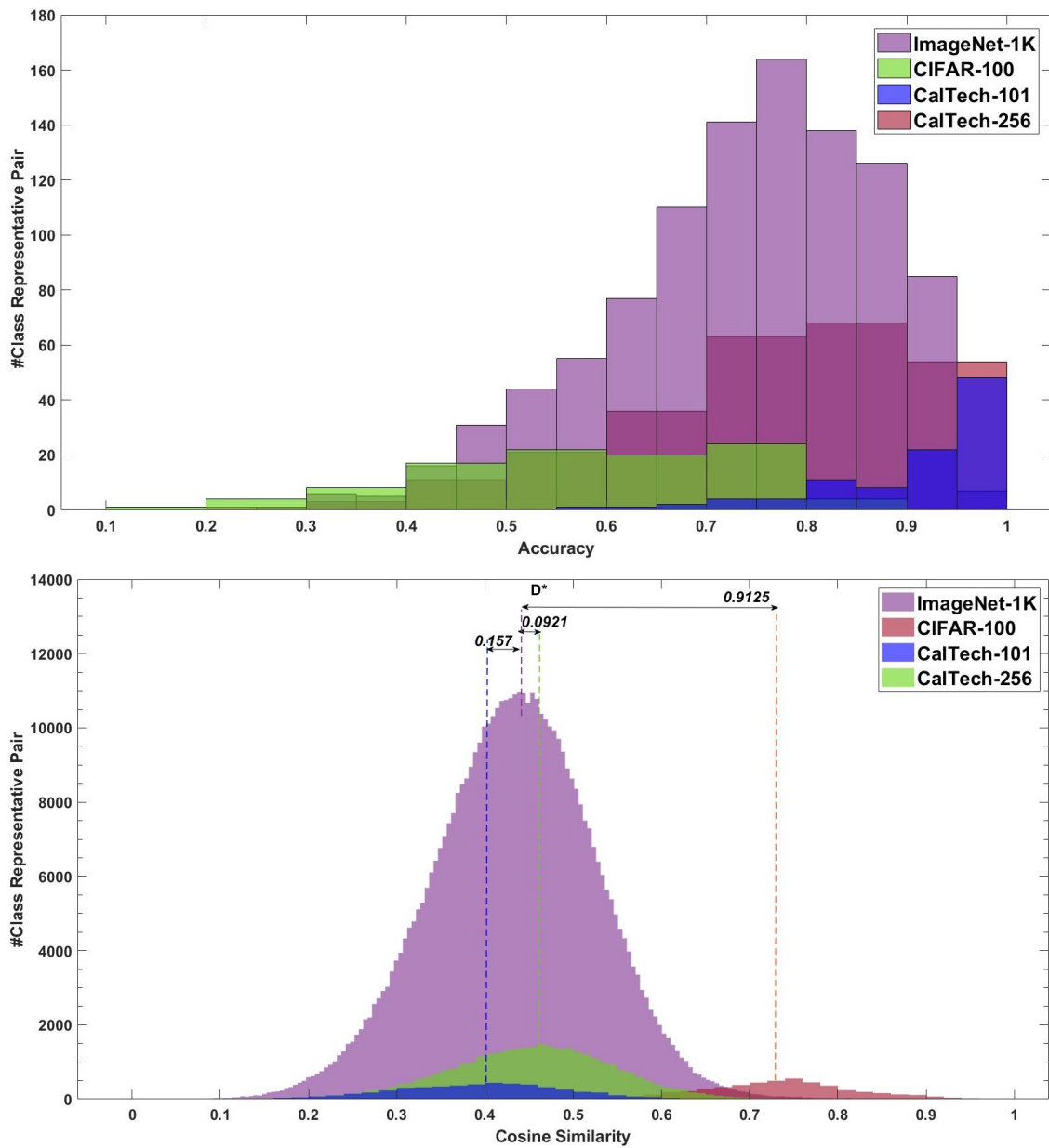


Figure 9: Accuracy Distribution and Cosine Similarity Distribution

Table 6: CRL Models based on Deep Learning Networks (Inception-V3 vs. ResNet-101)

Base Model	Inception-V3		ResNet-101	
	CRL Model	State-of-the-Art Model [76]	CRL Model	State-of-the-Art Model [77]
Parameter	0	21.8M	0	22.44M [83]
Feature	3072	2048	100352	2048
Shape	AvgPool([8x8x192])	linear [1x1x2048]	[7x7x2048]	linear [1x1x2048]
Layer	conv2d_94	48 layers deep (347 layers)	res5c_branch2c	101 layers deep (316 layers)
ImageNet-1K	74.07%	78.8%	68.4%	77.56% [83]
CalTech-101	93.69%	92.88%	93.46%	91.4% [84]
CalTech-256	77.78%	77.6%	76.76%	80.1% [84]
CIFAR-100	57.63%	76.2%	63.02%	72.77% [85]

2.5.4.2 Results for Layer Selection in Feature Extraction

From Inception-V3, the most suitable layer for the Feature Extraction was identified by the layer-wise experiment. As shown in Table 7, the accuracy evaluation was conducted with the models built using the features from the selected layer. For this comparison, the twelve layers (including ten different convolution layers, final concatenated convolution layer, and final average pooling layer) were considered. These layers are indexed, as shown in Figure 6.

The best layer was determined in terms of the feature size and the accuracy of the model. Table 7 shows the feature size, flatten feature size, and accuracy. For this evaluation, comparable datasets are considered to evaluate the effectiveness of the CR-based classification. The Homogeneous Transfer Learning (HTL) was used to conduct non-biased feature analysis and layer selection. Layer 10 (conv2d-94) shows the best accuracy in CalTech-101, while Layer 12 (AvgPool) shows the best accuracy in CalTech-256. For the flatten feature size, the feature set of Layer 10 shows the highest accuracy compare to other layers.

2.5.4.3 Results from Feature Reduction

Once the CR feature map is generated, the CRL model might be required to compress it for mobile deployment. In this chapter, we have applied three different sampling techniques for the model compression: (1) Max Pooling, (2) Average Pooling, and (3) Min Pooling. The Max Pooling is a sample-based discretization technique that is widely used in Deep Learning. The objective of the Max Pooling is to reduce the feature map's

dimensionality by applying the max operation to features contained in the sub-regions of the feature map. The initial input of the CR feature map, such as $X * X$ matrix (e.g., $8*8$), will produce to $Y * Y$ matrix (e.g., $4*4$) using a $Z*Z$ filter (e.g., $2*2$). A stride of S (e.g., 2) controls how the filter operates around the input matrix by shifting S units at a time without any overlap regions.

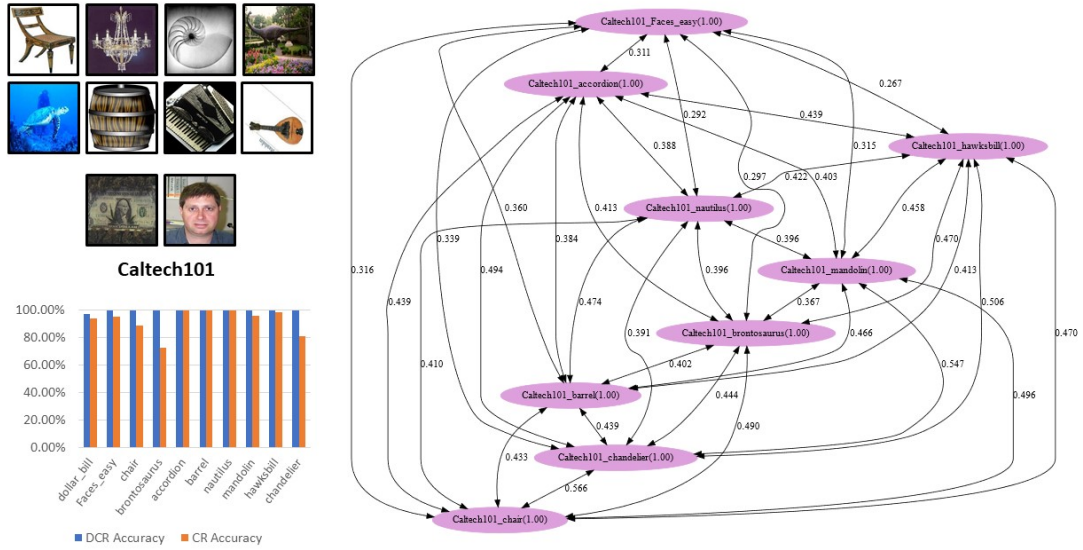
For each of the regions represented by the filter, the max of that region is computed to create the output feature map, in which each element is the max of a region in the original input. The Average Pooling and Min Pooling are very similar to the Max Pooling; the only difference is to utilize a different operation such as average and min operations for the feature map reduction.

Table 7: Inception-V3 Layerwise Accuracy on Similar Domain Datasets

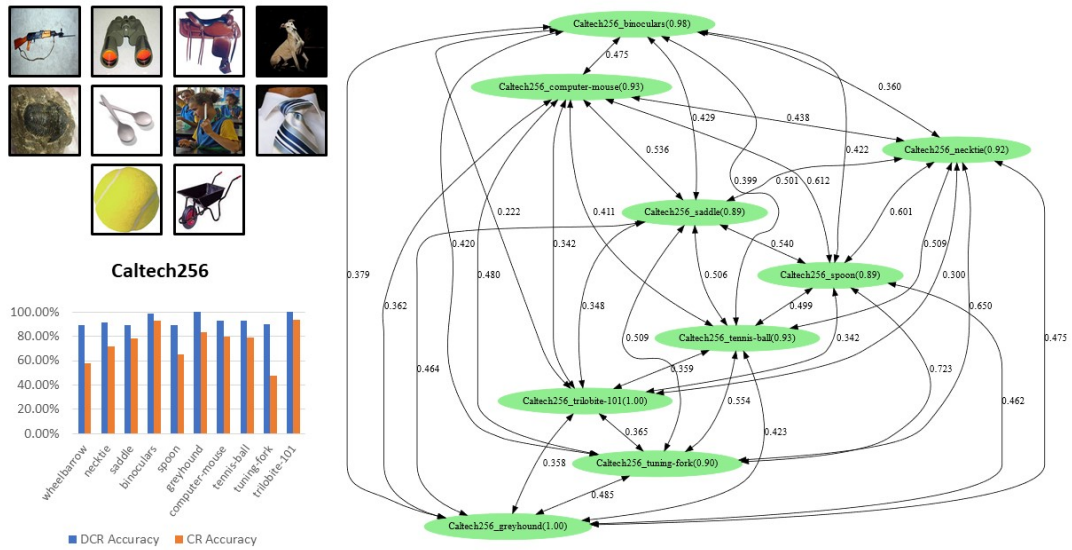
Num	Layer	Filter Size	Activation Shape	Activation Size	CalTech-101	CalTech-256
1	conv2d_10	3x3x64	35x35x96	117601	59.9%	24.0%
2	conv2d_20	1x1x288	35x35x64	78401	52.7%	23.7%
3	conv2d_30	3x3x96	17x17x96	27745	71.3%	32.0%
4	conv2d_40	1x1x768	17x17x192	55489	79.0%	44.1%
5	conv2d_50	1x1x768	17x17x192	55489	79.7%	47.1%
6	conv2d_60	1x1x768	17x17x192	55489	83.8%	52.2%
7	conv2d_70	1x1x768	17x17x192	55489	81.1%	49.5%
8	conv2d_80	3x1x384	8x8x384	24577	93.3%	74.2%
9	conv2d_90	1x1x2048	8x8x448	28673	85.3%	62.1%
10	conv2d_94	1x1x2048	8x8x192	12289	94.4%	78.2%
11	mixed10	-	8x8x2048	131073	91.9%	77.9%
12	avg-pool	8x8 [pooling]	1x1x2048	2049	90.0%	79.1%

Considering the feature size of the Layer 10 in Inception-V3, we evaluate by reducing the dimensions using standard reduction techniques, namely minimum pooling (MinPool), maximum pooling (MaxPool), and average pooling (AvgPool). The pooling in CBL was implemented on the [8x8,192] feature vector (Layer 10) with the filter size of 2x2 transforming into [4x4,192]. In CR-Inception-V3C, we applied AvgPool with a filter size [2x2] to the feature map of [8x8x192] extracted from Layer 10 and obtained the reduced feature map of [4x4x192]. As shown in Figure 14, the average pooling layer reduction is applied to the post-processing of CR-Generation. The same filter size was used for MaxPool and MinPool.

Table 8 shows the accuracy for CR-Inception-V3C by using the filter size based on the three pooling techniques. Based on the analysis with all of the datasets, CR-Inception-AvgPool showed the accuracy drop with an average of 1% in Top-1 accuracy when comparing with 12K or the original Layer 10 Accuracy. The interesting observation through this evaluation was the 12K CRL model's Top-5 Accuracy outperformed on all datasets compared with other available models.

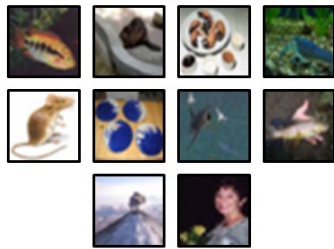


CalTech-101: Images, Accuracy, CR Graph

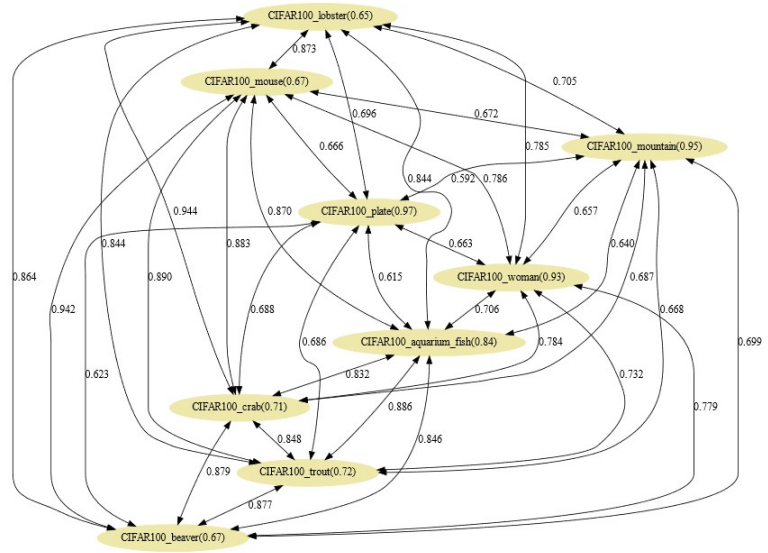
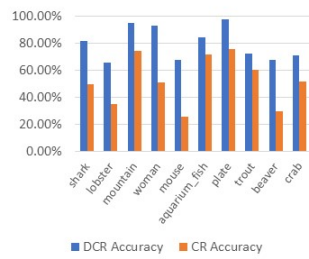


CalTech-256: Images, Accuracy, CR Graph

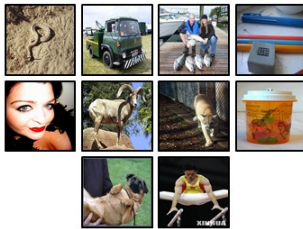
Figure 10: Dynamic Models: CalTech-101 & CalTech-256



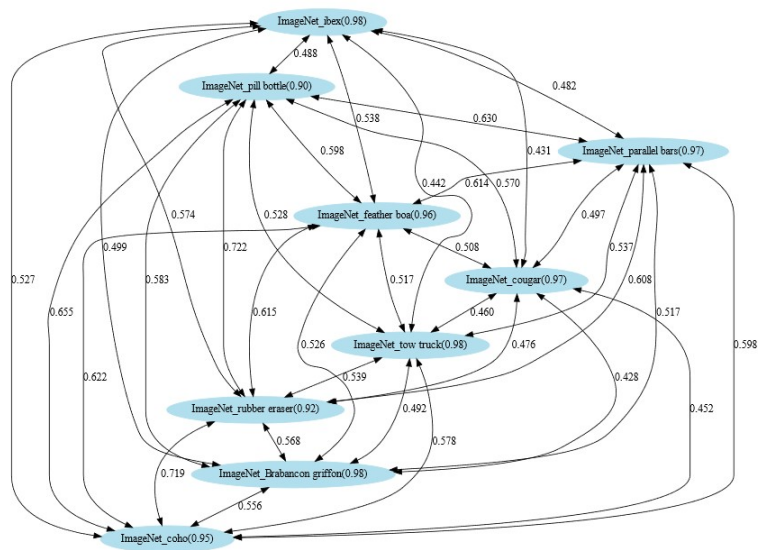
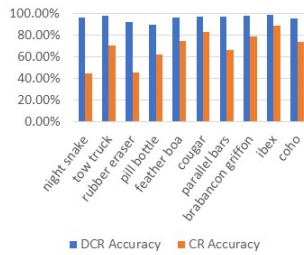
CIFAR100



CIFAR-100: Images, Accuracy, CR Graph



ImageNet



ImageNet: Images, Accuracy, CR Graph

Figure 11: Dynamic Models: CIFAR-100 and ImageNet-1K

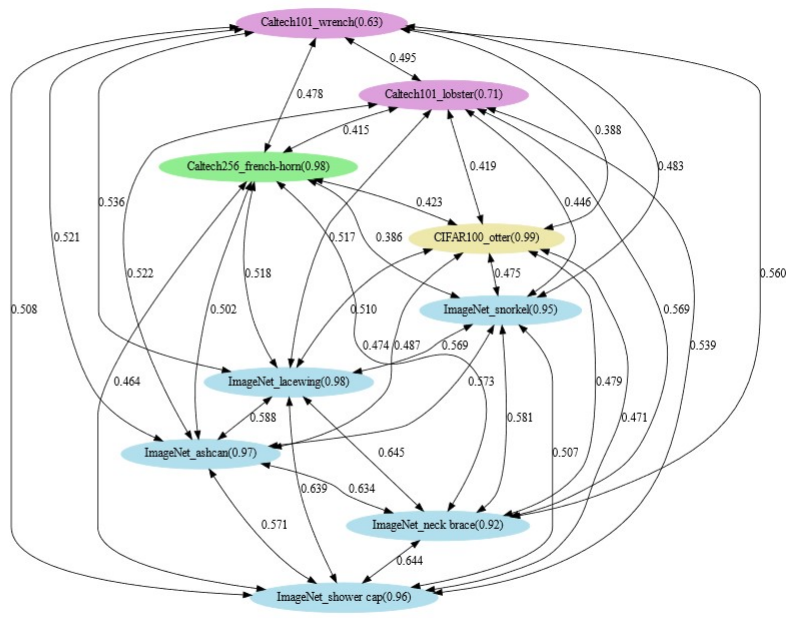


Figure 12: Dynamic Model: Mixed 4 Datasets - ImageNet, CalTech-101, CalTech-256, CIFAR-100

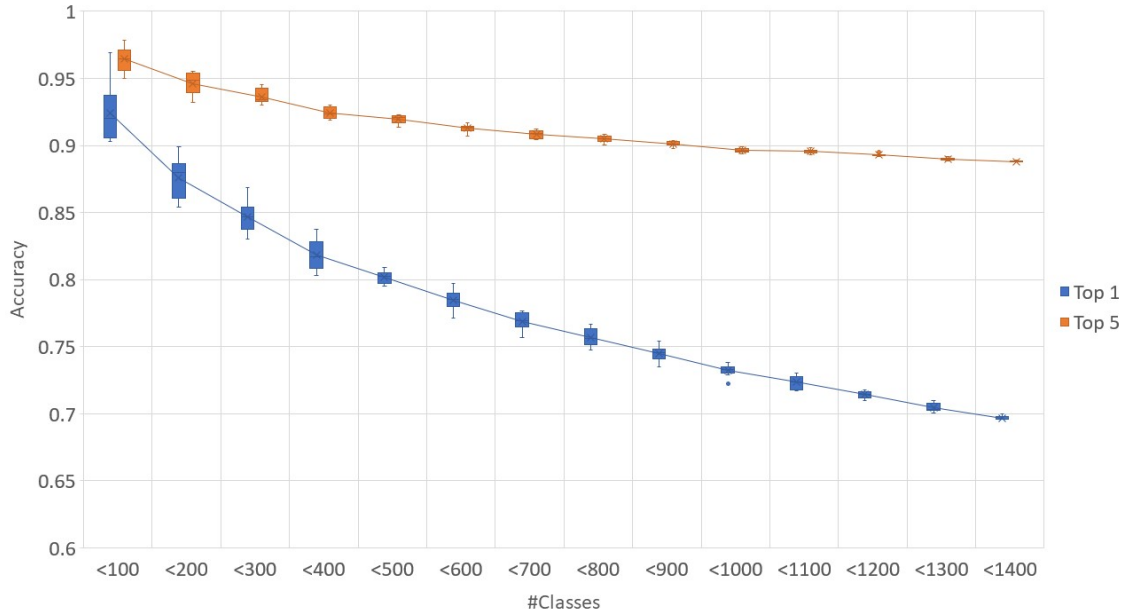


Figure 13: Accuracy with Increasing Class# in Dynamic Models

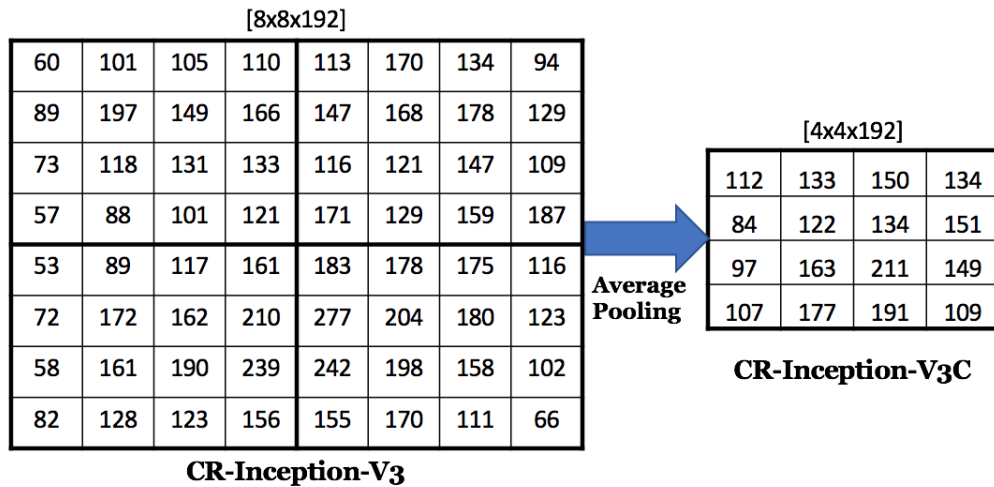


Figure 14: Reduction using Average Pooling

8X8 sized 192 channels [8x8x192] are reduced to 4x4 sized 192 channels [4x4x192]

Table 8: Feature Pooling and Top Accuracy

12K: Original; AVP: Average Pooling; MIP: Min Pooling; MAP: Max Pooling

	ImageNet-1K				CalTech-101				CalTech-256				CIFAR-100			
	12K	AVP	MIP	MAP	12K	AVP	MIP	MAP	12K	AVP	MIP	MAP	12K	AVP	MIP	MAP
Feature#	12288	3072	3072	3072	12288	3072	3072	3072	12288	3072	3072	3072	12288	3072	3072	3072
Top 1	73.93	74.07	72.47	70.12	93.96	93.69	93.46	91.89	77.87	77.78	74.83	71.14	57.96	57.63	55.76	55.25
Top 2	83.7	83.89	81.92	79.73	97.23	97.23	97.08	96.04	84.59	84.6	81.75	78.31	70.61	70.38	68.73	67.95
Top 5	93.31	90.56	88.41	86.59	99.15	98.73	98.42	97.89	92.43	89.83	86.99	84.37	90.5	83.42	81.89	81.48

2.5.4.4 Results from Data Imbalance Experiments

Most of the deep learning suffered from the data imbalance problem. Our experiments show that CRL is not sensitive to the data imbalance issue. Since CalTech-101 and CalTech-256 are imbalanced datasets, we have conducted experiments to show their classification performance is independent of the number of input.

In order to analyze the imbalanced data issue in the CRL model, we have evaluated the CR Generation with a varying number of images and accuracy. Table 9 shows the CR Generation accuracy for the image sets of 20, 30, 60, 100, and all. All represented in the training dataset (i.e., 70%) for any given class. The results show that the accuracy of CR-based classification does not vary significantly for a varying number of images. This effect is clearly shown with the imbalanced datasets such as CalTech-101 and CalTech-256. For example, CalTech-101, Class *airplanes* have 560 images with a class accuracy of 97.9% while *camera* has 35 images with a class accuracy of 96.8%. This indicates that the image imbalance problem does not affect the classification accuracy in the CRL model.

Table 9: Image Classification Accuracy for an Increasing Image#

Dataset	Image Count & Accuracy				
	20	30	60	100	All
ImageNet-1K	69.5%	70.9%	72.5%	73.0%	73.9%
CalTech-101	91.9%	93.5%	93.6%	93.8%	93.9%
CalTech-256	74.4%	75.8%	77.3%	77.8%	77.9%
CIFAR-100	50.9%	53.4%	55.9%	56.8%	57.9%

2.5.5 Dynamic Models

For evaluating the functionality of dynamic modeling that is capable of the class representative is demonstrated by a random selection of 10 classes from the four datasets separately, and 10 classes selection from four datasets combined, as shown in Figure 10-13 (refer to Section 2.2). Table 10 shows the random selection of classes from 10 to 100 from different individual datasets and all together.

Table 10: Dynamic Model Accuracy (Random Generation)

#Class	CIFAR-100	CalTech-101	CalTech-256	ImageNet-1K	Mixed Models (4 Datasets)
10	79.3%	99.7%	93.6%	95.8%	95.9%
20	79.7%	97.9%	90.0%	95.9%	96.3%
30	74.1%	95.9%	90.9%	94.6%	96.1%
40	63.9%	95.3%	90.6%	93.6%	89.6%
50	69.5%	95.6%	84.3%	91.3%	92.7%
60	66.2%	95.9%	87.7%	91.9%	91.0%
70	61.5%	95.4%	87.8%	92.7%	91.1%
80	60.9%	94.1%	87.0%	91.3%	92.1%
90	58.1%	94.0%	84.1%	92.0%	91.2%
100	58.0%	93.4%	85.4%	87.6%	90.1%

2.5.6 Model Performance Comparison

The CRL model’s performance is evaluated by comparing with Inception-V3 pre-trained model retrained with the target domain. The CRL model’s performance is calculated based on AFM Generation Time (refer to Section 2.3.2), CR-based Inference Time (refer to Section 2.3.5) and CR Model Generation Time (refer to Section 2.3.3).

Table 11 shows the comparison of the CRL model’s overall time vs. the time

taken for retraining the dataset using the Inception-V3 pre-trained model. The pre-trained model was run on the same system specification as the CRL model. Pre-training of the Inception-V3 Model was stopped at a reported number of epochs as the time taken was significantly higher than that of the CRL model. The CRL model with three datasets, (CalTech-101, CalTech-256, and CIFAR-100) have an average of 99% time reduction that is a significantly reduced time compared with that for the original Inception-V3 model. Within the same time window and based on the same pre-trained model, the Inception-V3 model performance has not reached the accuracy published in [33]. The CRL model's overall time shows genuinely outstanding performances in the target domains, even if the models never learned from the target domain.

Table 11: Transfer Learning Performance Analysis: CRL vs. Inception-V3 [80]
 Pretrained Model: Inception-V3 with ImageNet-1K

Domain		CRL						Inception-V3 [80]		
Source	Target	Step	Time	Overall Time	Accuracy	Time	Epoch	Accuracy		
ImageNet-1K	CalTech-101	AFM Generation	5m 21s	7m 12s	94.40%	4257m 55s	40 Epochs	88.73%		
		CR Model Generation	1m 31s							
		CR Inferencing	20s							
	CalTech-256	AFM Generation	10m 14s	13m 53s	78.20%	14193m 58s	14 Epochs	59.26%		
		CR Model Generation	1m 54s							
		CR Inferencing	1m 45s							
CIFAR-100	AFM Generation	13m 12s	15m 44s	57.96%	26941m 51s	10 Epochs	50.30%			
	CR Model Generation	56s								
	CR Inferencing	1m 36s								

2.5.7 Comparison with Lightweight Classification Models

The two CRL models, namely CR-Inception-V3 [based on 12K Layer 10] (refer to Figure 6) and CR-Inception-V3C [based on AvgPool] (refer to Figure 14) were considered for evaluation. The CRL models are compared with the state-of-the-art mobile Deep Learning models such as Mobile-Net-v1 [45], Mobile-Net-v2 [21] and NasNet-Mobile [22]. The Inception-V3 Model accuracy is also compared as the CRL model is based on Inception-V3 (refer to Section 2.5.4). The state-of-the-art accuracies shown in 12 were based on the work by Kornblith et al. [33]. Another work by Kornblith et al. [86] was based on the performance of checkpoints from TensorFlow-Slim repository.

The following a brief review of Light-Weight Classification models that are compared with CRL.

- **MobileNet-v1** MobileNet-v1 is one of pioneer work in light-weight convolution neural network. An efficient low latency model is achieved using Depthwise Convolution Filters [45]
- **MobileNet-v2** MobileNet-v2 is extention of MobileNet-v1, they improve light-weight model using inverted residual module with linear bottleneck. MobileNet-v2 1.4 is a version with neural network image input size of 224x224 and multipliers set to 1.4 [21].
- **NASNet-A-Mobile** NASNet is Convolution Neural Network where the Semantic Space is transfered from smaller dataset to bigger dataset using a Reinforcement Learning Search Method called Neural Architecture Search (NAS) [22].

The comparison between the mobile models and CRL is conducted in terms of the computational cost, model size, and accuracy. The computation cost is usually defined based on floating-point operations (FLOPs) and parameters. The FLOPs of the CRL model are less than the one for prediction with the base model, i.e., Inception-V3. As shown in Table 12, the number of parameters for the CRL model is Nil as there is no traditional learning component in the CRL model. The CRL model's computational cost outperforms all the other models.

The CRL model's size is given based on the size of each CRL of class. The significant difference between the traditional deep learning model and the CRL model is that the model size is dependent on the number of classes rather than the number of layers or size of the layer. The size as listed in Table 12, includes two parts the size of the pre-trained model plus the per CR per class size, i.e., 0.15MB for CR-Inception-V3 and 0.06MB for CR-Inception-V3C.

In Table 12, the two CRL models were compared with other mobile models. These models outperform the existing mobile models. Also, these models perform better than the original Inception model with CalTech-101 and CalTech-256 datasets. Their performances are reasonably comparable to the original ones with ImageNet-1K Dataset. However, the CRL models do not perform well on the CIFAR-100 Dataset. Overall, the CRL models are better than state-of-the-art mobile models if the target domains are similar to the source models. Otherwise, as seen from the CIFAR-100 model, the performance did not meet expectations. It is because there is a considerable gap between the source and target domains, and this gap may result from the lack of learning in the target domain.

This result confirms that the CRL model can be used to validate the distribution of data in terms of dissimilarities and similarities of CRs. The classification accuracy can be estimated based on the CR distribution model. Furthermore, outliers of data can be normalized, or mislabeled images can be detected with a CR.

Table 12: Comparison of Class Representative with Pre-trained Models([33], [87])

Model	Parameters ^a	Feature Size	Image Size	Model Size	Accuracy			
					ImageNet-1K	CalTech256	CalTech101	CIFAR100
MobileNet-V1 [45]	3.2M	1024	224	16MB	70.6%	N/A	90.7%	70.9%
MobileNet-V2 [21]	2.2M	1280	224	13MB	72%	N/A	91.26%	70.6%
MobileNet-V2 (1.4) [21]	4.3M	1792	224	24MB	74.7%	N/A	91.83%	73.4%
NASNet-A Mobile [22]	4.2M	1056	224	20MB	74%	N/A	91.52%	73.6%
Inception-V3 [76]	21.8M	2048	299	89MB	78.8%	N/A	92.98%	76.2%
CR-Inception-V3 (ours)	N/A	12288	299	0.15MB/CR*	73.93%	77.87%	93.96%	57.96%
CR-Inception-V3C (ours)	N/A	3072	299	0.06MB/CR*	74.07%	77.78%	93.69%	57.63%

* CR-Inception-V3 uses Inception-V3 as its pre-trained model which is 89MB.

2.5.8 Comparison with Zero-Shot Learning Algorithms

In this section, we evaluate the CRL model using three different evaluations; Recognition Task-based Accuracy, Accuracy with an increasing number of instances of the unseen dataset, and comparison with state-of-the-art Zero-Shot Learning (ZSL) approach. For this section, we consider two versions of the CRL model; (i) Inception-V3 based and (ii) VGG-19 based model. In Table 13, the performance of Inception-V3 model-based CRL (CR-Inception-V3) in the ZSL perspectives was presented.

The CRL model is capable of recognizing the target labels (unseen data) without having the source labels (seen data) as an option. This shows the advantage and ability as a classification model (see Section 2.5.7). Table 14 shows two versions of the recognition tasks with testing data from target set (\mathbb{T}); $\mathbb{T} \Rightarrow \mathbb{T}$ when the testing label could be only from the target set $y^* \in \mathbb{T}$ and $\mathbb{T} \Rightarrow \mathbb{S} \cup \mathbb{T}$ when the testing label could be from both the source set and target set $y^* \in \mathbb{S} \cup \mathbb{T}$. For this experiment, we consider all instances (70% of the dataset) from the dataset to generate CRs. The increase in the number of labels in the dataset was compared with the accuracy. There were significant drops when the source set was also considered. The interesting observation is that the Heterogeneous Domain (HD) such as CIFAR-100 does not have a significant drop in accuracy, which makes sense, as CIFAR-100’s CR space does not overlap with ImageNet-1K’s CR space (see Figure 7).

Table 13 shows the performance of CR-Inception-V3 with one image from each class to ten images from each class. For this experiment, we considered only ($\mathbb{T} \Rightarrow \mathbb{T}$) setting. The interesting to see that the CRL model with just ten images from each class all

the dataset Top-1 accuracy reach more than 75% of accuracy achieved when all instances are used.

Table 13: CR-Inception-V3 Accuracy with Increasing Target Instances (#Ins.)

#Ins.	CalTech-101		CalTech-256		CIFAR-100		ImageNet-1K	
	T-1	T-5	T-1	T-5	T-1	T-5	T-1	T-5
1	70.2	83.4	41.4	54.8	20	38	32.5	50.1
2	79.2	92.3	51.7	66.7	25.5	52.2	44	64.9
3	85.3	95.2	56.5	72.1	30.3	56.9	50.9	72.3
4	85.6	96.7	61.5	77.1	36	62.9	54.9	76.2
5	86.8	97	63.6	78	38	66.7	58.1	79
6	87	97.6	65.1	79.4	40.9	69.5	60.3	81.2
7	89.4	97.7	67.2	82.2	42.4	70.8	61.3	82.2
8	90.7	98	68.2	82.4	44.6	73.6	63.4	83.6
9	90.8	97.9	69.8	83.8	44.8	74.1	64.6	84.6
10	91.2	98.2	70	84.4	45.8	75.4	65.4	85.2
all	93.9	99.1	77.8	92.4	57.9	90.5	73.9	93.3

2.5.9 Results for Class Representative Graphs

Figure 15 shows the CR Graphs that are composed of class representatives (CRs) as well as their relationships in a given domain. In these CR Graphs, the nodes represent CRs, and the edges (CR-to-CR) are the relationship between two CR. The CR graph is undirected, as there is a symmetric relation between the source and target CRs that the similarity from the source CR to the target CR is equal to one from the target CR to the source CR. The attributes of the class are represented in the following manner; the node size indicates the number of images, and the node color indicates the CR accuracy. The

Table 14: Accuracy for CR-Inception-V3
Zero-Shot Learning Tasks

Dataset	Recognition Task Accuracy		
	Accuracy	$\mathbb{T} \Rightarrow \mathbb{T}$	$\mathbb{T} \Rightarrow \mathbb{S} \cup \mathbb{T}$
CalTech-101	Top 1	93.9%	87.4%
	Top 5	99.1%	97.5%
CalTech-256	Top 1	77.8%	40.9%
	Top 5	92.4%	54.9%
CIFAR-100	Top 1	57.9%	57.4%
	Top 5	90.5%	83.4%

color schema for nodes is relatively simple: red is bad, white is marginal, green is super high. The CR-to-CR similarity is represented as the edge color and Silhouette Width. Similar to the node color, the edge color represents the CR-to-CR similarity; brown is similar while white is not similar.

Through the CR Graph, we now explain the reason why the target domain, i.e., ImageNet-1K, does not perform well even if it is the same with the source domain ImageNet-1K Pre-trained Inception-V3, compared to the state-of-the-art accuracy (as shown in Table 12). As mentioned previously, the red nodes are the troublemaker, and the green nodes are the Superstar (see Section 2.4). The Superstars and Troublemakers show a visible pattern on their edge density. Troublemakers show very high edge density while Superstars show lower edge density in the CR Graph of ImageNet-1K.

Silhouette Width (SW) was used to validate the consistency within clusters of data. Specifically, the SW measurement was useful in the CR validation in terms of two aspects: (i) how well each CR is modeled within a domain (dataset) or (ii) how well each instance

is labeled within a CR (class). The SW value is a measure to check how similar an object is grouped to its own CR (cohesion) against other CRs (separation). The computation is based on Euclidean distance, and the range of these SW values is from -1 to $+1$. A high value indicates that the CR is well defined. In other words, the members are well-matched together while they are not matched to the ones in neighboring CRs.

Tables 15- 16 shows the statistics of the four CR Graphs showing in Figure 15. The Troublemakers' SW mean for all the datasets have a negative value. This indicates that there are significant similarities between the instances of the Troublemakers. Similarly, the Superstars show the positive SW value means, and this signifies how well the data fit the CR model.

Table 15: CR Graphs for ImageNet-1K, CalTech-101, CalTech-256, CIFAR-100

	ImageNet-1K	CalTech-101	CalTech-256	CIFAR-100
Total Class# (C)	1000	101	256	100
Total Relationship# ($C * (C - 1) / 2 + C$)	500500	5151	32896	5050
Mean Similarity (δ_s)	43%	41%	45%	74%
Mean Accuracy (δ_a)	75%	91%	77%	58%
Mean Neighbor Accuracy (δ_n)	73%	91%	76%	58%
Mean Image# (δ_c)	897	63	83	419
Mean SW (δ_w)	-0.0529	0.0265	-0.0763	-0.0197
Relationship(%)(Similarity $> \delta_s$)	52%	50%	50%	53%
Relationship(%)(Accuracy $< \delta_a$)	42%	37%	44%	52%
Troublemaker Candidate (%) (Accuracy $< \delta_a$ & Similarity $> \delta_s$)	26%	22%	26%	33%
Superstar Candidate (%) (Accuracy $> \delta_a$ & Similarity $< \delta_s$)	30%	37%	32%	27%
Troublemaker Classes (Bottom 10%)				
	ImageNet-1K	CalTech-101	CalTech-256	CIFAR-100
Troublemaker Class (%)	10%	10%	10%	10%
Troublemaker Relationship (%)	6%	5%	7%	27.6%
Troublemaker Mean Similarity	53%	63%	56%	84%
Troublemaker Mean Accuracy	48%	70%	47%	30%
Troublemaker Mean Neighborhood Accuracy	70%	82%	74%	53%
Troublemaker Mean Neighborhood#	303	28	92	36
Troublemaker Mean Image#	903	29	76	420
Troublemaker Mean SW	-0.2300	-0.0507	-0.2237	-0.0665
Superstar Classes (Top 10%)				
	ImageNet-1K	CalTech-101	CalTech-256	CIFAR-100
Superstar Class (%)	10%	10%	10%	10%
Superstar Relation (%)	6%	3%	3%	6%
Superstar Mean Similarity	35%	32%	37%	65%
Superstar Mean Accuracy	92%	100%	98%	80%
Superstar Mean Neighborhood Accuracy	75%	92%	78%	59%
Superstar Mean Neighborhood#	313.31	17.2	96	31.7
Superstar Mean Image#	895.44	46.4	146.3	419
Superstar Mean SW	0.0659	0.0728	0.1572	0.0459

Table 16: CR Graphs for ImageNet-1K, CalTech-101, CalTech-256, CIFAR-100

	ImageNet-1K	CalTech-101	CalTech-256	CIFAR-100
Total Class# (C)	1000	101	256	100
Total Relationship# ($C * (C - 1) / 2 + C$)	500500	5151	32896	5050
Mean Similarity (δ_s)	43%	41%	45%	74%
Mean Accuracy (δ_a)	75%	91%	77%	58%
Mean Neighbor Accuracy (δ_n)	73%	91%	76%	58%
Mean Image# (δ_c)	897	63	83	419
Mean SW (δ_w)	-0.0529	0.0265	-0.0763	-0.0197
Relationship* (%) (Similarity $> \delta_s$)	52%	50%	50%	53%
Relationship* (%) (Accuracy $< \delta_a$)	42%	37%	44%	52%
Troublemaker Candidate* (%) (Accuracy $< \delta_a$ & Similarity $> \delta_s$)	26%	22%	26%	33%
Superstar Candidate* (%) (Accuracy $> \delta_a$ & Similarity $< \delta_s$)	30%	37%	32%	27%

* marked rows represented percentage of classes in the dataset satisfying the condition mentioned.

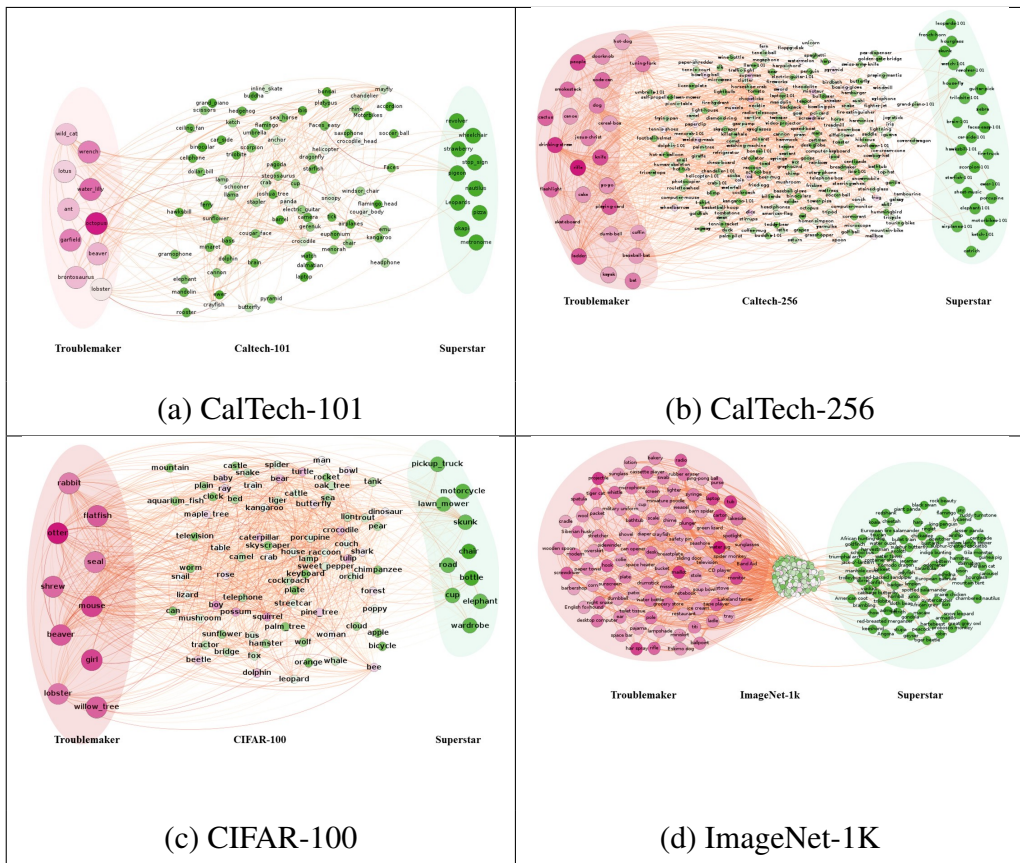


Figure 15: Class Representative Graphs

2.6 Discussion

The limitations of the CRL model are that a source environment (except the fully connect layers) is still required for generating a feature map for any input in the testing. The size of the source environment may be too big to fit in low-end devices like a mobile devices. In our research, we can provide a cloud service for the feature map generation as a basic interface, such as Application Programming Interfaces (APIs) for lightweight mobile applications.

The Class Representative (CR) generation was obtained by extracting the abstraction of the distribution of each feature in the Class Representative Feature Space (CRFS) of an input image. For the purpose, we used a simple average mean approach. Thus, a CR can be sensitive to outliers and sample size bias of the CRFS. The CRL model was extremely strong at the Top-5 inferencing compared to Top-1 inferencing (see Table 8). The CR computation might not be accurate due to bias or unexpected outliers. This indicates that the high similarity between some CRs can lead to misclassification. To overcome the limitation of the CR Generation, We will explore an advanced optical model such as Fisher Vector and Gaussian-Mixture-Model. We also can use unsupervised deep learning techniques such as the autoencoder in learning efficient data codings to reduce the CR's Feature Space to a more optimal representation. We can further extend it to determine the common and unique features of the CR vectors and find the weights that maximize the uniqueness between CRs.

The CRL inferences based on the simple similarity matching are subject to some limitations. We are currently using cosine similarity between CRs and an input image vector. It is possible that further fine tuning of the CRL inferencing for complex classifiers (e.g., very large CRs or multi-domain/modality CRs) could have improved predictive performance. The CR space could be simplified by applying reduction techniques (PCA or t-SNE). The improvements are to develop attention vectors or semantic CR model such as a hierarchy of CRs. The CR Graph possess an ability to help an inference process by using the CR Community.

The CRL model has a potential extension to have the open set recognition with

$\mathbb{T} \gg \mathbb{S}$, in this chapter, we show dataset with 0.278 openness factor [88]. The potential extension includes validating the CRL model using the metric for Generalised Zero-Shot Learning proposed in Xian et al. [89].

2.7 Conclusion

We presented the Class Representative Learning (CRL) model that is composed of class-level classifiers built by utilizing activation features of Convolutional Neural Network for classification problems. The CRL model has the ability to generate the CRs one by one through the aggregation of CNN activation features and inference by matching a new input to the CRs. Due to the independence of CRs, we could dynamically generate and utilize classifiers for image classification problems. We also represented the CR Graph with the CR accuracy and the relationships of CRs and interpreted a model performance by identifying good and bad performers correspond to classification performance as well as the similarity between CRs. We have conducted a comprehensive evaluation of the CRL model using the four benchmark datasets. We have shown the capacity of the proposed algorithm for the model generation by dynamically composing selected CRs. The accuracy of the classes in these models are significantly improved compared to ones in their original model. Excellent scalability of our dynamic models is shown for an increasing class# in a model. The CRL model outperformed state-of-the-art Zero-Shot Learning (ZSL) in terms of learning time and accuracy. It also showed improved performance compared to the existing MobileNet models for image classification. The work presented in this chapter was published as part of Chandrashekar et al.[90].

CHAPTER 3

CLASS REPRESENTATIVES FOR ZERO-SHOT LEARNING USING PURELY VISUAL DATA

3.1 Introduction

Deep learning technologies have received significant attention for large-scale image classification in the area of computer vision. The substantial requirements for such image classification tasks are the availability of a large amount of labeled data. Besides the limited amount of labeled data for supervised learning, we face severe challenges in applying image classification models to real-world problems, such as a lack of effectively transferring knowledge from one domain to another, or the effective adaption for newly generated data daily. There is a strong need for systematic image classification that supports active transfer learning and scalable solutions for real-world applications.

Recent efforts are focused on developing zero-shot learning (ZSL) or few-shot learning (FSL) that aims to handle the challenges of the real-world applications of image classification. ZSL aims to recognize instances from unseen or unknown (target) categories by using external linguistic or semantic information through intermediate-level semantic representations from seen or known (source) categories [61, 91, 62]. Rapid growth in recent years leads to the development of innovative methods in ZSL [4] and FSL [7, 8]. These studies focus on effective transfer learning by fully leveraging information from pre-trained models. The central idea behind these studies is to link known and

unknown classes through auxiliary information and visually distinguish them [89, 92]. It will allow them to learn from a few examples or even without seeing them. Alternatively, it is required to represent each of the new instances to match them on a semantic space, which minimizes training efforts and maximizes learning outcomes [9, 10, 11].

The ZSL works demonstrated their effectiveness in transferring from prior experiences to new classes. The semantic space model used in one of the most popular ZSL approaches is a joint embedding framework, called label-embedding space [4, 12], or based on attribute space [13]. Label-embedding space is based on a combination of visual embeddings and word embeddings, while attribute space is based on attribute annotations for the ZSL model. Such external or auxiliary information is used as a form of combining two or more sources of data, e.g., *image features + word embedding*, *image features + attribute information*, or *image features + ontologies*.

This chapter proposes a novel zero-shot learning model, called class representative learning (CRL), which builds class prototypes from image instances for each class and defines them as class representatives (CRs). The class prototypes for source and target domains will be generated using the features extracted using a projection function with typically a convolution neural network trained in the source (seen) domain. CRs are a universal representative of the classes in the source and target domains that aims for the effective transfer learning from the source domain to the target domain. This study demonstrates the usefulness of the CRL novel approach based on the universal representatives for transfer learning in image classification, with significantly improved performance for ZSL and G-ZSL problems.

The class representative learning (CRL) model can be categorized as a Class-Inductive Instance-Inductive projection method (as defined in [4]). In the training phase for the seen classes, the feature learning model is built from the training instances. During the testing phase for the unseen classes, both unseen and seen prototypes are projected into the same space, based on learned models. This study adopts the evaluation methods defined in Xian et al. [92, 89], but extends it to the seen classes built from the feature extraction learning from ImageNet dataset [67]. The performance of the generalized zero-shot learning (G-ZSL) algorithm was validated with the harmonic mean of seen and unseen classification performance [89, 92].

CRL is similar to the generative model in building a prototype class by class, using a class representative criterion [93, 48]. However, they are different since the CR generation is not based on probabilistic, but based on the generalized mean of aggregated features. For the classification step, a standard projection approach is used to compare the prototypes with each other. Compared to existing ZSL algorithms, CRL is required solely visual data, rather than both visual data and auxiliary information. Still, higher accuracy can be achieved compared to the state-of-the-art research in ZSL and G-ZSL.

The contributions of this chapter can be summarized as follows:

- Proposing the CRL model as an efficient way of building class-level classifiers by fully utilizing the features from a pre-trained Convolutional Neural Network (CNN) with purely visual data;
- Designing a universal representation, called class representative feature space (*CRFS*), for source and target classes that can be applied to multiple cross domains;

- Applying the CRL model to ZSL and G-ZSL problems;
- Designing the parallel ZSL and G-ZSL algorithms based on CRL;
- Through extensive evaluations, the proposed CRL model shows significant performance gain compared to the state-of-the-art research in ZSL and G-ZSL.

3.2 Related Work

3.2.1 Universal Representation

Ubernet [38] is a *universal* CNN that allows solving multiple tasks efficiently in a unified architecture. It provides a simple end-to-end network architecture for diverse datasets, and scalable and efficient low memory processing. Also, universal representations [39, 40] have been shown to work well in a uniform manner for visual domains. They have proven to be efficient for multiple domain learning in relatively small neural networks. Rebuff et al. [41] presented that universal parametric families of networks could share parameters among multiple domains using parallel residual adapter modules. Similar to our work, all these works presented universal representations for various domains or various tasks. However, unlike CRL, their model cannot adequately support effective transfer learning in multiple domains.

Feature selection is a crucial step in machine learning since it directly influences the performance of machine learning. The right choice of features drives the classifier to perform well. However, Kapoor et al. [42] observed that finding useful features for multi-class classification is not trivial. It is because of the volume in the high-dimensional

feature space and the sparseness over the search space. Dictionary learning [43] was presented to determine the subspaces and build dictionaries by efficiently reducing dimensionality for efficient representations of classes in the domain. They overcame the sparsity constraints and improved the accuracy by identifying essential components of the observed data. In CRL, the class representative feature space $CRFS$ provides a basis for building the prototypes, i.e., CRs, using the features from the CNN network that are a uniform representation of the images.

In the context of zero-shot learning and few-shot learning, the representatives are known as a class prototype, which is defined as vector representation in semantic space corresponding to each class [4]. In EXEM, a similar concept, class exemplar, was introduced as the center of visual feature vectors that are used in prediction and label embedding [94]. A hierarchical super prototype was proposed based on a combination of semantic prototype and visual data for seen and unseen classes [95]. Fu et al. proposed a prototype graph where each prototype is a node in the graph, and the semantic relationship between classes is an edge for their connectivity [96]. The prototype is also used in FSL, as demonstrated by Snell et al. [47].

In the class representative learning (CRL) model, the class prototype is part of the predictive step, unlike the previous work defined as an intermediate step towards a learning model. The class prototype (i.e., class representatives) can be built independently class by class to be self-contained and not dependent on other classes. For example, a class representative (CR) for *dog* class is built using the data only from one class (i.e., *dog*) without considering other classes, such as *horse* or *cat*.

3.2.2 Zero-Shot Learning

Zero-shot learning (ZSL) uses a semantic encoding in predicting new classes that were derived from a semantic knowledge base [24]. Besides the knowledge bases, explicit and external attributes are considered for visual learning [24]. In addition to standard image feature extraction techniques, other feature learning techniques such as boosting techniques [49], object detection [50], chopping algorithm [51], feature adaption [52], and linear classifiers [53] are used to enhance the accuracy of unseen classes.

Wang et al. [4] categorized zero-shot learning based on feature requirement into *engineered semantic space* and *learned semantic space*, as shown in Figure 16. *Engineered semantic space* is further sub-categorized into *attribute space*, *lexical space*, and *text-keyword space*. *Learned semantic space* is categorized into *label-embedding space*, *text-embedding space*, and *image-representation space*.

Table 17 shows the proposed method and the existing ZSL models compared in the evaluation section. Recent ZSL works mostly include two kinds of semantic spaces, namely *label-embedding spaces* [4, 12] and *attribute spaces* [4] (also known as probability prediction strategy [12]). The image representative space is present in all the zero-shot learning works but typically coupled with additional semantic space. The ZSL approaches based on the label-embedding space focus on learning a projection strategy. This strategy maps semantic features extracted from the image representative space to the labels that are represented in a high dimensional embedding such as Word2Vec [54] or Glove [55]. Image representative space are typically learned from convolutional neural networks [13, 56, 57, 58, 59, 60]. Attribute space or Probability prediction strategy pre-trains attribute

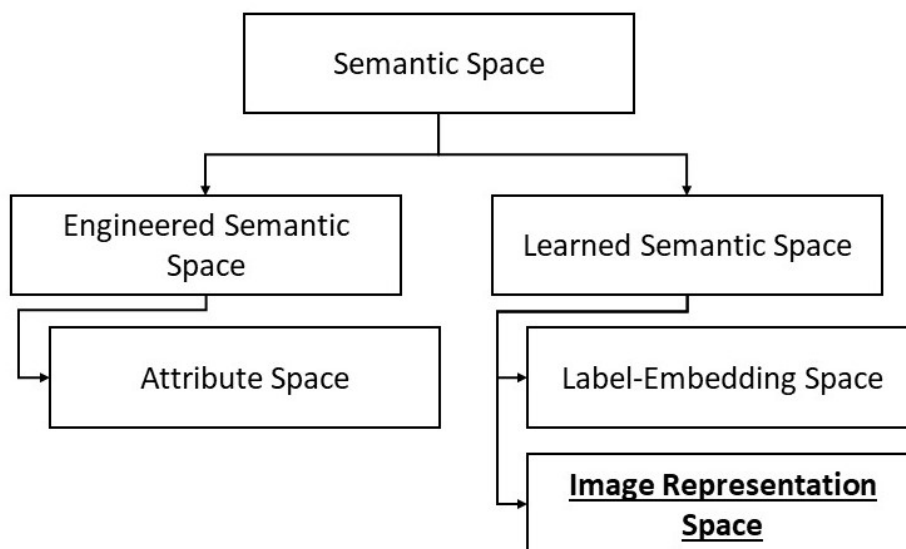


Figure 16: Zero-shot Learning and Semantic Spaces [4]

classifiers based on the source data [12], where an attribute is defined as a set of terms having the properties for a given class [4]. A class has distinguishing attributes, the embedding of visual features for the defining characteristics of the class, and assignment of the label to a class based on these features and attributes [61, 62, 17]. Unlike these works, CRL focuses on uses a mono-modal, specifically just the image representative semantic space. Using only the image representative gives CRL the advantage in terms of classification performance comparing to using a multi-modal semantic space approach.

Pure image representation space-based ZSL approaches are rarely observed. One of the few works was to use image deep representation (i.e., neural network-based) and fisher vector for the inference [63], and an extension of this approach was used to create an unsupervised domain adaptation [64]. Zhu et al. use a partial image representation method to achieve a universal representation for action recognition [65]. Like the above

approaches, CRL uses only image representative semantic space but with a unique method of creating class prototypes and perform inference all within the space.

3.2.3 ZSL Projection Methods

The ZSL approaches can be categorized into classifier-based and instance-based methods. Based on the categorization, the CRL model is defined as a zero-shot learning model that uses image representation semantic space and employs an instance-based projection method for model building and inference. The projection method provides insights on the labeled instances of an unseen class by projecting them onto the common feature space or the semantic space where instances and prototypes are compared [4]. We consider four different zero-shot learning inference methods, namely *classifier-based correspondence method*, *classifier-based relationship method*, *instance-based projection method*, and *instance-based synthesizing method*.

Table 17: Related Work: Zero-Shot Learning Methods

Zero-Shot Learning Method			
Instance-based		Classifier-based	
Projection Method	Synthesizing Method	Relationship Method	Correspondence Method
CRL Model(ours)			SJE [56]
DWV[69]	GAN+ALE[101]		LATEM [60]
Deep-SVR[61, 97]	GAN+Softmax[101]	SSE [105]	ALE [91]
ConSE[62]	CADA-VAE[102]	AMP [58]	DeViSE[57]
CMT[98]	cycle-GAN[103]	SynC [106]	ESZSL[59]
SAE[99]	BPL+LR[104]		SP-AEN [107]
Embed[100]			

Classifier-Based Correspondence Method constructs a correspondence between binary

one-vs-rest classifier and unseen class prototypes to classify unseen classes. The compatibility function is a key part of the correspondence method. It takes instances and prototypes as an input to compute a compatibility score denoting the probability of instances to classes [108, 89, 4]. A bilinear function is a widely used compatibility function including DeVISE [57], ALE[108], SJE [56], and ESZSL [59]. The other widely used method is the projection functions such as linear projection [109, 110]. Even though the correspondence method uses classifier and prototype method, no explicit relationships between classes are modeled due to one vs. all strategy. As the class relationship plays a vital role in understanding ZSL performance, the CRL model uses a pure-prototype approach that helps to discover the class relationship to increase the model’s interpretability.

Classifier-Based Relationship Method constructs classifiers for the unseen classes based on the relationships among classes [4]. SSE [105] uses binary one-vs-rest classifiers for seen classes, and unseen class prototypes establish their relationship with seen classes. AMP [58] uses a directed k -nearest neighbor graph where each edge establishes the relationship between the classes. The relationship method focuses on creating a classifier by creating a class-to-class relationship; this inter-class dependency creates scalability issues. To handle scalability issues, CRL focuses on inter-class independence during model building, and the relationship between class is determined using their prototypes.

Instance-Based Projection Method’s insight to classify is by obtaining labeled instances for the unseen classes by projecting both the feature space instances and the semantic space prototypes into a shared space [4]. The projection space is defined as a space where the classification is performed. The approaches using projection methods

are further categorized according to the projection space: *Semantic Space as Projection Space*, *Visual Space as Projection Space*, and *Transductive Projection Strategy*.

Semantic Space as Projection Space: Projection function is learned to project the visual feature space to semantic space with a linear model or non-linear model. Cross-modal transfer (CMT) was proposed based on semantic word vector representations and Bayesian framework to differentiate the semantic manifold of seen classes for transfer learning of unseen classes [98]. CMT and few other works use regression function as the projection method and softmax classifier on the semantic space [98, 24, 111]. ConSE uses a projection method achieved from a convex combination of seen class prototypes (ConSE) [62]. ConSE has an n-way classifier built on seen data and is used to predict probabilities on the unseen instances. The projection function of Deep-SVR consists of classifiers for attributes. The probabilities from attribute classifiers are coordinates with projected instances in semantic space using 1-NN classification [61, 97].

Visual Space as Projection Space: Projection Function is learned to project the semantic space to visual feature space. Linear regression projection approach was used to mapping from the target space to the source space [112]. However, this approach is based on a strong assumption like a multivariate normal distribution of data and also lacks advanced similarity measures like cosine similarity or multi-modal data distributions. Non-linear regression (DEM) uses the visual space as the embedding space, resulting in fewer hubness problems than other ZSL approaches [100]. Unseen visual data synthesis (UVDS) introduced a latent embedding space that takes into account semantic space and visual feature space [113]. Jiang et al. Multiple projection spaces, such

as semantic auto-encoder (SAE), were designed to learn a more generalized projection function [99].

Transductive Projection Strategy Manifold regularization was used together with data augmentation strategies to enhance the semantic space, resulting in easy access to testing data in the training phase [114]. Matrix factorization with testing instances and unseen class prototypes was designed for unsupervised domain adaptation to overcome the projection domain shift problem [115]. The self-training strategy aims to adjust the prototypes of unseen classes with the testing instances when performing 1-NN classification. For an unseen class, the prototype is adjusted as the mean value of the k nearest testing instances [116, 70, 117, 118, 119]. This unseen prototype creation is also used in a few-shot learning settings [47]. Markov chain process-based projection method was proposed to compute semantic manifold distance in embedding space as a seamless fusion of the semantic relatedness and embedding based methods for ZSL [58]. Fu et al. presented a unified framework based on vocabulary-informed learning. It incorporates distance constraints from vocabulary atoms for projecting closer to their correct prototypes in semantic manifold-based recognition [69].

As mentioned earlier, the CRL model can be categorized as a projection method. Unlike most of the existing work, CRL uses the visual feature space created using projection function to inference without depending on classifiers. Class representative generation is similar to prototype creation by self-training strategy. The zero-shot learning approaches using self-training uses the visual feature and some additional semantic space for creating the prototype. The few-shot learning approach uses a massive network classifier

model post prototype generation. The unique feature of class representative generation is the sole usage of visual feature space.

Instance-Based Synthesizing Method based zero-shot learning has been prevalent last year with the use of a generative neural network to create additional data points, especially for unseen classes, to handle the problem pertinent to data imbalance issues. GAN+ALE [101], GAN+Softmax [101], and cycle-GAN [103] uses various types of (GAN). The GAN-based model typically samples a random vector and combines that with the unseen class prototype to form the input to the generator, whereas CADA-VAE [102] uses variational auto-encoder for data synthesizing. BPL[104] uses semantic feature synthesis by perturbation approach, which incorporates by directly perturbing the seen class samples to unseen class prototypes. In the context of the projection method, BPL [104] uses bidirectional projection learning as part of a competitive learning strategy between seen samples and unseen prototypes. Even though synthesizing methods differ in architectural sense from CRL Model, we consider them as some of the top-performing models in Zero-Shot Learning.

3.3 Class Representative Learning Model

In this chapter, we present the class representative learning (CRL) model designed to project the input data onto a global space and generate a universal representation for domains and use it for inference in zero-shot learning. The space of the CRL model is similar to the universal representation proposed by Tamaazousti et al. [66], where visual elements in the configuration (e.g., scale, context) can be encoded universally for

the transfer learning. Unlike their work, our CRL representation is defined based on the aggregation patterns from the activation of neurons of the projection function (typically a pre-trained model, such as CNN). The CRL model’s fundamental concept is its ability to create the representatives of class independently from other classes for a given domain.

Figure 17 shows the conventional label-embedding based zero-shot learning model, and Figure 18 shows the class-representative model in a similar setting. As shown in Figure 17, Nourouzi et al. introduced zero-shot learning two-step mapping function [62], where the first step is the projection function, and the second step is inference function in the label-space. As shown in Figure 18, the two-step mapping stays in place. However, the second mapping becomes non-essential because the class representatives (image feature prototypes) have the label as tagged information. In the label-based embedding model (Figure 17), the second mapping plays an essential role where the image feature space maps into the label space. In most of the existing works, the classification happens in the label space, wherein CRL’s classification happens in image feature space, also known as class representative feature space (CRFS) [90].

3.3.1 Problem Setup

Assume the given source data $D_s = \{x_i, y_i\}_{i=1}^{m_s}$ of m_s labeled points with a label from the source class \mathbb{S} , where $x_i \in \mathbb{R}$ is the feature of the i^{th} image in the source data, and $y_i \in \mathbb{S}$, S is the set of source classes. The target data is represented as $D_t = \{x_i, y_i\}_{i=1}^{m_t}$ classes where $y_i \in \mathbb{T}$. For each class $c \in \mathbb{S} \cup \mathbb{T}$, has a class representative $CR(c)$ which is the semantic representative of class c . Furthermore, the source label \mathbb{S} and the target

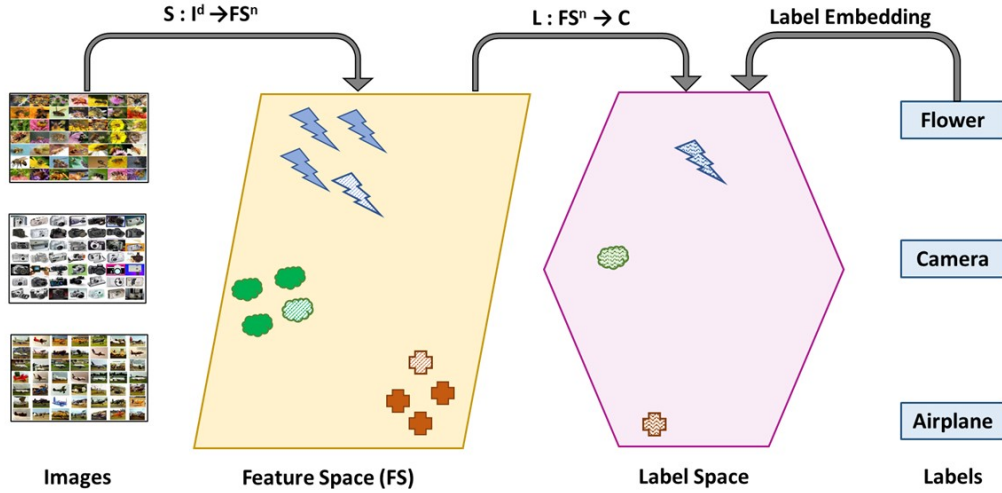


Figure 17: Label Embedding Based Zero-Shot Learning

label \mathbb{T} are considered such that $\mathbb{S} \cap \mathbb{T} = \emptyset$. For simplicity, the source and target datasets have overlapped labels, but these overlapped classes are considered distinct. In the CRL model, the source data are considered as *seen*, and the target data are *unseen*. In other words, the target data are not used in the learning process. Table 18 summarizes all the symbols and notations used in the CRL model.

The goal of the CRL model is that given a new test data x^* , the model will classify it into one of the classes y^* , where $y^* \in C$. The CRL model defines a universal problem for a ZSL approach as well as G-ZSL as follows:

- Zero-Shot Learning (ZSL): $y^* \in \mathbb{T}$
- Generalized Zero-shot Learning (G-ZSL): $y^* \in \{\mathbb{S} \cup \mathbb{T}\}$

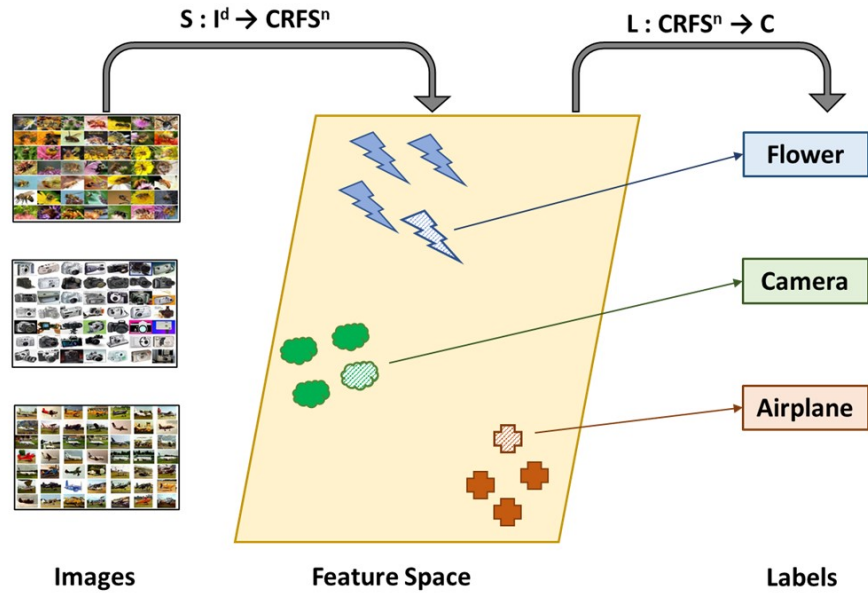


Figure 18: Class Representative Based Zero-Shot Learning

There are no dependencies among CRs in any of the models. The difference between these two models is in the properties of the inference. If the CR of the target set was introduced, then it would be *ZSL*, and if both CRs of the source set and the target set are introduced, then it would be *G-ZSL*.

Definition: Class representative (CR) is a representative of K instances in a single class. The activation feature map of the CR is a unique characteristic pattern of visual expression using the projection function; for example, the feature map generated through the deep learning process using convolutional neural networks as the projection function (CNN). Activation feature map (AFM) is a vector of features extracted from a base model, which is learned from the source dataset. Thus, CR is an abstraction of instances of a class by

computing an aggregation of the average mean vectors of the AFM for the K instances. The CR characterizes a class and differentiates one class against another. The class representatives $CR(c)$ for Class c is represented as $\{CR_c^1, CR_c^2, \dots, CR_c^n\}$ with n dimensions. Each dimension corresponds to a separate feature. If a feature occurs in CR, its value in the vector is non-zero.

Table 18: Formal Symbol and Notations in the CRL model

Notation	Description
$D_s \& D_t$	Source and Target Domain
$m_s \& m_t$	#Data Points from Source and Target, respectively
$\mathbb{S} \& \mathbb{T}$	Source and Target Label Set
C	#Classes
x	Feature Vector of Labeled Data Point
y	Label of Data Point
j	#Neurons in a Given Activation Layer
b	Base vector learned in $P(\cdot)$ $\{b_1, b_2, \dots, b_j\}$
$CR(c)$	Class Representative of Class c where $c \in C$
x^*	Unlabeled Data Point
y^*	Predicated Label for x^*
$CRC(\cdot)$	Class Representative Classifier
$P(\cdot)$	Projection Function
$L(\cdot)$	Inference Function
$CRFS^n$	Class Representative Feature Space
n	Dimensions of the $CRFS^n$

3.3.2 Class Representative Classifier

We introduce a class representative classifier $CRC : I^d \rightarrow c$ that maps an input image space I^d of the dimension d and $c \in \mathbb{S} \cup \mathbb{T}$. Classifier has two essential functions, namely the projection function $P(\cdot)$ and inference function $L(\cdot)$. The projection function

takes the input images and learns feature space for class representatives through source data D_s . The inference function uses the class representative feature space $CRFS^n$ of the dimension n to classify it to class c . CRC is defined as a composition of two functions, as shown in Equation 3.1.

$$\begin{aligned}
 CRC &= L(P(\cdot)) \\
 P : I^d &\rightarrow CRFS^n \\
 L : CRFS^n &\rightarrow c
 \end{aligned} \tag{3.1}$$

The CRC model first learns a projection function S using the seen data, which aids the mapping of the input images I^d into class representative feature space $CRFS^n$ with n dimensions. \mathbb{T} the inference function L creates class representatives CR in the $CRFS^n$ of either seen data or unseen data depending on the setting. The L maps any new images from $CRFS^n$ to Label c where $c \in \mathbb{T}$ or $c \in \mathbb{S} \cup \mathbb{T}$.

3.3.3 Projection Function

In the class representative learning method, we use a projection mechanism that primarily uses only visual feature space. The visual feature space in the CRL method is known as the class representative feature space. The projection function is prevalent as part of the class representative classifier in two ways. First, learning the projection method using seen data and building the class representative feature space $CRFS^n$. Second, applying the pre-learned projection mapping mechanism to seen data or unseen data

depending on the setting for inference.

Class representative feature space (CRFS) is defined as a n dimensional semantic feature map in which each of the n dimensions represents the value of a semantic property. These properties may be categorical and contain real-valued data or models from deep learning methods [24]. Class representative feature space is the projection space. The class representative feature space represents n dimensional representative features is a form of the activation feature map (AFM). The design of the CRFS is based on the equations defines in [71]. The data points from D_t $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m_t)}, y^{(m_t)})\}$ with each $x^{(i)} \in \mathbb{R}$ and $y^{(i)} \in \mathbb{T}$ (as shown in Equation 3.3). A set of the mean of the base features is defined as $\hat{x}^{(i)}$ Note that the data points can also be defined in D_s that will be used in CRFS to understand both source and target domains (refer to Section 3.4.4).

$$\begin{aligned} \hat{a}(x_i) &= P(x_i) \\ \hat{a}(x_i) &= \underset{a^{(i)}}{\operatorname{argmin}} \|x^i - \sum_j a_j^{(i)} b_j\|_2^2 + \beta \|a^{(i)}\|_1 \end{aligned} \quad (3.2)$$

where

$$\begin{aligned} D_s &= \{x_i, y_i\}_{i=1}^{m_s} \\ D_s &\xrightarrow{P_b} \hat{D}_s \\ \hat{D}_s &= \{(\hat{a}(x_i), y_i)\}_{i=1}^{m_s} \end{aligned} \quad (3.3)$$

Class representative feature space is created based on the base vector b which has j dimensions with each $b_j \in \mathbb{R}_n$. The base vector b is generated with the projection function using D_s using optimization function such as stochastic gradient descent. Equation 3.2

points out the self-training based on unsupervised data. The activation $\hat{a}(x_i)$ consists of $\{\hat{a}(x_i^1), \dots, \hat{a}(x_i^n)\}$ with each $\hat{a}(x_i) \in \mathbb{R}$ forms the semantic property of CRFS. Each dimension of an activation vector $\hat{a}(x_i)$ is the transformation of input x_i using the base b_j , note that j is independent of size of input x_i . The base b is learnt through smooth approximation with L_1 sparsity penalty $a(i)$. Even though approximation does not lead to sparse features, Raina et al. reports using re-calibrated β value before computing labeled data representation improve the classification accuracy [71].

Projection function $P(\cdot)$ can potentially be a convolution network layer or a residual network layer. The advantage of having a projection function purely for mapping; any pre-trained models can be used in its place. The pre-trained model's availability to classify the source classes does not influence projection function mapping, instead of changes on how the base vector is optimized. By having projection function, being independently learned from the source classes gives the ability to use any advanced pre-trained model available as part of the class representative learning model.

3.3.4 Class Representative Generation

Class representatives (CR) are generated using the nearest prototype strategy by aggregating feature vectors. As the name specifies, class representatives create representatives from the instances projected in class representative feature space $CRFS^n$ using Projection Function $P(\cdot)$. Class representatives (CR) and the instances share the same feature space $CRFS^n$. The nearest mean feature vector with instances of the given class, i.e., class representatives, is computed for every class. Correctly, the average feature mean

operation was used to summarize the instances of classes. The CR is an aggregated vector of the mean features for all the elements in the feature maps.

For the class representative generation, we considered the transformed source dataset \hat{D}_s as the input (as shown in Equation 3.2). As we emphasize the parallelism and independence, we considered the individual activation vector $\hat{a}(x_i)$ such that $y_i = c$ where $c \in \mathbb{S}$, that will be used in formulating the CR as shown in Equation 3.4.

$$\begin{aligned}
 \hat{D}_s &= \{(\hat{a}(x_i), y_i)\}_{i=1}^{m_s} \\
 CR(c) &= \begin{cases} CR^1 = \frac{1}{m_s} \sum_{i=1}^{m_s} \hat{a}(x_i^1), & \text{if } y_i = c \\ CR^2 = \frac{1}{m_s} \sum_{i=1}^{m_s} \hat{a}(x_i^2), & \text{if } y_i = c \\ \vdots \\ CR^n = \frac{1}{m_s} \sum_{i=1}^{m_s} \hat{a}(x_i^n), & \text{if } y_i = c \end{cases} \quad (3.4)
 \end{aligned}$$

$$CR(c) = \{CR^1, CR^2, \dots, CR^n\}, \forall c \in \mathbb{S}$$

Equation 3.4 shows the feature-wise class representative generation in the class representative feature space. For the CR generation, the projected source data \hat{D}_s is considered. For each class $c \in \mathbb{S}$, the projected source data $\{(\hat{a}(x_i), y_i)\}$ is considered where $y_i = c$. Each feature is considered based on the $CRFS^n$ dimension ranged 1 to n , the average is considered to represent the corresponding dimension for $CR(c)$.

3.4 Class Representative Inference

3.4.1 Zero-Shot Learning Setting

Algorithm 1: CRC-Inference: Zero-Shot Learning Setting

Input: $D_t = \{x_i, y_i\}_{i=1}^{m_t}; x^*$
Output: y^*
ZSL Setting: $y^* \in \mathbb{T}$

1 Projection Function $P_b(x)$:
 | /* Base Vector b was learnt from D_s , Based on Equation 3.2 */
 2 | $\hat{a}(x) = P(x)$
 3 | return $\hat{a}(x)$

4 Inference Function $L(CR(c) \forall c \in \mathbb{T}, \hat{a}(x^*))$:
 5 |
 |
$$y^* = \operatorname{argmax}_{c \in \mathbb{T}} \{\cos(CR(c), \hat{a}(x^*))\} \quad (3.5)$$
 | where
 |
$$\cos(CR(c), \hat{a}(x^*)) = \frac{CR(c) \cdot \hat{a}(x^*)}{\|CR(c)\| \|\hat{a}(x^*)\|} \quad (3.6)$$
 |
$$= \frac{\sum_{j=1}^n \{CR^j(c) * \hat{a}(x^{j*})\}}{\sqrt{\sum_{j=1}^n (CR^j(c))^2} \sqrt{\sum_{j=1}^n \hat{a}(x^{j*})^2}}$$
 | return y^*

6 CR-Classifer Function $CR(C, x^*)$:
 | /* Projection of Unseen Data D_t into $CRFS^n$ Similar to
 | Equation 3.3 */
 7 | $D_t \xrightarrow{P_b} \hat{D}_t$ **for** $i \in 1 \rightarrow m_t$ **do**
 8 | | $\hat{a}(x_i) = P_b(x_i)$
 9 | **end**
 10 | $\hat{D}_t = \{(\hat{a}(x_i), y_i)\}_{i=1}^{m_t}$
 | /* Projection of New Data Point x^* into $CRFS^n$ */
 11 | $\hat{a}(x^*) = P_b(x^*)$ /* CR Generation: Based on Equation 3.4 */
 12 | **for** $c \in \mathbb{T}$ **do**
 | | 13 $CR(c)^j = \frac{1}{m_t} \sum_{i=1}^{m_t} \hat{a}(x_i^j), \text{ if } y_i = c \quad \& \quad \forall j \in (1, n)$
 14 | **end**
 | /* Classifier Function */
 15 | $y^* = L(CR(c) \forall c \in \mathbb{T}, \hat{a}(x^*))$ 96

The zero-shot learning setting for the class representative classifier (*CRC*), would involve using the pre-learned projection function P using source or seen dataset D_s on the unseen or target dataset D_t . The CR-based classifier involves three steps, namely *projection function* (as describes in Section 3.3.3), *class representative generation* (as described in Section 3.3.4) and finally the *inference function* L . Algorithm 1 presents the projection function P_b pre-trained on the source dataset D_s is used to map the target D_t instances and the test data point x^* to class representative feature space $CRFS^n$. Note that as recorded in all existing projection methods, each unseen class needs at least one labeled instance to create a prototype, i.e., class representative [4]. The class representative creation is independent of the number of instances per class, with the only requirement of having at least one labeled instance per class.

The aggregation of projected unseen dataset \hat{D}_t creates the class representative for all classes $c \in \mathbb{T}$. As shown in Equation 3.4, the dimensions of the projected are maintained for the class representative as well. The class representatives $CR(c) \forall c \in \mathbb{T}$ and the new data point $\hat{a}(x^*)$ reside the same feature space i.e., the class representative feature space $CRFS^n$.

For the inference, the cosine similarity between each class representative in T and projected new data point $\hat{a}(x^*)$ is calculated. The label information is retained and coupled with each class representative. For the final inference step, the label of the class representative with the highest cosine similarity is returned as the predicted class y^* for the new data point x^* .

The CRC classifier uses an instance-based projection method and inference method

with no learning for unseen data or any new data as an ideal zero-shot learning setting.

3.4.2 Generalized Zero-Shot Learning

Algorithm 2: CR-Inference: Generalized ZSL Setting

Input: $D_s = \{x_i, y_i\}_{i=1}^{m_s}$; $D_t = \{x_i, y_i\}_{i=1}^{m_t}$; x^*

Output: y^*

G-ZSL Setting: $y^* \in \{\mathbb{S} \cup \mathbb{T}\}$

1 CR-Classifier Function $CRC(D_s, D_t, x^*)$:

```

/* Projection of Unseen Data  $D_t$  into  $CRFS^n$  Similar to
   Equation 3.3 */
2  $D_t \xrightarrow{P_b} \hat{D}_t$ 
/* Projection of Seen Data  $D_s$  into  $CRFS^n$  Similar to
   Equation 3.3 */
3  $D_s \xrightarrow{P_b} \hat{D}_s$ 
/* Projection of New Data Point  $x^*$  into  $CRFS^n$  */
4  $\hat{a}(x^*) = P_b(x^*)$ 
// CR Generation: Based on Equation 3.4
5 for  $c \in \{\mathbb{S} \cup \mathbb{T}\}$  do
6      $CR(c)^j = \frac{1}{m_t} \sum_{i=1}^{m_t} \hat{a}(x_i^j),$ 
7      $\text{if } y_i = c \quad \& \quad \forall j \in (1, n)$ 
8 end
// Classifier Function
9  $y^* = L(CR(c) \forall c \in \{\mathbb{S} \cup \mathbb{T}\}, \hat{a}(x^*))$ 

```

Algorithm 2 shows the variation of Algorithm 1, incorporating generalized zero-shot learning(G-ZSL) setting. In the G-ZSL setting, the source dataset, target dataset, and new data point are projected into the feature space during the projection function. The projection method used here is based on pre-trained base vectors b learned from the source or seen dataset. Having projected both seen and unseen dataset onto the class

representative feature space $CRFS^n$, we can generate class representatives for all class $c \in \mathbb{S} \cup \mathbb{T}$ as shown in Equation 3.4. The class representative $CR(c) \forall c \in \mathbb{S} \cup \mathbb{T}$ and the projected data point $\hat{a}(x^*)$ reside in the same feature space $CRFS^n$ for the inference step. The inference step involves getting c , where the class representative $CR(c)$ has the highest cosine similarity with the data point, leading to the conclusion that data point is predicted with the label c .

3.4.3 Model Parallelism

Algorithm 3: CRC-Inference: ZSL Setting [Parallel Mode]

Input: $D_t = \{x_i, y_i\}_{i=1}^{m_t}; x^*$
Output: y^*
ZSL Setting: $y^* \in \mathbb{T}$

1 **Inference Function** $L(CR(c) \forall c \in C_p, \hat{a}(x^*))$:
2 |
3 |
$$y_p = \operatorname{argmax}_{c \in C_p} \{\cos(CR(c), \hat{a}(x^*))\} \quad (3.7)$$

4 | /* $score_p$ is the cosine similarity between $CR(y_p)$ and x^* */
5 | return $(y_p, score_p)$
6 **CR-Classifer Function** $CRC(D_t, x^*)$:
7 | **Broadcast:**
8 | Pre-trained base vectors b and new data point x^*
9 | /* C_p represents subset of classes split based on distribute
10 | function. D_p is data point for C_p with size of m_p */
11 | $D_p, C_p = \text{distribute}(D_t, \mathbb{T})$
12 | $Y, SCORE = \{\text{empty}\}$
13 | **PARALLEL MODE**
14 | **in parallel do**
15 | /* Projection of Distributed Data D_c s into $CRFS^n$ Similar to
16 | Equation 3.3 */
17 | $D_p \xrightarrow{P_b} \hat{D}_p$
18 | /* Projection of New Data Point x^* into $CRFS^n$ */
19 | $\hat{a}(x^*) = P_b(x^*)$
20 | /* CR Generation: Based on Equation 3.4 */
21 | **for** $c \in C_p$ **do**
22 | | $CR(c)^j = \frac{1}{m_p} \sum_{i=1}^{m_p} \hat{a}(x_i^j),$
23 | | if $y_i = c \quad \& \quad \forall j \in (1, n)$
24 | **end**
25 | /* Classifier Function */
26 | $(y_p, score_p) = L(CR(c) \forall c \in C_p, \hat{a}(x^*))$
27 | $Y, SCORE \leftarrow y_p, score_p$
28 | **end**
29 | **REDUCTION STEP**
30 | /* Reduction Step combines results generated from each C_p */
31 |
$$y^* = \operatorname{argmax}_Y \{SCORE\} \quad (3.8)$$

Algorithm 3 showcases the parallelized zero-shot learning inference for the class representative classifier. As there is no learning for target classes, *CRC* can support parallel processing for even larger datasets. The algorithm is designed with the CRCW (concurrent read concurrent write) model, which allows parallel computing, including I/O, with the shared memory and processors. The parallelized class representative classifier function *CRC* has two major steps, first involves in the declaration of the global variables, broadcasting certain inputs, and distribution of other inputs. Global variables are the variables that can be accessed and updated across the parallel process. Algorithm 3 declares predicted label set Y and corresponding cosine similarity score set $SCORE$ as global variables. The broadcast variables are the ones that represent the new projected data point $\hat{a}(x^*)$ and pre-trained base vectors b for the projection function $P(\cdot)$. Next, the target data points D_t and target domain \mathbb{T} gets distributed into smaller sets of classes using *distribute(.)* function. The *distribute* function splits the target classes \mathbb{T} and the corresponding data points D_t into smaller sets of classes C_p and their corresponding data points D_p . The distribution can be aligned according to the parallelization capacity of our method. The class set C_p can be as small as a single class due to the complete independence for every class in target classes $c \in \mathbb{T}$. Each parallel processor works with each set data points D_p and classes C_p to create class representatives $CR(c)$ for all classes in C_p and compare with the new data point $\hat{a}(x^*)$. At the end of each parallel execution, a predicted label y_p and the cosine similarity $score_p$ between $CR(y_p)$ and $\hat{a}(x^*)$ is returned. The predicted label and score are accumulated onto the global variables, Y , and $SCORE$. The final resultant label y is obtained using a reduction step where the label

with the highest cosine similarity in Y , $SCORE$, is returned (refer to Equation 21).

3.4.4 Validation using Domain Adaptation

We validate domain compatibility by checking the compatibility between source (seen) domain and target (unseen) domain inspired from existing domain adaptation techniques. The domain adaptation problem arises when the source domain data distribution is different from target domain data distribution. Domain adaptation aims to learn a predictor function in given a feature space using the source domain and apply it to the target domain. The hypothesis of domain adaptation is verified by measuring the distance between using the probability distribution obtained as a resultant of the predictor function between source and target [120, 121].

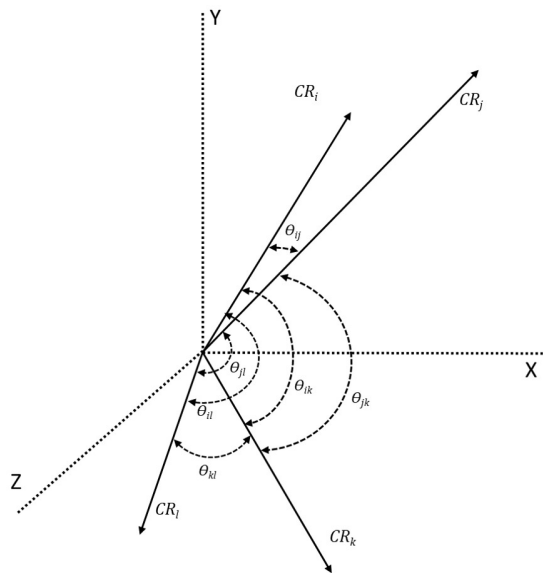


Figure 19: Source Domain - Cosine Similarity Distribution

We use domain adaptation to validate the class representative distribution of source

and target domains. The standard way of domain adaptation uses the predictor function and the instance distribution of source and target to measure the divergence. Unlike the conventional way, we formalize domain distribution through the cosine similarity between class representatives of all the sources classes and the cosine similarity between class representatives of the target classes. The domain distribution using cosine similarity between class representatives showcases the confusion between the classes, i.e., if the current feature space is favorable for the discrimination of classes within the source and target domain. The goal is to see if the target domain has the same CR-to-CR cosine similarity distribution as the source domain.

Figure 19 shows an abstract three dimensional (X,Y,Z) vector space model of CR cosine similarity distribution. Note that the dimensions of vector space model corresponds the n dimensions of $CRFS$. We consider three classes (i,j,k) from source domain and their corresponding class representatives (CR_i, CR_j, CR_k) . We also consider three classes (i',j',k') from target domain and their corresponding class representatives $(CR_{i'}, CR_{j'}, CR_{k'})$. The cosine similarity between each pair of class representative is calculated to each domain distribution in the class representative feature space (refer to Equation 3.9).

$$\begin{aligned}
f(\mathbb{S}) &= \{\theta_{i,j}, \theta_{i,k}, \theta_{k,j}\} \\
f(\mathbb{T}) &= \{\theta_{i',j'}, \theta_{i',k'}, \theta_{k',j'}\}
\end{aligned} \tag{3.9}$$

where

$$\theta_{i,j} = \cos(CR_i, CR_j)$$

The domain adaptation’s hypothesis typically uses the distribution over the instances in a given domain (i.e., either source or target)[120, 121]. Usually, this distribution is the probability distribution over the prediction function; instead, we use the distribution of cosine similarity. ESZSL was the first to introduce zero-shot learning as a domain adaptation problem based on the probability distributions over the instances [59]. Romera et al. presented the theoretical model using \mathcal{A} -distance as the measurement between source and target distributions. The \mathcal{A} -distance is defined as the total variation or the L^1 norm between the distribution within a given measurable subset \mathcal{A} with the domains [122].

Intuitively, \mathcal{A} -distance shows the most substantial change in the similarity of a given set. The set can be considered based on manual choice or conditional filtering, and we show a conditional filtering set in Table 20. The first conditional set is obtained by filtering higher cosine similarity $\forall(0.5 \leq \theta)$ to understand the highly similar class representatives showcasing the potential confusion between classes. Similarly, the second conditional set is generated by filtering lower cosine similarity $(0 < \theta < 0.5)$, which points of the class representative pair with the least similarity.

$$\begin{aligned}
V_{ks} &= \sup |f(\mathbb{S}) - f(\mathbb{T})| \\
V_{\mathcal{A}} &= 2 \sup_{\mathcal{A}} |f(\mathbb{S}_{\mathcal{A}}) - f(\mathbb{T}_{\mathcal{A}})|
\end{aligned}
\tag{3.10}$$

The Kolmogorov-Smirnov (*KS*) test is used to measure variation across the entire distribution, whereas *A*-distance is used to measure variation across a subset obtained from both the distribution for a given condition. As shown in Equation 3.10, $f(\mathbb{S})$ and $f(\mathbb{T})$ are the distribution functions based on the cosine similarity distribution for the source and target domain, respectively. V_{ks} shows the Kolmogorov-Smirnov distance between the entire source CR distribution and target CR distribution. $V_{\mathcal{A}}$ shows the *A*-distance between the source and target domain on a given subset of data. For $V_{\mathcal{A}}$ and V_{ks} , a lower score means the target is closer to the source, and the higher score means the targets are further away from the source. The higher score indirectly indicates incompatibility between target and source, and the target might perform poorly because of the far distance, i.e., with lower accuracy.

3.5 Experiments

3.5.1 Datasets

The CRL model was evaluated using six different targets (unseen) datasets in two different settings. For our experiments, ImageNet-1K (2015), with 1000 classes [67], was used as the source dataset. The target dataset info is shown in Table 19. We considered four standard zero-shot learning (ZSL) datasets and two classification datasets for

the target dataset. The four ZSL datasets include Animals with Attributes-2 (AWA2), Caltech-UCSD Birds-200 (CUB200), Scene Attribute dataset (SUN_A), and ImageNet-360 (IN360). IN360 includes 360 unique classes present in ImageNet-1K (2010) version, which is different from the source domain dataset, ImageNet-1K (2015) [70, 69, 104]. The two classification datasets are Caltech-101 (C-101), Caltech-256 (C-256).

Table 19: Benchmark Dataset: Seen and Unseen Classes

Dataset	#Class	#Image	Setting-1		Setting-2	
			Seen	Unseen	Seen	Unseen
C-101 [123]	101	8,677	-		1000	101
C-256 [124]	256	30,608	-		1000	256
AWA2 [92]	50	30,475	40	10	1000	50
CUB200 [125]	200	11,788	150	50	1000	200
IN360 [67]	360	2,44,800	1000	360	1000	360
SUN_A [126]	806	14,340	697	109	1000	806

Setting-1: We investigated the effects of the seen and unseen split for Setting-1, similar to work presented by Xian et al. [92]. The seen and unseen data in Setting-1 are prepared for a fair comparison with the state-of-the-art ZSL models. It is noted that the seen and unseen split is mainly for evaluation purposes. We want to highlight that there is no training activity happening with the unseen data in the CRL model since the ImageNet-1K pre-trained models are used as projection function as the default method (refer to Section 3.5.2). The ImageNet-1K pre-trained model should be sufficient for inference with the seen and unseen data for this setting.

Setting-2: The Setting-2 is a unique setup for CRL, where the source domain

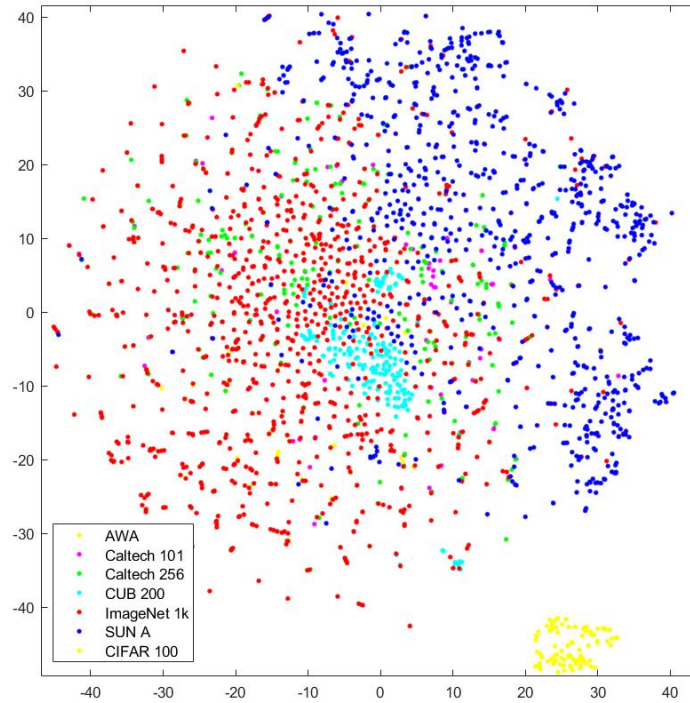


Figure 20: t-SNE Visualization for Class Representatives of Benchmark Datasets [82]

(ImageNet-1K) included in the pre-trained model are considered as seen classes. Typically, the generalized zero-shot learning setting consists of both source and target classes as a class set during inference. For this G-ZSL setting, we argue that ImageNet-1K should be included with the given dataset in the class set as it is present during the learning of projection function (as we are using a pre-trained model). As CRL focuses on no learning for unseen classes, Table 19 Setting-2 shows the ImageNet-1K as seen classes (source domain) and each dataset’s categories as the unseen classes (target domain).

3.5.2 Pre-trained Model as Projection Function

As described in Section 3.3.3, the projection function $P(\cdot)$ mapping the input images into the class representative feature space. For the experiments, we use widely available image classification models pre-trained using ImageNet-1k (2015). MATLAB’s pre-trained deep neural networks [80] were used as projection functions, namely Inception-V3 [76], ResNet101 [77], VGG-19 [78] and GoogleLeNet [127]. One of the CNN models were defined as the projection function (pre-trained with ImageNet-1K) for the CRL experiments. The last convolution layer from the CNN model was considered as the base vector b . The j dimensions are based on the input size of the layer, and the layer’s output dimensions become n dimensions of class representative feature space $CRFS$.

3.5.3 Evaluation Metrics

For zero-shot learning setting (Section 3.3.1), we use the average of class-wise accuracy Acc_T and flat-hit@ k is reported. Equation 3.11 shows that class-wise accuracy is calculated by the average of correct predictions for each class. This evaluation was used to interpret the accuracy of the best performing class and the worst-performing class. flat-hit@ k evaluation is defined as the percentage of the test images for which the model returns the matched label in its top k predictions. It is useful to determine whether the $CRFS$ is facilitating enough to get better accuracy by considering k nearest class representatives to a given instance.

$$Acc_T = \frac{1}{\|T\|} \sum_{c=1}^{\|T\|} \frac{\# \text{ correct predictions in } c}{\# \text{ samples in } c} \quad (3.11)$$

For the generalized zero-shot learning setting, the harmonic mean (H) of the source dataset accuracy (Acc'_S) and the target dataset accuracy (Acc'_T) are reported [89, 92]. Acc'_S is calculated by considering correct predictions source instances by considering the label space to be both source and target ($\mathbb{S} \cup \mathbb{T}$). Similarly, Acc'_T is calculated by considering correct predictions of target instances (Equation 3.12).

$$H = \frac{2 * Acc'_S * Acc'_T}{Acc'_S + Acc'_T}$$

$$Acc'_S : \mathbb{S} \Rightarrow \mathbb{S} \cup \mathbb{T} \tag{3.12}$$

$$Acc'_T : \mathbb{T} \Rightarrow \mathbb{S} \cup \mathbb{T}$$

3.5.4 Model Parallelism and System Specifications

The feature extraction has been conducted class by class as a form of parallel processing to build a CR for each category (Section 3.4.3). The CR generation was implemented parallel with Spark’s resilient distributed datasets (RDDs), a collection of data points partitioned across the nodes of the cluster. The *distribute* function was implemented as the RDD partition with the condition that all data points of a given class are present in the same partition. The CR generation was performed in the map stage, where each partition is independent of each other.

The projection of the data through the pre-trained projection method was implemented on a single GPU, which is Nvidia GeForce GTX 1080 (with 12GB GDDR5X RAM) on MATLAB 2018b version. The CR generation and CR-based inference were

implemented using Spark 2.4.3 version [79]. The parallel and batch process was conducted through the RDD based parallelism on a single CPU with 4GHz Intel Core i7-6700K (quad-core, 8MB cache, up to 4.2GHz with Turbo Boost) and 32GB DDR4 RAM (2,133MHz) (i.e., local parallelism of 4 cores).

3.6 Results and Discussion

3.6.1 Domain Adaptation

Table 20: Source Domain and Target Domain: Accuracy, Kolmogorov-Smirnov Test Scores, \mathcal{A} -distance

Target Domain		Source & Target Comparison		
Dataset	Acc_T	V_{ks}	$V_{\mathcal{A}}$ $\forall \theta(0.5 \leq \theta)$	$V_{\mathcal{A}}$ $\forall \theta(0 < \theta < 0.5)$
ImageNet-1K	73.7%	0	0	0
C-256	70.5%	0.0921	0.0908	0.1132
C-101	91.2%	0.1570	0.1350	0.2767
SUN_A	31.9%	0.3560	0.1398	0.6205
AWA2	76.8%	0.4120	0.2848	0.7228
IN360	38.1%	0.4580	0.5877	0.9009
CUB200	40.1%	0.4740	0.9737	0.2450
CIFAR-100	57.9%	0.9125	1.6429	1.3115

The Kolmogorov-Smirnov Test and \mathcal{A} -distance are used to identify the type of transfer learning [81] that happened when during the zero-shot learning process from a given source dataset and target dataset. For this experiment, Inception-V3 based class representatives were considered. Each class representative was generated using ten labeled instances. Homogeneous transfer learning happens when the source and target feature

spaces have the same attributes, labels, and dimensions. In this experiment, ImageNet-1K is identified as homogeneous transfer learning with V_{ks} and $V_{\mathcal{A}}$ scores as zero. Note that source to source mapping is not typically considered as transfer learning. The CRL model takes just feature space established based on a pre-trained projection method and uses a different inference method. Thus, the CRL model in ImageNet-1K is identified as homogeneous transfer learning. Heterogeneous transfer learning happens when the source and target domains share limited or no features or labels, and dimensions of the feature space differ as well. All the target domains can be considered heterogeneous transfer learning when no or little label overlaps with the source domain, i.e., ImageNet-1K. The critical observation is if the heterogeneous transfer learning negatively impacts the target domain performance, that brings the issue of negative transfer. Negative transfer learning happens when the target domain's performance has negative implications on knowledge transfer from the source domain. The negative transfer learning is generally found when the source domain has minimal similarity with the target domain. The KS test and \mathcal{A} -distance is used to identify if the target domain has a negative transfer.

With the highest score in all variations, i.e., V_{ks} and both $V_{\mathcal{A}}$, the dataset CIFAR-100 has the negative transfer. Figure 20 shows the target domain data are projected on the semantic space that is quite distinct from the source domain. Although the CIFAR-100 is semantically relevant to other datasets, the CRL space of CIFAR-100 is divergent from the source space in terms of image modality, such as image quality and image size. The size of CIFAR-100 images is [32x32] while one of the source domain ImageNet-1K [400x400]. More specifically, the dimension of the projection method Inception-V3 is

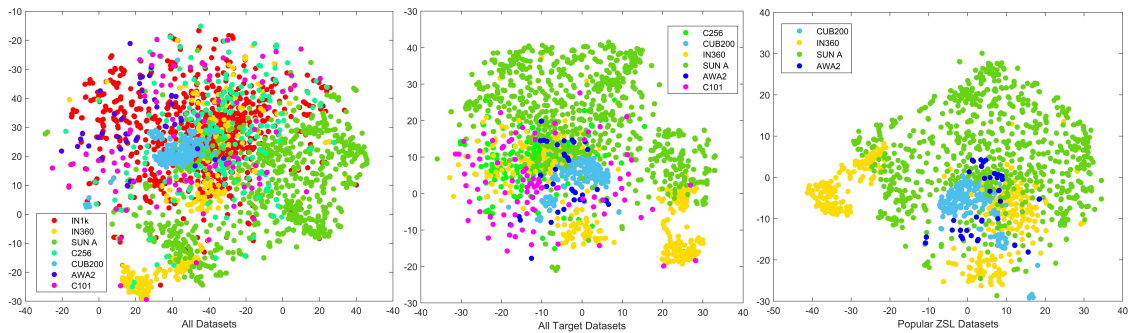


Figure 21: t-SNE Visualization of Class Representatives [82]

[299x299].

Figure 22 demonstrates the similarity distribution in the feature space of the source and target datasets. This figure further confirms the existence of negative transfer. The next dataset, which might have negative performance, is CUB200, which a score of 0.9737 on cosine similarity greater than 0.5. This indicates that the distribution created by higher similar class representatives in CUB200 is very different from ImageNet-1K. This distribution disparity can also be observed in Figure 21, where CUB200 forms a highly dense cluster in all the figures. It supports the nature of the dataset as CUB200 is specific to bird categories, whereas ImageNet-1K has a variety of animate and inanimate objects.

Table 20 shows that the accuracy of datasets does not perfectly correlate with the scores V_{k_s} and $V_{\mathcal{A}}$. This lack of correlation is due to the size of the dataset, i.e., several classes. Figure 23 shows the class-wise distribution of the dataset, and the star marker indicates the number of classes. The SUN_A dataset’s performance is dependent on multiple factors such as the size of the dataset, the standard deviation of the class-wise accuracy, and the domain compatibility score. The standard deviation is class-wise accuracy, and

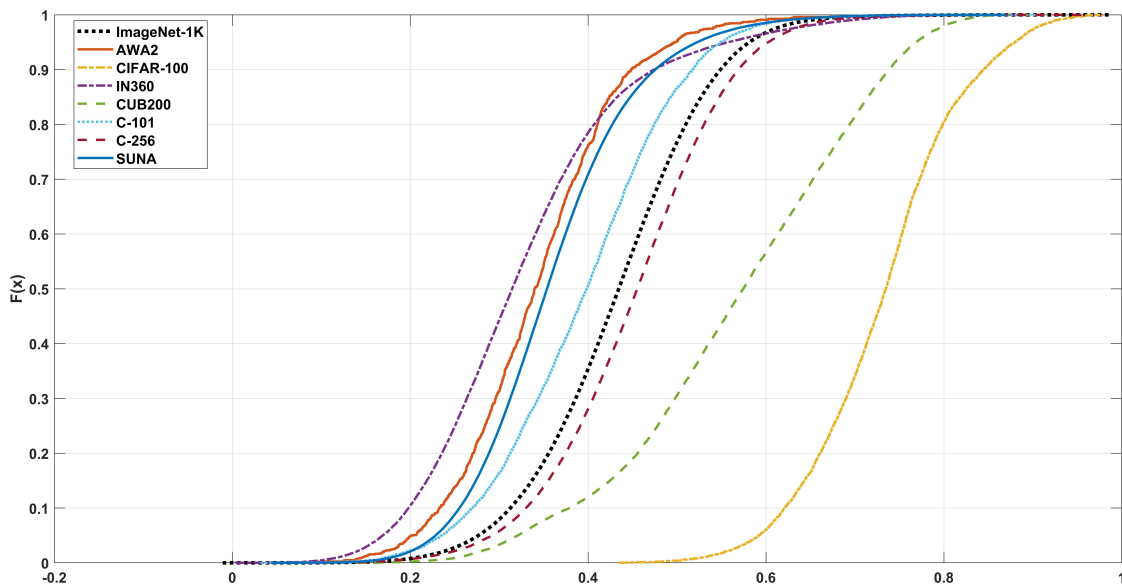


Figure 22: Cosine Similarity Distribution of 8 Benchmark Datasets

the significant difference between two V_A can be correlated. This inconsistency is due to a lack of quality images for certain classes. Figure 24 shows an example of two best performing and two worst-performing classes.

3.6.2 Comparison with ZSL Algorithms

We have evaluated the CRL model in terms of the three perspectives, such as ZSL performances with the six different benchmark datasets, ZSL performance with an increasing number of instances, and comparison with the state-of-the-art ZSL algorithms. The two types of CRL models (the projection methods) was included Inception-V3 based model and VGG-19 based model. Table 21 and Figure 25 show the CRL’s ZSL performance with flat-hit@ k using Inception-V3 projection with the dataset under Setting-2. Table 22 shows the comparison of the state-of-the-art zero-shot learning algorithms using

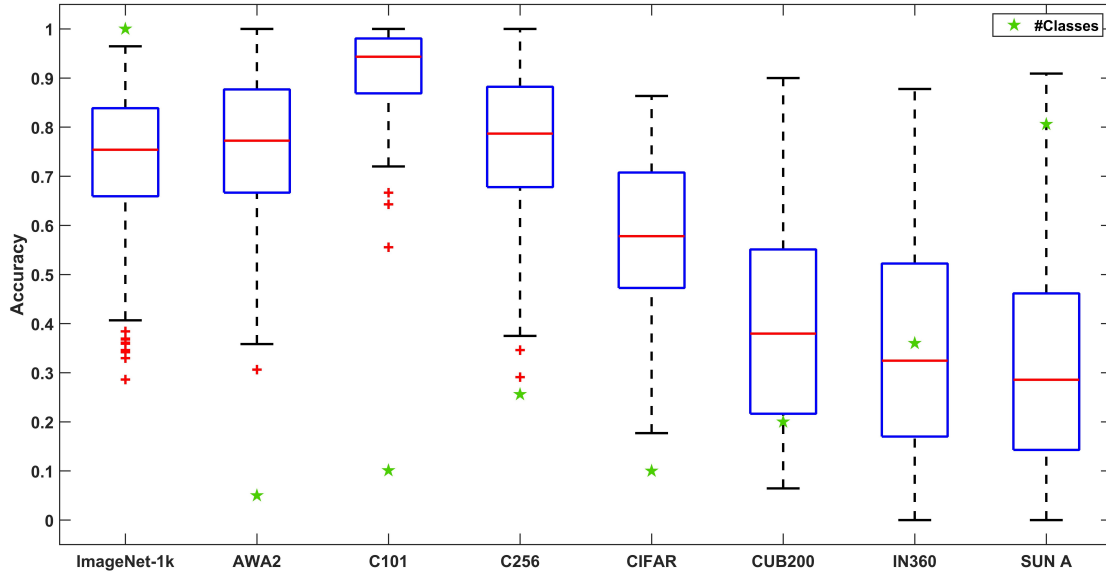


Figure 23: Accuracy Distribution of 8 Benchmark Datasets

the VGG-19 model, considering just the IN360 dataset under Setting-2.

The experiments show the CRL model’s performance for the ZSL task that recognizes the target (unseen) labels without having the source (source) labels. Table 21 shows two versions of the recognition tasks with the testing data from the target set (\mathbb{T}); $\mathbb{T} \Rightarrow \mathbb{T}$ when the testing label could be only from the target set $y^* \in \mathbb{T}$ and $\mathbb{T} \Rightarrow \mathbb{S} \cup \mathbb{T}$ when the testing label could be from both the source set and target set $y^* \in \mathbb{S} \cup \mathbb{T}$. For this experiment, we consider all instances (70%) from the dataset to generate class representatives. The results show the influence of an increasing number of labels in the dataset to the ZSL accuracy similar observation was made from Figure 23. When comparing the Flat-hit@ k with $k = 1$ (Top-1) and $k = 2$ (Top-2), we can see an average of 21% increase from












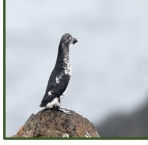
SUN_A		AWA2		CUB200	
	Mosque Indoor 16.7%		Walrus 19%		Cliff Swallow 4.7%
	Lean-to 16.7%		Mouse 30%		Chipping Sparrow 4.7%
	Aircraft Carrier 100%		Giant Panda 100%		American Goldfinch 88%
	Badminton Indoor Court 100%		Lion 100%		Least Auklet 89.6%

Figure 24: Inference Performance for Two Best and Two Worst Cases

Top-1 accuracy to Top-2 to accuracy. This increase signifies that *CRFS* provides a well-formed neighborhood, with a 21% increase chance of getting the correct prediction at the second nearest class representative. Comparing the two different recognition tasks, we note the significant drop in efficiency is shown when the source set was also considered. Note that these results were based on Setting-2, source set is ImageNet-1K. The number of classes in $\mathbb{S} \cup \mathbb{T}$ would be a minimum of 1050 (case of AWA2 dataset) and a maximum of 1806 (case of SUN_A dataset). A negative correlation between the number of classes and accuracy was observed.

Figure 25 shows the performance of CR-Inception-V3 with the images at a specified number ranged from one to ten from each class. For this experiment, we considered

Table 21: Accuracy for Zero-Shot Learning Tasks (Setting-2)

Dataset	Recognition Task Accuracy		
	Flat-hit@K	$\mathbb{T} \Rightarrow \mathbb{T}$	$\mathbb{T} \Rightarrow \mathbb{S} \cup \mathbb{T}$
C-101	1	91.2%	86.4%
	2	97.4%	93.5%
	5	98.5%	97.0%
C-256	1	70.5%	55.6%
	2	78.3%	69.2%
	5	84.7%	78.7%
AWA2	1	76.8%	48.6%
	2	87.9%	72.9%
	5	95.5%	86.5%
CUB200	1	40.1%	38.4%
	2	53.9%	52.5%
	5	71.0%	69.9%
SUN_A	1	31.9%	28.5%
	2	43.3%	40.4%
	5	58.7%	56.3%
IN360	1	38.1%	28.3%
	2	47.6%	40.1%
	5	59.7%	54.3%

only ($\mathbb{T} \Rightarrow \mathbb{T}$) setting. Interestingly, the Top-1 accuracy with just ten images reaches higher than 75% accuracy compared to the accuracy reported in Table 21, which considers 70% of the data.

Table 22 shows the state-of-the-art zero-shot learning (ZSL) algorithms to be compared with CRL in Table 17. This experiment considered the class representative model built using pre-trained VGG-19 as a projection method with ImageNet-1K as a source and IN360 as the target. Table 22 shows that the performance of the CRL model is superior

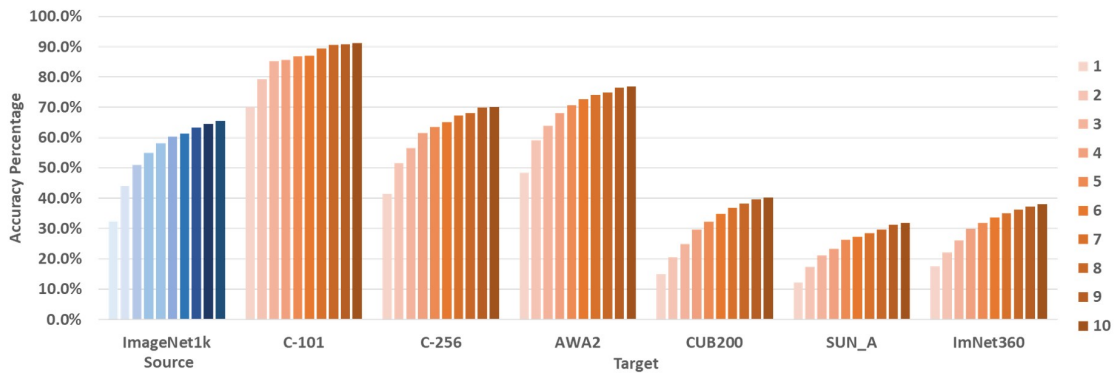


Figure 25: Accuracy on CRL Model based on Increasing Instances in Setting-2 ($\mathbb{S} \Rightarrow \mathbb{S}$ and $\mathbb{T} \Rightarrow \mathbb{T}$)

in both cases of 3000 instances and all instances. In the 3000 instance case, the CRL model's Top-1 shows a 27% increase compared to the top performer, Deep WMM-Voc. In all cases, the CRL model's Top-1 accuracy is significantly higher than the others; on average, the CRL model's Top-1 accuracy is considerably higher than Deep WMM-Voc's.

Table 22: Comparison between the CRL model with VGG-19 Model for IN360 (Setting-1)

ZSL Type	Methods	3000 Instances		All Instances	
		Top-1	Top-5	Top-1	Top-5
Projection Method	CRL (ours)	11.78%	25.52%	31.6%	55.1%
	DWV [69, 70]	9.26%	21.99%	10.29%	23.12%
	SAE [99]	5.11%	12.26%	9.32%	21.04%
	Deep-SVR [61]	5.29%	13.32%	5.7%	14.12%
	Embed [100]	-	-	11.0%	25.7%
	ConSE [62]	5.5%	13.1%	7.8%	15.5%
Correspondence	ESZSL [59]	5.86%	13.71%	8.3%	18.2%
	DeViSE [57]	3.7%	11.8%	5.2%	12.8%
Relationship	AMP [58]	3.5%	10.5%	6.1%	13.1%

*Results reported for SOTA Models are from Fu et al.[69]. The CRL model is configured with the same settings such as VGG-19 with 3000 instances, i.e., 3 images per class and all 50000 instances, i.e., 50 images per class

Table 23: Accuracy of Generalized Zero-Shot Learning Algorithms with Setting-1

ZSL Type	Model	AWA2 (Top-1)			CUB200 (Top-1)			SUN_A (Top-1)			IN360 (Top-5)		
		Acc _S '	Acc _T '	HM	Acc _S '	Acc _T '	HM	Acc _S '	Acc _T '	HM	Acc _S '	Acc _T '	HM
Instance-based Projection Method	CRL-IN (Ours)	86.0%	90.8%	88.3%	73.3%	70.4%	71.8%	39.7%	41.2%	40.5%	89.4%	87.5%	88.4%
	CRL-RL (Ours)	87.4%	83.5%	85.4%	47.5%	54.7%	50.8%	37.8%	41.7%	39.6%	75.5%	70.9%	73.1%
	Embed [100]	84.7	32.8	47.3	57.9	19.6	29.3	34.3	20.5	25.7	72.2	22.6	34.4
	SAE[99]	71.3	31.5	43.7	36.1	28.0	31.5	25.0	15.8	19.4	89.4	21.8	35.1
	CMT [98]	86.9	8.4	15.3	60.1	4.7	8.7	28.0	8.7	13.3	70.2	11.3	19.5
Instance-based Synthesizing Method	BPL+LR[104]	69.4	60.9	64.9	71.6	42.7	53.5	36.9	39.2	38.0	95.5	26.1	41.0
	CADA-VAE [102]	72.8	57.3	64.1	53.5	51.6	52.5	35.7	47.2	40.7	74.6	25.2	37.7
	cycle-GAN [103]	64.0	56.9	60.2	61.0	45.7	52.3	33.6	49.4	40.0	81.3	24.4	37.5
	GAN+Softmax [101]	61.4	57.9	59.6	57.7	43.7	49.7	36.6	42.6	39.4	79.1	24.2	37.1
	GAN+AIE [101]	57.2	47.6	52.0	59.3	40.2	47.9	31.1	41.3	35.5	78.8	23.4	36.1
Classifier-based Relationship Method	SynC [106]	87.3	8.9	16.2	70.9	11.5	19.8	43.3	7.9	13.4	94.3	10.7	19.2
	SSE [105]	80.5	7.0	12.9	46.9	8.5	14.4	36.4	2.1	4.0	84.8	10.8	19.2
Classifier-based Correspondence Method	SP-AEN [107]	90.9	23.3	37.1	70.6	34.7	46.5	38.6	24.9	30.3	84.8	20.4	32.9
	AIE [91]	76.1	16.8	27.5	62.8	23.7	34.4	33.1	21.8	26.3	72.4	22.1	33.9
	DeViSE[57]	68.7	13.4	22.4	53.0	23.8	32.8	27.4	16.9	20.9	68.9	21.8	33.1
	SJE [56]	74.6	11.3	19.6	59.2	23.5	33.6	30.5	14.7	19.8	70.1	19.5	30.5
	LATEM [60]	71.7	7.3	13.3	57.3	15.2	24.0	28.8	14.7	19.5	77.3	18.8	30.2
	ESZSL[59]	75.6	6.6	12.1	63.8	12.6	21.0	27.9	11.0	15.8	69.1	16.5	26.6

Setting-1: AWA2: Seen Class#: 40 and Unseen Class#: 10, CUB200: Seen Class#: 150 and Unseen Class#: 50

SUN_A: Seen Class#: 697 and Unseen Class#: 109, IN360: Seen Class#: 1000 and Unseen Class#: 360

Our Results- CRL-IN: Inception V3, CRL-RL: ResNet-101 for AWA2, CUB200, SUN_A & LeNet for IN360

Other Results reported for SOTA G-ZSL Models are from Guan et al. [104]

3.6.3 Generalized Zero-Shot Learning

Table 23 shows the performance of the generalized zero-shot learning (G-ZSL) model comparing CRL (built on Inception-v3) to the four state-of-the-art G-ZSL models. The SOTA models for AWA2, CUB200, and SUN_A were built on ResNet-101, and the SOTA model for IN360 was built on Google-LeNet. The number of images for the G-ZSL model datasets is as follows: 600 images per class for AWA2, 50 images per class for CUB200, 20 images per class for SUN_A, and 180 images per class for IN360. The number of images is the same as the setting specified by Guan et al., for fair evaluation [104].

Table 23 also shows the accuracy of the target (unseen) and harmonic mean of the CRL model outperforms the other SOTA models, including the BPL+LR model, which includes synthesized data. On average, the accuracy of CRL was 20% better than the other SOTA models. Compared to a fine-grained dataset, the CRL model performs better in the coarse datasets, such as SUN_A with only 40.5% of the Harmonic Mean Accuracy. The CRL accuracies for the seen classes have been improved with all datasets excluding IN360. For the IN360 dataset, BPL+LR shows the highest accuracy of 95.5%, and CRL shows the second-highest accuracy of 89.4%. The pattern of source accuracy being better than target accuracy can be observed in works in instance-based projection method, classifier-based correspondence method, and classifier-based relationship method. Reportedly difference in accuracy between source and target domain is around 60% in those three types except CRL. CRL Model is an instance-based projection method that still reports an accuracy equivalent to the instance-based synthesizing method. In most cases,

the CRL model outperforms the Synthesizing Methods. The most significant advantage of CRL compared to the synthesizing method is that it does not use any complex and time consuming generative or auto-encoder models.

3.6.4 Time Performance

The CRL model’s performance is evaluated by comparing it with the Inception-V3 pre-trained model retrained with the target domain. The CRL model’s performance is calculated according to the projection time (Section 3.3.3), CR model generation time (refer to Section 3.3.4) and CR-based inference time (refer to Section 3.4). For this experiment, we use Inception-V3 as our pre-trained projection. The class representative for this experiment was built with 70% of the given dataset. Note that since most of the zero-shot learning methods do not report time performance, image classification setting was considered. The CRL image classification setting is the same as the CRL ZSL setting ($\mathbb{T} \Rightarrow \mathbb{T}$) since no learning happens for either of them. Our previous work [90] reports more details on the comparative evaluation of the CRL model with others. The comparison is with the Inception-V3 pre-trained model from MATLAB (same as CRL’s projection method), where the last layer of softmax is retrained with the new dataset.

Table 24 shows the comparison of the CRL model’s overall time vs. the time taken for retraining the dataset using the Inception-V3 pre-trained model. Both pre-trained models were run on the same system specification. Pre-training of the Inception-V3 Model was stopped at a reported number of epochs as the time taken was significantly higher than the CRL models. The CRL model with three datasets (CalTech-101, CalTech-256,

and CIFAR-100) has an average of 99% time reduction that is a significantly reduced compared with that for the original Inception-V3 model. Within the same time window and based on the same pre-trained model, the Inception-V3 model performance has not reached the accuracy published in [33]. The CRL model’s overall time would show genuinely outstanding achievements in the target domains, even if the models were never learned from the target domain.

Table 24: Transfer Learning Performance Analysis: CRL vs. Inception-V3 [80]
Pre-trained Model: Inception-V3 with ImageNet-1K

Target	CRL				Inception-V3 [80]		
	Step	Time (minutes)		Acc_T	Time (minutes)	Epoch	Acc_T
C-101	Projection	5.35	7.2	94.4%	4257.91	40	88.7%
	CR Generation	1.51					
	Inference	0.33					
C-256	Projection	10.23	13.88	78.2%	14193.96	14	59.3%
	CR Generation	1.9					
	Inference	1.75					
CIFAR-100	Projection	13.2	15.73	57.9%	26941.85	10	50.3%
	CR Generation	0.93					
	Inference	1.6					

3.7 Discussion

The class representative generation was obtained by extracting the abstraction of each feature’s distribution in the class representative feature space $CRFS$ for the target domain. For the purpose, we used a straight-forward aggregation approach. Thus, the class representative learning model might be susceptible to outliers, sample size bias, and hubness. The CRL model was extremely strong at the flat-hit@ k with $k = 2$ & $k = 5$

compared to $k = 1$ (Table 21). This indicates that the high similarity between some CRs might lead to misclassification. To overcome the limitation of the CR generation, an advanced optical model such as the fisher vector and gaussian mixture model might be incorporated in the future. We will consider unsupervised deep learning techniques to learn efficient data codings and reduce the CR's feature space to a more optimal representation. We can further extend it to determine the CR vectors' common and unique features and find the weights that maximize the uniqueness between CRs. Currently, CRL is mainly based on the use of visual data only for zero-shot learning. We will extend the CRL model to handle multi-modal data distributions such as text data in the future. The CRL model has a potential extension to have the open set recognition with $\mathbb{T} \gg \mathbb{S}$. Currently, CRL has an openness factor of 0.278 [88].

3.8 Conclusion

This chapter proposes the class representative learning (CRL) that projects the abstract features extracted from a deep learning environment to the high-dimensional visual space. In the CRL model, class representatives (CRs) are designed to represent potential features for given data from the abstract embedding space. A projection-based inferencing method is intended to reconcile the dominant difference between the seen classes and unseen classes. The CRL model has three distinct advantages than existing ZSL approaches. (1) There is no dependence among CRs to be built in parallel and used freely, depending upon the context. (2) Unlike other ZSL approaches, the CRs can be generated only using

the abstract visual space by eliminating the need for semantic spaces or auxiliary information. (3) The abstract embedding space of the source (seen classes) is solely used to project the instances of the target (unseen classes) without any learning involved. The current research demonstrated the benefit of using the class-based approach with class representatives for ZSL and G-ZSL on eight benchmark datasets. Extensive experimental results confirm that the proposed CRL model significantly outperforms the state-of-the-art methods in both ZSL and G-ZSL settings. The CRL model is presented herein as an instance-based projection-method zero-shot learning method, but surprisingly this outperforms complex state-of-the-art instance-based synthesizing methods. The work presented in this chapter was published as part of Chandrashekar et al.[128].

CHAPTER 4

MCDD: MULTI-CLASS DISTRIBUTION MODEL FOR LARGE SCALE CLASSIFICATION

Data level parallelization or distribution has become the new normal among the proposed machine learning or deep learning algorithms. The model built from a large amount of data is singular but not decomposed. One of the significant problems in machine learning and deep learning with big data is that it is becoming a complex and bigger model. As shown [19], an increase in either the number of layers or the number of parameters in a deep learning model leads to better accuracy. The well-known deep neural network models include AlexNet [129], VGG [78], GooGleNet [127], ResNet [77], and Inception-Resnet [130]. They mainly focus on large and very complex networks to improve accuracy by adding additional layers or combining other networks to existing network models.

As evaluated in [131], ensemble learners could be an effective mechanism to combine and scale the deep learning classification model as a base learner. Some ensemble learners, such as Random Forest [132] and Ada Boost [133], showed improvement in classification accuracy compared to using a single classifier in terms of data distribution and model distribution. Ju et al., [131] introduced advanced ensemble learners, including Base Learner, Super Learner, Bayes Optimal Classifier, Unweighted Average, and Majority Voting in the Deep Neural Network setting. Considering Random Forest, Multiple

Models (trees) are built based on randomly distributed data and a simple voting paradigm as a decision making process.

Here is the limitation of these algorithms: (1) the randomness in the data distribution; (2) the model has no structure; (3) there is no distributed decision making on the classification (i.e., if there are ten classes, they still decide on ten classes). Due to these issues, the computation cost is quite expensive. There is no significant improvement as there are still a large number of classes to be classified.

This chapter proposes a distributed classification model called multi-class discriminative and distribution (MCDD), focusing on building multiple models in a distributed and parallel manner. The motivation of this work is how a model can be partitioned into several various sub-models. Depending upon the need, a more prominent model can be composed. The smaller models can be organized in a hierarchical manner (divide-and-conquer) to produce the most optimal recognition for any given input. The effectiveness of the MCDD model was shown with the CNN based image classification and segmentation on multiple large datasets, Caltech-101 [123], CIFAR-100 [134], ImageNet-1K [135]. The MCDD-based distribution model shows a better performance than the state-of-the-art performance.

Our primary contributions in this chapter are as follows.

- We define a measurement called *confusion factor* that computes a misclassification cost and detects communities with high classification errors and build a deep CNN model to over the limitations.

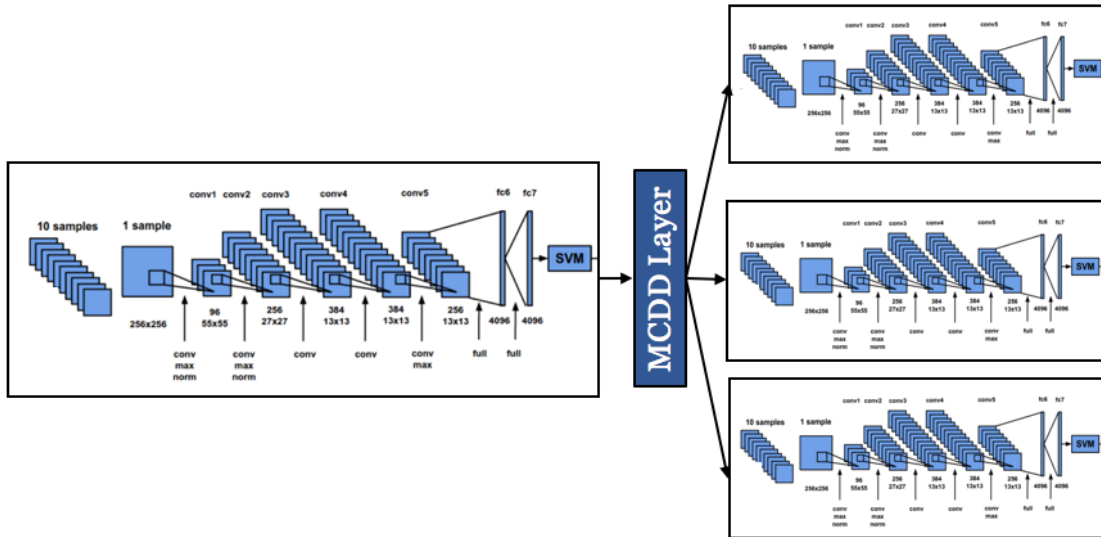


Figure 26: MCDD Deep Learning Networks

- We develop a hierarchical clustering algorithm to break the "confusing" communities down into smaller communities. This clustering will result in minimizing the confusion factor of the communities and maximize the classification accuracy.

The MCDD networks are designed as a hierarchical deep learning network that is composed of multiple AlexNets [129] as shown in Figure 26 and it is also evaluated with multiple datasets, Caltech-101 [123], CIFAR-100 [134], and ImageNet-1K [135].

4.1 Related Work

4.1.1 Classification Measurements

A distance metric [136] was proposed to find the most influential neighbors and approximate their influence on classification. However, they cannot detect the classes of the misclassification. The visualization methods for representing models of misclassified cases were developed in [137] and [138]. A method for improving the performance of classification in small datasets such as MNIST in [139]. Recently, the accuracy improvement in large-scale classification using CIFAR-100 [134] and ImageNet [135] using a graph-based tool in [72]. Even if this work presented community detection based on the confusion graph [72], they are limited to finding an optimal solution for significant improvement in the classification.

4.1.2 Hierarchical Classification

The hierarchical structure was proved to be useful in the multi-class classification problem. In the works [140, 141, 142], similarity priors were analyzed to determine if the nodes in classifiers are too close to nearby nodes. Similar to our work, a dissimilarity constraint was used to differentiate nodes from their ancestors [143, 144]. The weighted classification error model was proposed to assign different nodes according to the hierarchical loss of model learning [145, 146, 147].

Hierarchical class structures were introduced to improve the performance of deep networks. In [148], similar classes were grouped and classified with an augmented deep network. In [149], a hierarchical convolutional neural network (HD-CNN) was designed

based on the two-level organization of coarse and fine-grained categories by sharing common layers. In [135], common and discriminative features were analyzed across classes and learned for representing the feature in pooling layers of a hierarchical structure.

4.1.3 Classifier Structure

Hierarchical models have been built for multi-class classification based on confusion matrices obtained from SVM classifiers [150, 151, 152], constructed as a binary branch tree in [151], a label-embedding tree in [150], and a probabilistic label tree in [152]. Furthermore, a relaxed hierarchical structure was introduced in [153] for allowing the confusion classes to belong to more than one node in the hierarchy. In contrast, visual trees were constructed by clustering techniques [154, 155, 156, 157], AP clustering by [157], and Spectral clustering by [155]. Clustering methods are intuitive and efficient. However, they may not provide a higher classification accuracy compared to classification methods.

The greedy learning method is typically utilized for class prediction. Most hierarchical classification approaches [157, 158, 159, 43] are greedy learning by making predictions in each layer for maximizing the classification probability. However, greedy learning-based inferences may produce propagated errors at consequent layers (i.e., a prediction error at the lower level will provide ones at a higher level).

These works mainly focus on the similar or the same features that are shared by all classes. On the contrary, our proposed MCDDNet enforces to group classes that have various features instead of having a similar feature. Recently, SplitNet [160] addressed

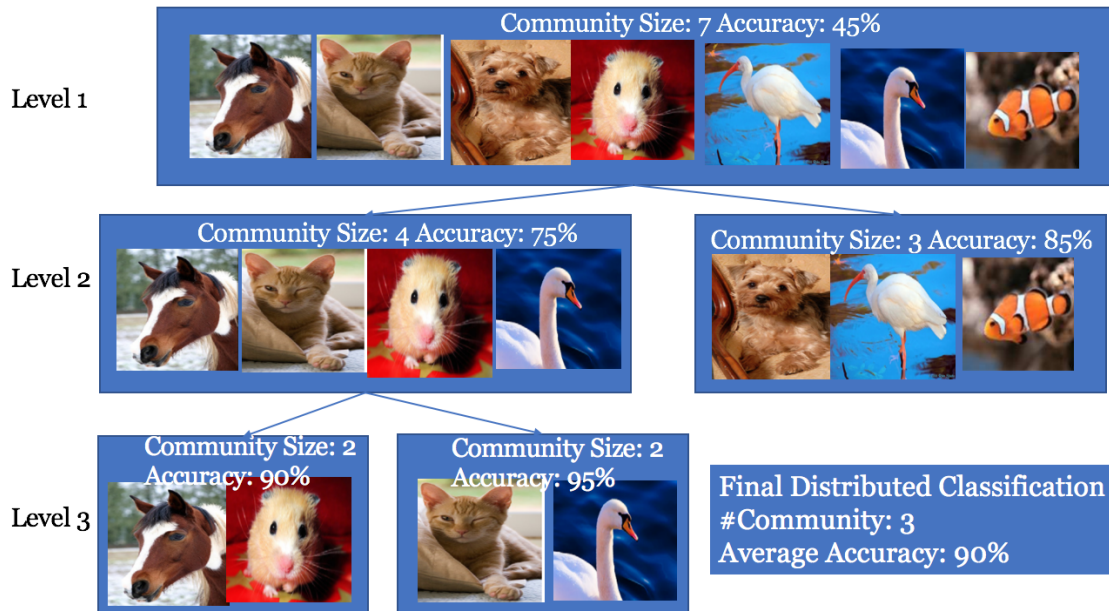


Figure 27: Example of the MCDD Distribution and Classification

the problem of splitting classes or features for sharing common features within a group of related classes and results in reducing the number of parameters and computation cost for large-scale problems. In SplitNet, the networks are partitioned into subnetworks according to the semantic taxonomy, and they are assigned and trained for groups. On the other hand, semantic knowledge of class relatedness is not well-fitted for a distributed machine learning setting to learn a network structure that is well-fitted for a distributed machine learning setting.

4.2 Multi-Class Discriminative Distribution Model

We design a new model, called the *Multi-Class Discriminative Distribution Model* (MCDD), for maximizing the classification accuracy in large scale multi-class classification. For the purpose, classes will be clustered based on the baseline levels of confusion factors among multiple classes, as shown in Figure 27. In the MCDD framework, the following properties are satisfied:

- The confusion matrix should present the representative confusion between the classes in the classification.
- The members of a community (classifier) should be as diverse as possible.
- The average accuracy of the child-level classifiers should be higher than the parent-level classifier's accuracy.
- The singleton community (the size of the community is one) is prohibited. Thus, during the clustering, singletons are forced to team up with others.
- The communities should be mutually exclusive and collectively exhaustive.
- The confusion between classes within a community during the multi-class classification should be minimized so that the classification accuracy could be maximized.

Unlike the recent work [72], we propose a novel concept of a diversity community detection and a community clustering model. The MCDD framework will minimize the confusing factors among classes using a deep learning hierarchical model and improve

the overall and community-wide classification accuracy using a hierarchical deep learning model (shown in Figure 26).

4.2.1 Confusion Measures

The *confusion factor* (β) measures the degree of confusion among the classes that are misclassified by the classifier. The confusing factor of classes is determined with *false-positive* instances and *false-negative* instances of the confusion matrix. We assume that if two classes are similar, then their confusion factor is high. Also, if there are high instances of *false-positive* as well as *false-negative*, the *confusion factor* is high.

The confusion factor is defined by a harmonic and integrated model of the false-positive factor and the false-negative factor for a given confusion matrix. The confusion matrix (CM) is an n by n matrix CM , where the each entry, CM_{ij} , where $1 \leq i, j \leq n$ has the percentage of correctly or incorrectly classified cases (i.e., true-positive, false-positive, false-negative, true-negative) as follows:

- true-positive (TP): M_{ij} is correctly classified of class i which is classified as class j
($i = j$)
- false-positive (FP): M_{ij} is misclassified of class i which is actually a class j .
- false-negative (FN): M_{ij} is misclassified of class j which is actually a class i .
- true-negative (TN): M_{ij} is correctly classified of class j which is actually a class j
($i \neq j$)

We design a probabilistic matrix called the confusion factor (CF) that will be used to detect a homogeneous community (i.e., high confusion factors among classes in the community) and partition such a community into smaller ones. The CF matrix is generated by computing the confusion factor using

- Confusion factor between false-negative (FN) and true-positive (TP)
- Confusion factor between false-positive (FP) and true-positive (TP) for a given confusion matrix CM.

More specifically, the false-negative Confusion Factor ($FN - CF_{ij}$) between classes c_i and c_j in CM (CF_{ij}) is the confusion factor of classes c_i and c_j . The false-positive confusion factor ($FP - CF_{ij}$) between classes c_j and c_i in CM (CF_{ji}) is the confusion factor of classes c_i and c_j .

The Confusion Factor (CF) represents the confusion measure between classes c_i and c_j in the CM matrix computed by the following formula:

$$CF_{ij} = \sqrt{(CM_{ij} - CM_{jj})^2 + (CM_{ji} - CM_{jj})^2} \quad (4.1)$$

where $1 \leq i, j \leq n$. In Equation 4.1, CM_{ij} is a false-negative (FN) case, CM_{jj} is a true-positive (TP) case, and CM_{ji} is a false-positive (FP) case.

The CF matrix is represented where $m = n$ as follows:

$$CM_{m,n} = \begin{pmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,n} \\ m_{2,1} & m_{2,2} & \cdots & m_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{m,1} & m_{m,2} & \cdots & m_{m,n} \end{pmatrix}$$

Table 25 and Table 26 show an example of confusion matrix with five classes and their confusion factor matrix, respectively. The number of total instances in the matrix is 25. In Table 26, the confusion factor for two classes, Beaver (B_e) and Airplanes (A_i) is computed as follows: $CF_{A_i, B_e} = CF_{B_e, A_i} =$

$$\sqrt{(CM_{B_e, A_i} - CM_{A_i, A_i})^2 + (CM_{A_i, B_e} - CM_{A_i, A_i})^2} = \sqrt{(0 - 1)^2 + (4 - 1)^2} = 3.16.$$

Table 25: Example of Confusion Matrix

	<i>Ac</i>	<i>Ai</i>	<i>An</i>	<i>Ca</i>	<i>Be</i>
<i>Ac</i>	4	0	0	0	1
<i>Ai</i>	0	1	0	0	4
<i>An</i>	0	1	2	1	1
<i>Ca</i>	0	1	1	2	1
<i>Be</i>	0	0	1	0	4

Table 26: Example of Confusion Factor Matrix

	<i>Ac</i>	<i>Ai</i>	<i>An</i>	<i>Ca</i>	<i>Be</i>
<i>Ac</i>	0	4.12	5.66	5.66	5
<i>Ai</i>	4.12	0	1	1	3.16
<i>An</i>	5.66	1	0	1.41	1.41
<i>Ca</i>	5.66	1	1.41	0	2.83
<i>Be</i>	5	3.16	1.41	2.83	0

Table 27: Example of Normalized Confusion Factor Matrix

	Ac	Ai	An	Ca	Be
Ac	100%	67%	100%	100%	86%
Ai	67%	100%	0%	0%	46%
An	100%	0%	100%	45%	9%
Ca	100%	0%	45%	100%	39%
Be	86%	46%	9%	39%	100%

The Normalized Confusion Factor (NCF) matrix is the normalization of the Confusion Factor (CF) using Equation 4.2.

$$NCF_{ij} = \frac{CF_{ij} - Min_{ij}}{Max_{ij} - Min_{ij}} \quad (4.2)$$

where $Min_{ij} = \min_{1 \leq i \leq n, 1 \leq j \leq n}$, $Max_{ij} = \max_{1 \leq i \leq n, 1 \leq j \leq n}$, $1 \leq i, j \leq n$.

In Table 27, the normalized confusion factor ($NCF_{Be, Ai}$) for two classes, Beaver (B_e) and Airplanes (A_i) is computed as follows: $NCF_{A_i, B_e} = \frac{3.16-1}{5.66-1} = 46\%$.

In Confusion Factor Matrix Computation, A_c : Accordion, A_i : Airplane, A_n : Ant, C_a : Cannon, B_e : Beaver

4.2.2 Community Detection

In this chapter, we design our community detection algorithm to find out the community having a high number of confusing classes from the classification. A community having such classes will be detected by an unbalanced distribution factor in the confusion space. Several primary parameters (including the community classification performance, the size of the community (δ), and the confusion factors (β) of the classes in the community) need to be determined for the partitioning of the community into smaller diverse

communities.

We used the standard deviation as a measurement to check how the members are spread out in the community distribution. Each of the confusion factor values in the distribution deviates from the center of the community. We first compute (1) the subtract the center's confusion factor from each number's, (2) square the results (the squared differences), (3) compute the average of those squared differences (variance), (4) take the square root of the variance (standard deviation). We utilize simple statistic testing such as analysis of variance (ANOVA) to validate "variation" among and between communities by comparing means of communities for statistical significance. Precisely, we first compute the confusion factor for a given community. Then we use the p -th percentile ($0 < p < 100$) as the cutting point to detect the large confusing community. Any community whose confusion factor is less than the p -th percentile will be partitioned. From this analysis, we will find significant differences in the members' performance between the best and worst communities.

We define two distinguished classes that influence the classification accuracy of a community: *Trouble maker* and *Superstar*.

- *Trouble maker (TM)* is a class that causes difficulty or problems in classification, especially that results into degrading the classification performance of the community. It can also be considered as class which has lower individual accuracy and is also highly confused with other classes.

$$TM(C) = c_i > 75\% * Avg(C) \text{ where } c_1 \in C \quad (4.3)$$

The purpose of the clustering in MCDD is quite different from other clustering methods. We are clustering classes in terms of diversity between them. In other words, the classes in a community formed after the cluster have classes that differ so that the confusion that existed during the classification.

Each model in MCDD will show the minimum confusion factor so that MCDD supports the high learning performance for both the community-based classification and the overall classification. The HKM supports the hierarchical distribution of multiple categories according to the MCDD model for the multi-class classification problem.

The HKM algorithm aims to minimize a confusion factor between categories and to find an optimal distribution of these categories. The HKM algorithm was designed by extending the KM algorithm for the construction of classification models in a hierarchical manner. The HKM algorithm is an unsupervised learning technique for partitioning entities (annotated terms extracted from images) into K different contexts by clustering them with the nearest mean. For our purpose, we applied it using the Confusion Factor (CF) matrix. The condition for clustering the CF matrix into a set of smaller CF matrices (smaller communities) will be defined based on the threshold (α) of the community-wide accuracy. The number K can be determined in a heuristic manner. We have applied a simple rule ($k = \sqrt{n}$, where n is the number of classes in the community for the sake of simplicity) level-by-level during the clustering.

$$\sum_{i=1}^k \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \mu_i\|^2 \quad (4.5)$$

where μ_i is the mean of points in S_i .

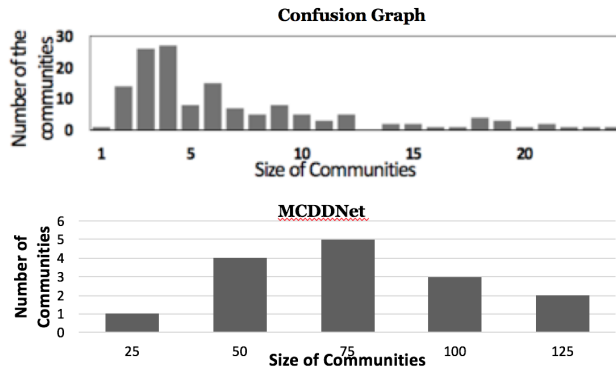


Figure 30: ImageNet Community Size: Confusion Graph vs. MCDDNet

The HKM algorithm is an excellent way to discover the optimal distribution of categories by minimizing the confusion factor between classes and hierarchically organizing them.

4.2.3.2 Distribution Algorithm and Strategies

The Distribution algorithm of the MCDD is intended to group the categories into communities such that the community achieves better accuracy compared to belonging to the global community.

4.3 Experimental Results

In this section, we evaluate the performance of MCDD on three datasets: Caltech-101 [123], CIFAR-100 [134], ImageNet-1K [135]. First, we verify the effectiveness of MCDD in accuracy and efficiency by comparing it with other works. We investigate the effect of level-by-level clustering so that the community will be organized for distributed

Algorithm 4: Class Distribution using Confusion Factor

1 **Input:** Data set d with n number of categories, α represents the minimum accuracy threshold for community splits, t represents the minimum possible size

Output: k communities with dissimilar categories sets

2 **WHILE** $noChange$ is true

3 $FM_{ij} = ClassificationAlgorithm(d)$

4 $ED_{ij} = \sqrt{(FM_{ij} - avg_{ij})^2 + (FM_{ji} - avg_{ij})^2}$

5 $k' = 0$

6 $k = \sqrt{n}/2 - k'$

7 $CommunityModel = Clustering..$

8 $Algorithm(ED, k)$

9 **for** $i \in n$ **do**

10 | $CommunitySet[i] = Community..$

11 | $Model.predict(ED[i])$

12 **end**

13 **for** $i = 1$ **to** $CommunitySet.length$ **do**

14 | **if** $CommunitySet[i].accuracy > \alpha$ **then**

15 | | $noChange = false$

16 | **end**

17 | **if** $CommunitySet[i].size < t$ **then**

18 | | $k' = k' + 1$

19 | **end**

20 **end**

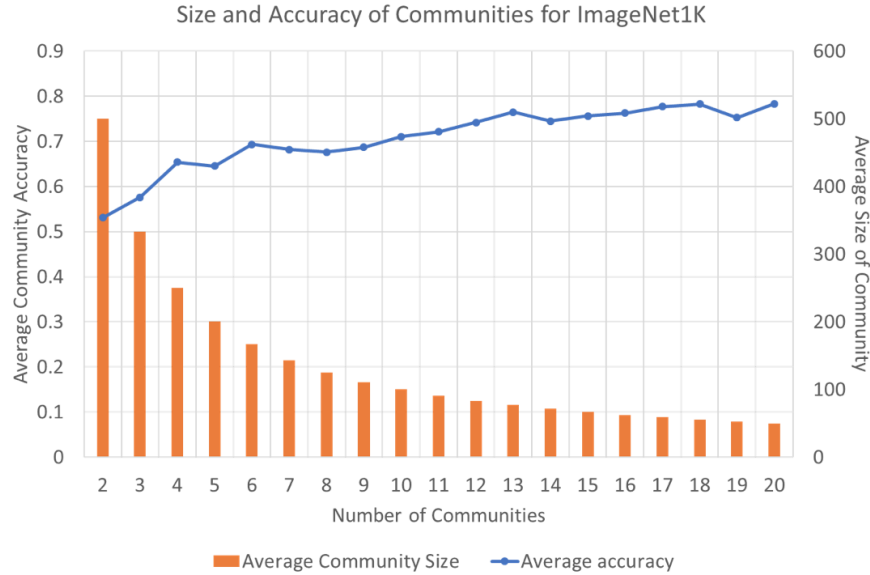


Figure 31: ImageNet Community and Classification Accuracy

classification hierarchically. We also examine the impact of distributed classification on large-scale multi-class classification and learning on several communities and hierarchical levels. Finally, we validate the proposed MCDD model in terms of the accuracy analysis for the optimal distribution. It has been conducted using the combination (the order does not matter and repetition is not allowed) where n is the number of classes to choose from, and we choose k from them. The number of ways to distribute n classes into k learners (communities), each community will have $\delta \leq \sqrt{n}$ classes.

We validate the proposed MCDD method for image classification tasks using three benchmark datasets. As shown in Table reftable:datasets, the CIFAR-100 dataset [134] contains 32 X 32 pixel images from 100 generic object classes. For each class, there are 500 images for training and 100 images for the test. We set aside 50 images for each class

Table 28: Community-Based Classification: Confusion Graph [72] vs. MCDDNet on AlexNet with ImageNet

Model	Confusion Graph		MCDD	
Type	Best	Worst	Best	Worst
Size	3	3	25	41
Comm.	352, 353, 354	651, 829, 856	Community 1	Community 4
Error%	28%	65.71%	4.9%	54.6%
Size	3	3	37	62
Comm.	231, 232, 233	589, 791, 792	Community 2	Community 5
Error%	42.66%	51%	8.1%	50.1%
Size	3	3	47	69
Comm.	409, 587, 848	571, 692, 797	Community 3	Community 6
Error%	44.82%	50.35%	9.4%	40.6%

from the training dataset as a validation set for cross-validation.

Secondly, the ImageNet-1K dataset [161] that consists of 1.2 million images from 1,000 generic object classes. For each class, there are 1K-1.3K images for training and 50 images for validation, which we use for the test, following the standard procedure.

- **Baselines.** To compare different ways to obtain grouping, we test multiple variants of our MCDDNet and baselines.
- **Base Network.** Base networks with full network weights. We use AlexNet [129], VGG [78] and GoogLeNet [127] variants as the base network for the ImageNet-1K [135].
- **Semantic.** We use a semantic taxonomy model (6 top-level Caltech-101 semantic categories) in [162] and 18 CIFAR-100 semantic categories in [72].

The communities detected by MCDDNet are much larger than ones from the confusion graph, as shown in Figure 30 and Figure 31. Thus, the number and the size of

communities are more significant than ones from the confusion graph. Also, the overall accuracy of the community-based classification in MCDDNet is significantly improved compared to the accuracy of the detecting communities in the confusion graph.

Figure 32 and Figure 33 confirm that the community classification performance is strongly related to the following two factors: the number of trouble makers and the accuracy distribution among the members in each community. In the best communities Figures 32a- 32c shows very few trouble makers and Figures 33a- 33c shows a distribution pattern (a high mean value (m) and a low standard deviation (sd)) in members' classification performance (similar font sizes). For example, Community 1: mean = 0.95 and sd = 0.0391. In the worst communities Figures 32d- 32f shows many trouble makers (red color) and Figures 33d- 33f shows a high distribution pattern (a low mean value (m) and a high standard deviation (sd) of the community classification accuracy. For example, Community 4: mean = 0.49 sd= 0.16.

4.3.1 Evaluation for Community-based Classification

Table 28 shows the accuracy improvement (error rate in %) in the experiments for two models (Confusion Graph [72] and our MCDDNet). We tested each refined model using images from the ImageNet validation set.

Table 29 shows the comparison of the MCDDNet and SplitNet in terms of three datasets, CIFAR-100, ImageNet, and Caltech-101 on AlexNet and VGG Network. MCDDNet shows a significant improvement in the classification errors by community detection and

indigo_bunting goldfinch
bullet_train peacock
yellow_lady's_slipper daisy
giant_panda geyser manhole_cover
echidna trolleybus
lorikeet oystercatcher jacamar
monarch pickelhaube
odometer web_site rock_beauty
European_fire_salamander
cardoon great_grey_owl
lesser_panda school_bus
rugby_ball

(a) Community 1

European_gallinule
cabbage_butterfly dining_table
pool_table broccoli affenpinscher
scoreboard killer_whale stupa
coucal triumphal_arch meerkat
Petri_dish go-kart four-poster
lifeboat steam_locomotive trilobite
keeshond anemone fish tench
tiger hoy aircraft_carrier jinrikisha
wombat gromotiger_beetle slot robin
jay water_tower axolotl
red-backed_sandpiper
hippopotamus scuba_diver
cliff_dwelling

(b) Community 2

chickadee motor_scooter
proboscis_monkey water_ouzel
sulphur-crested_cockatoo
hartebeest forklift brambling snow_leopard
ringlet drilling_platform Gila_monster
bee_eater rapeseed sorrel gondola
strawberry steel_arch bridge chow
drake papillon zebra chiffonier gyromitra
hyena agaric jack-o'-lantern flamingo
limpkin carbonara lionfish carousel lycopenid
admiral electric_locomotive Model_T
black_swan stone_wall coral_fungus
frilled_lizard chambered_nautilus
goldfish king_penguin whiskey_jug
sulphur_butterfly
ruddy_turnstone
African_hunting_dog

(c) Community 3

wooden_spoon
water_bottle harmonica
syringe park_bench cowboy_hat
perfume cleaver ant violin
hook ladle prison wool nail
drum mask spider_monkey
hair_spray plunger Great_Dane lipstick
buckle knot slug bucket spotlight
dumbbell chain_saw drumstick
pencil_sharpener hand_blower toilet_tissue
combination_lock

(d) Community 4

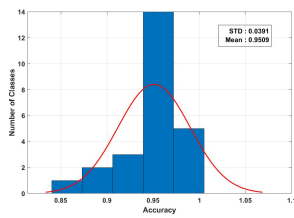
Labrador_retriever American_Staffordshire_terrier
collar_telephone weasel handkerchief
paper_towel milk_can hatchet letter_opener
screwdriver standard_poodle soccer_ball
bow purse mongoose chain_mail
sunglass mink cockroach Band_Aid
scale night_snake cup pick_maraca
shoe_shop vulture boxer rule_mortar flute
mouse missile langur radio reel chime
Weimaraner hog barrow
sarong acorn_squash gar candle
tray crayfish toy_terrier
loudspeaker crash_helmet space_heater
wine_bottle punching_bag
umbrella breastplate safety_pin

(e) Community 5

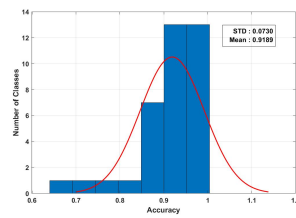
barbershop notebook ban_constrictor
vacuum Rhodesian_ridgeback pop_bottle
saltshaker steel_drum estethoscope scoreboard
monastery butternut_squash sandal corkscrew
Cardigan tiger_cat cocker_spaniel
knee_pad plate Rack sea_slug stole ballpoint laptop
Ibizan_hound burrito titi packet toy_poodle
whippet ram barn_spider crate visagheine
Egyptian_cat siamang coffeepot vase church
iron dishwasher maillot parallel_bars
common_newt ear sea_lion bathing_cap muzzle
silky_terrier Norfolk_terrier hognose_snake
baker's
cornet bassoon giant_schnauzer hamper
modem Irish_wolfhound thunder_snake plastic_bag
power_drill soap_dispenser Irish_terrier

(f) Community 6

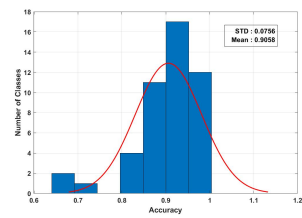
Figure 32: ImageNet Communities: 3 Best and 3 Worst Communities



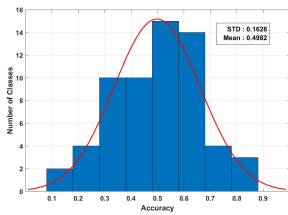
(a) Community 1



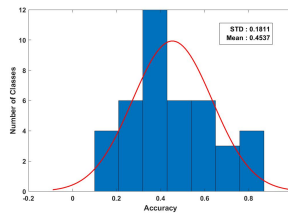
(b) Community 2



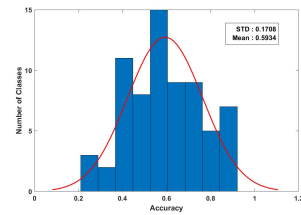
(c) Community 3



(d) Community 4



(e) Community 5



(f) Community 6

Figure 33: ImageNet Communities: Accuracy Distribution

Table 29: Comparison: SplitNet vs. MCDDNet

Method	Dataset	Network	Depth	Group	Baseline	Error(%)	Reduced(%)	Semantic
SplitNet	CIFAR-100	AlexNet	6	2	24.28%	23.96%	0.32%	24.46%
	ImageNet	AlexNet	3	1-1-3	41.72%	42.07%	-0.35%	N/A
		ResNet	3	1-1-3	25.58%	24.90%	0.68%	N/A
MCDDNet	CIFAR-100	AlexNet	3	10	55.3%	19%	36.3%	N/A
				18	55.3%	12%	43.3%	33.6%
	ImageNet	VGG	2	5	64%	34%	30%	N/A
		AlexNet	2	16	58.3%	27%	31.3%	N/A
	Caltech-101	AlexNet	3	8	21.1%	6%	15.1%	12.6%
		VGG	3	8	23%	5%	18%	N/A

Table 30: MCDD Hierarchical Classification Model Results on AlexNet

Dataset	Level	Error%	Comm#	Size of Community
CIFAR-100	1	55%	1	100
	2	25%	5	20
	3	19%	10	10
Caltech-101	1	21%	1	101
	2	9%	5	20
	3	6%	8	13
ImageNet-1K	1	58%	1	1000
	2	27%	16	63
	3	17%	43	23

construction steps. Table 30 shows the effective of the hierarchical community construction using the MCDD model. This table clearly shows the significant improvement in the accuracy of going deeper and smaller communities. Table 31 shows the effectiveness of the classification for varying sizes of images in the CIFAR-100 dataset for training. We have found that the classification performance with 200 images is still excellent compared to other cases.

The tag clouds of the best 3 communities and worst 3 communities are shown in Figure 32a- 32c and Figure 32d- 32f, respectively. The tag cloud is a visual representation of the community, specifically used to depict classes in the community. Tags are class names of the ImageNet data, and the accuracy contribution of each class is shown with font size or color. The green color and red color are used to represent the superstars and trouble makers. The font size is representing the accuracy of each class in their community. We can easily find out that the font sizes of classes in the best three communities are similar, while the worst three communities have quite diverse classes (some are big, and some are small).

4.4 Conclusion

In this chapter, we presented Multi-Class Discriminative Distribution (MCDD) for effective deep learning with large scale datasets. In the MCDD framework, we presented an optimal distribution of classes by computing a misclassification cost (i.e., confusion factor). The classification hierarchical deep neural network model was built by learning

Table 31: Performance with CIFAR-100 Image# on AlexNet

Image#	Level	Accuracy	Comm.#
100	1	41%	1
	2	72%	5
	3	75%	9
200	1	45%	1
	2	75%	5
	3	81%	10
300	1	46%	1
	2	75%	5
	3	78%	9
400	1	50%	1
	2	76%	5
	3	81%	9
500	1	49%	1
	2	74%	5
	3	81%	11

an optimal allocation of classes with a higher accuracy performance of the learning process. The MCDD framework was validated using large real-world datasets (Caltech-101, CIFAR-100, ImageNet-1K) with higher accuracy than existing models. The work presented in this chapter was published as part of Chandrashekar et al.[75]. Zhao et al. [163] and Vaka et al. [164] are a few application papers based on fundamental machine learning ideas.

CHAPTER 5

ZERO SHOT LEARNING FOR TEXT CLASSIFICATION USING CLASS REPRESENTATIVE LEARNING

5.1 Introduction

Zero-Shot Learning (ZSL) has been a very active research area in the field of image processing. However, ZSL in text classification has attracted very little attention despite the increasing interest in the research of NLP and text classification. ZSL is classically defined as the classification of unknown or unseen (target) categories by using external linguistic or semantic information through intermediate-level semantic representations from seen or known (source) categories [61, 91, 62].

With the rapid growth in topics in social media and having fewer and fewer labeled training data, Zero-Shot Learning-based Text Classification has become an essential research problem. The recent ZSL works demonstrated their effectiveness in transferring from prior experiences to new classes, a form of transfer learning. The most used semantic space in the ZSL model is supported by a joint embedding framework called Label-Embedding Space containing a combination of visual embeddings and word embeddings [4, 12, 165]. In this work, we attempt to take advantage of already well-established Label Embedding Space and use it for text classification.

With the prevalence of word embeddings, more and more work adopts pre-trained word embeddings to represent the meaning of words, to provide the models with the

knowledge of labels through Generative Models [166], External Knowledge Graph [167] or Class Descriptions [167, 168]. In contrast to prior works, we mainly extract the deep neural network features learned from text inputs of seen classes creating image representatives. We do not rely on any other features such as attribute annotations or external information. The goal is to propose an innovative model called *Class Representative Learning (CRL)* for zero-shot learning-based text classification for seen and unseen data. In this model, the focus is on creating a universal representation called the class representatives, which is typically based on Seen Classes based pre-trained deep learning models. Given this goal, architectural improvements are not our purpose; instead, we explore universal representatives that could be used for classification. It is desired to enable the universal representation to be trained from any existing architectures or datasets with reduced efforts and resources. The minimum requirement for the CRL model is to have a suitable *seen* (pre-trained) model that can be mapped to the given datasets (*unseen*).

The contributions of our work can be summarised as follows:

- We propose a new framework for Zero-Shot text classification, a framework for the design of abstract representatives as classifiers, and suggest innovation in ZSL research. Unlike previous ZSL works [61, 91, 62] in which they require extra information besides text data, our framework does not require any external or auxiliary information.
- We create prototypes for each class as classifiers, called Class Representatives, independent of other classes.
- We evaluate the Class Representative Learning (CRL) framework using metrics

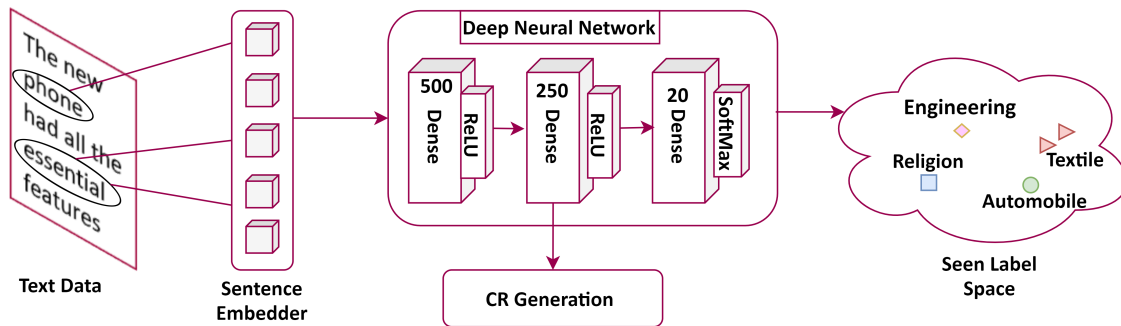


Figure 34: Class Representative - Text Classification

established by Xian et al. [92].

5.2 Related Work

In [169], a method for word embedding with LSTM network and aspect-based LSTM was proposed for Zero-Shot multiclass classification by learning the relationship between a sentence and embedding of sentence’s tags and applying it for inferencing with unseen sentences and tags into the same embedding space. In [166], a Discriminative and Generative LSTM model was proposed for ZSL with label-embedding space as an auxiliary task. In [168], a neural architecture was proposed for handling few-shot and zero-shot labels in the multi-label setting in the form of a DAG for labels with their natural language descriptor. In [167], a zero-shot text classification framework was designed using data augmentation and feature augmentation. For an efficient ZSL, semantic knowledge, including word embeddings, class descriptions, class hierarchy, and a general knowledge graph, were incorporated into the proposed framework. In [170], natural language descriptions were mapped to probabilistic assertions grounded in latent class

labels. A classifier was trained with quantitative constraints for guiding predictions from the learned models.

A ZSL method was proposed for semantic utterance classification (SUC) by linking categories and utterances through a semantic space [171]. The discriminative semantic features were learned without supervision and will guide the learning of the semantic features. A latent feature generation framework was proposed for generalized zero-shot learning (GZSL) that aims at improving the prediction on codes for the diagnoses of diseases [172]. For improved semantic consistency between the generated features and real features of the International Classification of Diseases (ICD), an adversarial generative model was designed for the GZSL on multi-label text classification.

A joint space of embedding documents and labels was designed for multi-label text and ZSL classification [173]. The zero-shot learning algorithm has been applied to the multi-label classification task in Medical Subject Headings (MeSH) assignment for biomedical publications.

5.3 Text-based Class Representative Learning

Class Representative Learning (CRL) Framework consists of two main steps (as shown in Figure 34), namely: the first step is Projection Function and Inference Function. In this chapter, three different variations of the the projection function $P(\cdot)$. The first variation is a combination of sentence encoder and two-layer feed-forward network. The second and third variation is sentence encoder based on universal sentence encoder(USE)

and sentence bidirectional encoder representations from transformers (S-BERT). Equation 5.1 shows the first variation of $P(\cdot)$, which has a sentence encoder and two-layer feed-forward network. Sentence Encoder (SE) transforms given documents to feature matrix. The Learner Network (LN) learns better intermediate feature vectors by taking the document feature matrix from SE as the input and mapping them to seen classes S . The third step is Class Representative Generation (CRG), which takes both the seen S and unseen U classes through the pre-trained SE and LN and extracts an intermediate feature vector to create Class Representatives CR .

$$\begin{aligned} \text{Projection Function } P(\cdot) : D_k^i &= LN(SE(T_i)) \\ \text{Inference Function } I(\cdot) : y^* &= I(\hat{x}^*, D_k^i) \end{aligned} \tag{5.1}$$

Table 32: Notations for Text-based Zero-Shot Learning

Notation	Description
(T^i, y^i)	Input Documents , $(T^1, y^1), (T^2, y^2), \dots, (T^n, y^n)$ where $i \in (1, n)$
$\mathbb{S} \ \& \ \mathbb{U}$	Seen Classes & Unseen Classes respectively
D_k^i	Document Feature Matrix for the Text Document T^i with k dimensions
$P(\cdot)$	$LN(SE(\cdot))$ Combination of Sentence Encoder and Learner Network are used as Projection Function
$SE(\cdot)$	Sentence Encoder takes a input of T^i and produces output of D_k^i
$LN(\cdot)$	Learner Network takes a input of D_k^i and produces output of \hat{D}_k^i
$I(\cdot)$	Inference Function

5.3.1 Projection Function

5.3.1.1 Sentence Encoder

Sentence Encoder (SE) is the first step of our framework, which focuses on converting sentences to encoded vectors. We consider a set of n text documents $T = T^1, T^2, \dots, T^n$ as an input set, and convert the text document T into Document Feature Matrix D_k . Equation 5.2 shows the transformation of each text T^i , where $i \in n$ to Document Feature Vector D_k^i through Sentence Encoder Function $SE(.)$

$$D_k^i = SE(T^i) \quad (5.2)$$

The Document Feature Matrix D_k consists of n Document Feature Vector D_k^i , where $i \in n$ with the corresponding class label y^i where $y^i \in S$. The dimension of the Document Feature Matrix k is dependent on which a sentence encoder is used for transformation.

In this chapter, we consider the existing framework, Universal Sentence Encoder (USE), as the sentence encoding. USE encodes text into high-dimensional vectors, where the model is trained and optimized for sentences, phrases, and short paragraphs. The USE's deep averaging network model is used as part of our CRL Model [174].

As shown in Figure 36, deep averaging network (DAN) averages the input embedding of the words and bi-grams together and then the average embedding is passed through a feed-forward network [175].

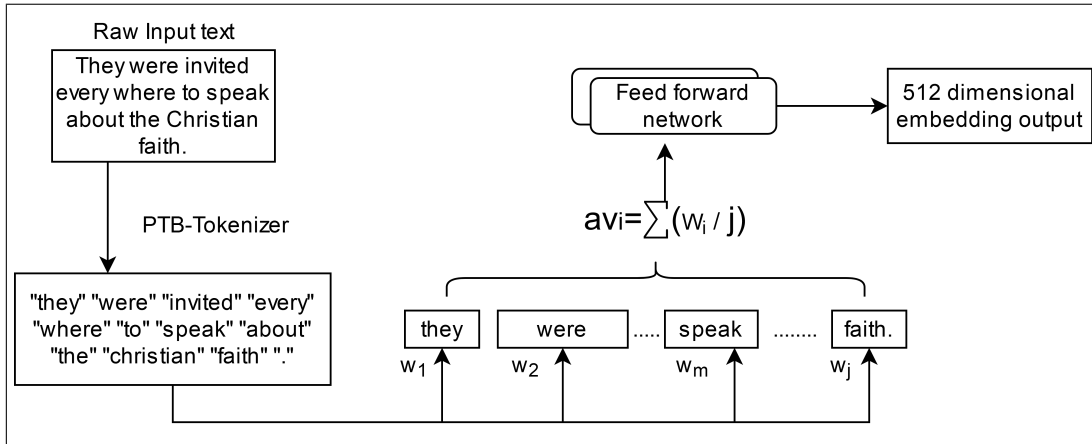


Figure 35: Overview of Universal Sentence Encoder using Deep Averaging Network

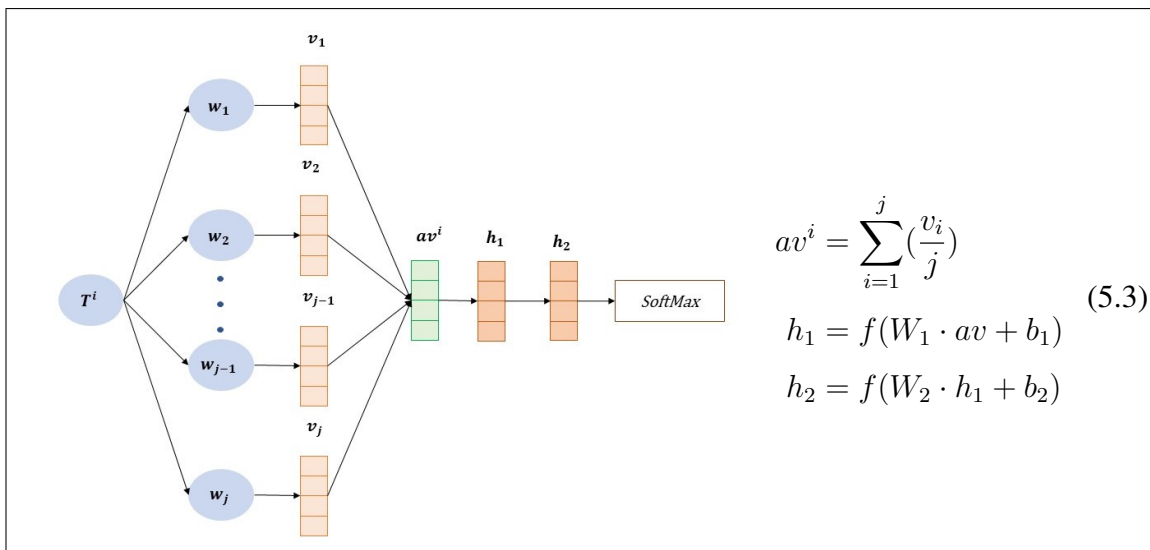


Figure 36: Universal Sentence Encoder based Deep Averaging Network

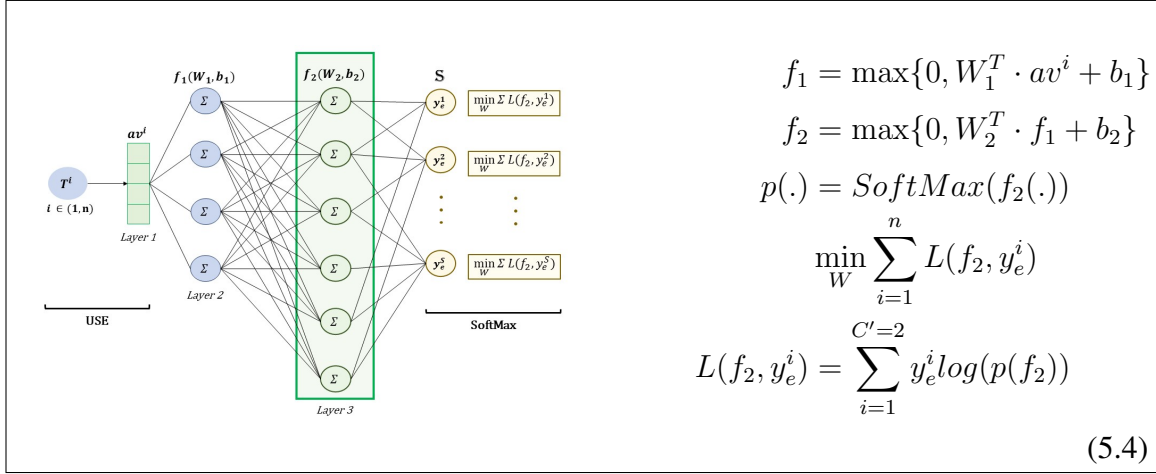


Figure 37: Two-layer feed-forward Network

5.3.1.2 Learner Network

Learner Network (LN) can be based on any neural network model that aims to build a ZSL model with the seen classes S for predicting class labels of the documents for unseen classes U . In this chapter, for the learner network, we introduce a Three-Layer Deep Neural Network(DNN), as shown in Figure 34. The DNN consists of two dense layers coupled with the rectified linear unit (ReLU) activation function and the final layer for the seen classes.

Two dense layers are based on the non-linear activation function. Equation 5.4 shows the non-linear mapping function $DL(\cdot)$, which incorporates both the dense layers. The ReLU activation function was implemented element-wise over each feature vector D_k^i in Document Feature Matrix D_k . The weights and biases of the dense layer 1 & 2 are represented as (W_1, b_1) & (W_2, b_2) , respectively.

The final layer is a multi-class probabilistic classifier that produces a S -dimensional

vector of probabilities p for each feature vector D_k^i from Document Feature Matrix D_k , where $i \in n$, as shown in Equation 5.4.

The layer of SoftMax is calculated for the seen classes S . During the training of the Learner Network, the model is building based on the seen classes' data. The probability calculated in Equation 5.4 is used just to facilitate the binary cross-entropy loss function. Unlike the general neural network model, in the CRL framework, the CRs play class-base discriminant roles in the final classification instead of the Softmax regression.

As shown in Equation 5.4, we aim at learning the non-linear mapping $DL(\cdot)$, i.e., obtaining network weights W_1 and W_2 using Binary Cross-Entropy Loss. Binary Cross-Entropy Loss ($L(\cdot)$) sets up a binary classification problem between $C' = 2$ classes for every class in Seen Class Set S . Equation 5.4 shows the minimization function across the weights W_1 and W_2 (shown as W) with the Loss function, which takes each Document Feature Vector D_k^i with the corresponding label y^i as the input. (Note: y_e^i is the one-hot encoded version of the label.)

5.3.2 CR Generation

Class Representatives (CRs) are generated using the nearest prototype strategy by aggregating feature vectors and is independent of the Learner Network. The closest mean feature vector with instances of the given class (i.e., CRs) is computed class by class. In order to generate CRs, an average mean operation with the feature maps was used to summarize the instances of classes. For each class, the instances of each feature in the feature maps are aggregated into an abstract mean feature. The CR is an aggregated

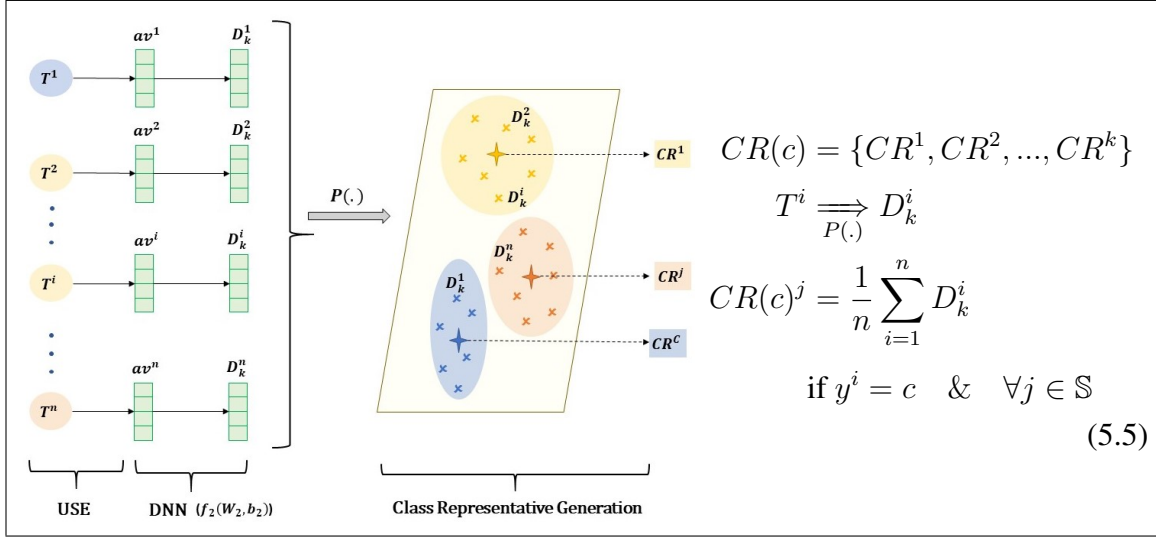


Figure 38: Class Representative Generation

vector of the mean features for all the elements in the feature maps. The feature maps are generated from a dense layer of the DNN network.

The Class Representative is generated for both Seen S and Unseen U classes using the pre-trained Sentence Encoder (SE) and pre-trained Learner Network (LN) from the previous steps.

Figure 38 shows the class representative generation. The equation 5.5 shows the aggregation formula where j ranges from 1 to l representing the feature dimensions, c is the class of the input text and $c \in S \cup U$, and N_c is the number of data points for the class c . Class Representative of the given class c is represented as the group of CR features values CR^j where j ranges from 1 to l feature dimensions. CR^j is generated from the mean of $DL(\cdot)$ with pre-trained weights of every input document D_k^i in a given class c as shown in Equation 5.5. Each class c in Seen and Unseen Classes ($S \cup U$) has a CR

generated at this stage.

5.3.3 CR-based Inference

The CR-based inference is matching the input data into the Class Representatives (CRs) and classifying with the best matched CR to the input. The CR-based inference can be done in parallel since the CRs are independent of each other.

Here are the steps for the CR-based inference. The input is vectorized using Equation 5.4: $\hat{a}(x^*) = DL(D_k^i)$. The cosine similarity between the new text document (NI) and Class Representatives for class c ($CR(c)$), where $c \in C$ is computed using Equation 5.7. The CRL Model assigns the new input with the label associated with Class c that has the highest cosine similarity score. The higher cosine similarity score indicates the closeness between the Class Representative $CR(c)$ and the new input (NI) in the Class Representative Feature Space (CRFS).

$$y^* = \operatorname{argmax}_{c \in \mathbb{U}} \{\cos(CR(c), \hat{a}(x^*))\} \quad (5.6)$$

where

$$\cos(CR(c), \hat{a}(x^*)) = \frac{CR(c) \cdot \hat{a}(x^*)}{\|CR(c)\| \|\hat{a}(x^*)\|} = \frac{\sum_{j=1}^k \{CR^j(c) * \hat{a}(x^{j*})\}}{\sqrt{\sum_{j=1}^k (CR^j(c))^2} \sqrt{\sum_{j=1}^k \hat{a}(x^{j*})^2}} \quad (5.7)$$

As shown in Equation 5.7, the label for the input from CRL Model \hat{c} is predicted by selecting the class from all seen (source) classes S or all unseen (target) classes T that has the highest cosine similarity to the new input. The CRL model will conduct inferring

by matching the new input against the available CRs and label it with a class having the highest cosine similarity score.

5.4 CRL Experiments and Evaluations

We have conducted a set of extensive experiments for text classification and Zero-Shot Learning (ZSL) with three datasets. For the text classification experiment, we have built and evaluated four different CRL models with various embeddings methods. For the ZSL evaluation, we have built three CRL models and compared the results with several state-of-the-art ZSL methods.

5.4.1 Datasets

IMDB Movie Dataset: It is a large dataset representing a binary (positive or negative) sentiment for each of the movie review. The IMDB[176] sentiment analysis dataset consists of 100,000 movie reviews taken from the IMDB large movie rating and review site. One key aspect of this dataset is that each movie review has several sentences.

The 100,000 movie reviews are divided into 25,000 reviews for labeled training and testing, and 50,000 unlabeled instances. There are two types of labels: Positive and Negative, and these labels are balanced in both the training and the test sets. The dataset can be downloaded at [177]. This data is already preprocessed with NLP techniques, including lemmatization and stemming, segmentation, stop-word removal.

20 Newsgroup Dataset: It is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups. These 20 different

newsgroups are organized according to their topic so that each group corresponds to a specific topic. Some of them are very closely related to each other, for example, `comp.sys.ibm.pc.hardware`, `comp.sys.mac.hardware`, while others are highly unrelated, for example, `misc.forsalem`, `soc.religion.christian`. The dataset is divided into 11,314 training samples and 7,352 testing samples.

DBPedia Dataset: The DBPedia dataset contains hierarchical classes representing the structured information from Wikipedia [178]. The DBPedia dataset is constructed by picking 14 non-overlapping classes[179] from DBPedia 2014. From each of these 14 classes, the fields we used for this dataset contain the title and abstract of each Wikipedia article.

5.4.2 CRL Experiments for Classification

We have conducted experiments with four widely used embeddings in Table 34, to know which embedding techniques are useful in building competent Class Representatives (CRs). In this experiment, we have used both sentence-level embedding (e.g., USE) and word-level embedding (e.g., GloVe [55], Word2Vec [180], NNLM) techniques to see how the model might have to perform in the novel method for class representations. As seen from Table 34, USE outperforms the word embedding techniques (GloVe or Word2Vec) for the Class Representation Learning.

For this experiment (as shown in Table 33), the IMDB dataset was used as the base model. The two benchmark datasets, 20Newsgroup, and IMDB, are used for training and testing. Class Representatives (CRs) were generated for positive and negative categories

for the IMDB movie reviews.

In this experiment, the four kinds of the CRL model have been built and evaluated.

Word2Vec+CRL: Word2Vec [180] compute high dimensional word vectors from a very large corpus. High-quality word vectors were trained using model architectures with the CBOW and Skip-gram models. This model has been efficiently encoded in embedding space to improve the semantic and syntactic generalizations by increasing the volume of training data.

NNLM+CRL: A model architecture, called neural network language model (NNLM) [181], was designed to learn both the word vector representation and a statistical language model. The NNLM architecture is composed of a feed-forward neural network with a linear projection layer and a non-linear hidden layer.

USE+CRL: Universal Sentence Encoder (USE)+CRL Universal Sentence Encoder [174] provided sentence level embeddings for transfer learning in a deep averaging network for NLP tasks. USE showed a better performance in transfer learning with NLP tasks compared to word-level embeddings alone. In the text classification, the CRL with USE embeddings showed the best performance among the four different CRL models.

USE+DNN+CRL For this experiment, we have also built a Deep neural network (DNN) that is composed of two hidden units with dimensions as [500, 100] along with the input and output layer. In this architecture (USE+DNN+CRL), (1) words classes were embedded using USE, (2) they were learned in the DNN network for the conditional probability of levels and words, (3) CR feature vectors were generated with the features extracted from DNN, and (4) CRL was used to estimate the class-based model. For the DNN

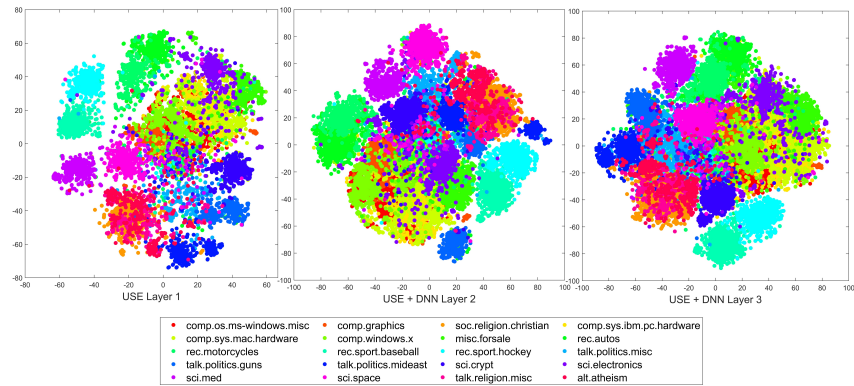


Figure 39: T-SNE Visualization for 20 Newsgroup Instances (USE+DNN+CRL)

training, 1000 epochs, i.e., 128,000 iterations with a batch size of 5 with 25000 training samples.

5.4.3 CRL Experiment for Zero-Shot Learning

The second set of experiments focuses on applying the class representation technique for Zero-shot learning. For these experiments, we divide our dataset into seen and unseen classes. Two different rates of unseen classes, 50%, and 25%, were chosen, and the corresponding sizes of a set of the seen classes and unseen classes are shown in Table 35.

5.4.3.1 Comparison based on Learner Network

For evaluating CRL in Zero-Shot Learning Setting, we use Convolution Neural Network and Deep Neural Network as Learner Network to validate.

USE+DNN+CRL: This model is what we proposed for Zero-Shot learning in this chapter. The framework of USE+DNN+CRL, as described in Section 5.3.1.2. Figure 39 shows the

t-SNE visualization of the 20Newsgroup dataset for the first, second, third layers of the CRL-ZSL architecture (USE+DNN+CRL) architecture, respectively. TSNE Visualization shows the clarity obtained through instance grouping, which corresponds to the accuracy shown for each layer in Figure 40.

GloVe+CNN+CRL: GloVe [55] is an unsupervised learning approach for word representations in a large corpus, rather than representing the entire sparse matrix or individual context windows. GloVe leverages a word-word co-occurrence matrix using a global log-bilinear regression model and outperforms other models on word similarity and named entity detection. The CRL has conducted with CRs generated from the model architecture of GloVe embedding and Convolution Neural Network (CNN). The first convolution is built on GloVe. We have used Glove embedding here since the sentence level embedding doesn't support convolution. As shown in Figure 40, USE+DNN+CRL model clears out-performs GloVe+CNN+CRL Model on both the datasets. Interestingly, in classification accuracy, we noted that USE+CRL performs better USE+DNN+CRL (See Table 34). Whereas in Zero-Shot Learning USE+CRL i.e., Layer 1 in USE+DNN+CRL Model has at least 5-10% less accuracy compared than other layers on both the datasets.

5.4.3.2 Comparison with SOTA ZSL Model

The USE+DNN+CRL Model is compared to the four state-of-art models, as shown in Table 36.

Label Similarity: The label similarity model was previously introduced for predicting unseen documents with labels using a semantic similarity between the label and the corpus



Figure 40: CNN/DNN Layer-based CRL Performance for ZSL with 20NG/DBP Datasets

[182]. For the class prediction, the cosine similarity measure was used to compute the similarity between class-based word embeddings and N-gram based word embeddings. The multi-label ZSL model [182] was revised to a single-label ZSL model and the revised model was used in comparative evaluation in [167].

LSTM+FC LSTM+FC predicts unseen sentences by learning the relationship between the sentences and embedding of their tags using the LSTM networks [169]. The LSTM network is designed 512 hidden units together with two dense layers, having 400 and 100 units, respectively. This model is generalized to predict if a given sentence is related to a tag or not, rather than classifying the sentence with a label.

CNN+FC: CNN+FC is revised in [167] by replacing the LSTM in the LSTMFC model with a CNN for building the zero-shot classifier.

CNN+ZSL: CNN+ZSL is a CNN-based two-phase framework using data augmentation and feature augmentation [167]. For ZSL, semantic knowledge, including word embeddings, class hierarchy, class descriptions, and a knowledge graph, are incorporated into the proposed framework.

As shown in Table 37, for comparison between SOTA Models, we report Seen Accuracy i.e., ($S \rightarrow S$), Unseen Accuracy i.e., ($U \rightarrow U$) and Overall Accuracy. The Unseen Accuracy of USE+DNN+CRL clearly outperforms with a minimum of 40% increase comparing to any other SOTA Model. The Seen Accuracy still lacks especially comparing to CNN+FC and CNN+ZSL models, but with only a maximum of 7% accuracy drop.

Algorithm 5: CRC-Inference: Zero-Shot Learning Setting

Input: $D_t = \{x_i, y_i\}_{i=1}^{m_t}; x^*$
Output: y^*
ZSL Setting: $y^* \in \mathbb{U}$

1 Projection Function $P(x)$:
 /* Projection Function as shown in Figure 36 & Figure 37 */
 $\hat{a}(x) = P(x)$
 return $\hat{a}(x)$

4 Inference Function $I(CR(c) \forall c \in \mathbb{U}, \hat{a}(x^*))$:

$$y^* = \operatorname{argmax}_{c \in \mathbb{U}} \{\cos(CR(c), \hat{a}(x^*))\} \quad (5.8)$$

 return y^*

7 CR-Classifier Function $CRC(D_t, x^*)$:
 $D_t \xrightarrow{P} \hat{D}_t$
 for $i \in 1 \rightarrow m_t$ **do**
 $\hat{a}(x_i) = P(x_i)$
 end
 $\hat{D}_t = \{(\hat{a}(x_i), y_i)\}_{i=1}^{m_t}$
 $\hat{a}(x^*) = P(x^*)$
 /* CR Generation: Based on Equation 5.5 */
 for $c \in \mathbb{U}$ **do**

$$CR(c)^j = \frac{1}{m_t} \sum_{i=1}^{m_t} \hat{a}(x_i^j),$$

 if $y_i = c \quad \& \quad \forall j \in (1, n)$
 end
 /* Classifier Function */
 $y^* = I(CR(c) \forall c \in \mathbb{U}, \hat{a}(x^*))$

Table 33: Class Representative Text Classification Setting

Dataset	#Class	Base	Embedding
20NG	20	IMDB	USE, NNLM,
IMDB	2		Word2Vec

Table 34: Class Representative Learning Text Classification- Testing Accuracy

Model	IMDB	20NG
Word2Vec + CRL	64.9%	54%
NNLM + CRL	68.6%	58%
USE + CRL	79.3%	65%
USE + DNN + CRL	76%	64.2%

Table 35: Dataset for Zero-Shot Learning Evaluation

Dataset	Unseen Split	#Seen	#Unseen
20NG	25%	15	5
	50%	10	10
DBPedia	25%	11	3
	50%	7	7

Table 36: State-of-Art Zero-Shot Learning Model

Approach	Model	Pre-trained	Semantic Knowledge
Label Similarity	Unsupervised		
LSTM+FC	LSTM	Google News	Semantic Emeddings for Tags, Word Embeddings (Word2Vec)
CNN+FC	CNN		
CNN+ZSL	CNN		Word Embeddings, Class Descriptions, Class Hierarchy, ConceptNet (Knowledge Graph)
CRL+ZSL (ours)	DNN	None	Embeddings (USE)

Table 37: Comparison of Zero-Shot Learning Models

Dataset	Split(%)	Category	Label Similarity	RNN+FC	CNN+FC	CNN+ZSL	CRL+ZSL (Ours)	CRL+GZSL
20NG	75:25	Seen	0.279	0.614	0.792	0.745	0.713	0.603
		Unseen	0.287	0.065	0.134	0.622	0.346	
		Overall	0.280	0.482	0.633	0.649	0.439	
	50:50	Seen	0.293	0.709	0.684	0.767	0.646	0.671
		Unseen	0.266	0.052	0.126	0.629	0.455	
		Overall	0.280	0.381	0.405	0.616	0.544	
DBPedia	75:25	Seen	0.377	0.895	0.985	0.975	0.954	0.859
		Unseen	0.426	0.046	0.204	0.852	0.724	
		Overall	0.386	0.713	0.818	0.900	0.785	
	50:50	Seen	0.401	0.960	0.991	0.982	0.944	0.907
		Unseen	0.369	0.044	0.069	0.878	0.765	
		Overall	0.386	0.052	0.530	0.929	0.829	

Table 38: Class-wise Accuracy for Class Representative Learning in Zero-shot Learning setting

20NG	50% Seen (10 classes)			50% Unseen (10 classes)		
	Class	F1-Score	Avg	Class	F1-Score	Avg
	alt.atheism	0.53	0.646	rec.sport.hockey	0.81	0.629
	comp.graphics	0.63		sci.crypt	0.59	
	comp.os.ms-windows.misc	0.54		sci.electronics	0.61	
	comp.sys.ibm.pc.hardware	0.57		sci.med	0.82	
	comp.sys.mac.hardware	0.55		sci.space	0.82	
	comp.windows.x	0.57		soc.religion.christian	0.67	
	misc.forsale	0.70		talk.politics.guns	0.63	
	rec.autos	0.75		talk.politics.mideast	0.55	
	rec.motorcycles	0.72		talk.politics.misc	0.45	
	rec.sport.baseball	0.90		talk.religion.misc	0.34	
DBPedia	50% Seen (7 classes)			50% Unseen (7 classes)		
	Class	F1-Score	Avg	Class	F1-Score	Avg
	Company	0.93	0.944	Natural Place	0.90	0.878
	Educational Institution	0.91		Village	0.89	
	Artist	0.97		Animal	0.85	
	Athlete	0.98		Plant	0.86	
	Office Holder	0.90		Album	0.79	
	Mean of Transportation	0.95		Film	0.91	
	Building	0.96		Written Work	0.95	

Class-wise Accuracy: The class-wise accuracy for USE+DNN+CRL is shown Table 38. For understanding accuracy for seen and unseen classes, we consider 50-50 split of 20Newsgroup and DBPedia Datasets. The Advantage of Class Representative can be effectively showcased that the average accuracy of unseen classes is almost as high as the average accuracy of seen classes.

5.5 Conclusions

In this chapter, we proposed a novel class representative framework learned from deep neural network features and sentence level word embeddings. The experiments

show that CRL+CL improved the accuracy using Universal Sentence Encoder in classification instances for text classification, while CRL+ZSL improved the overall accuracy for zero-shot learning in transferring knowledge from seen to unseen classes. From the experiments on three benchmark datasets across various domains, we achieved the highest overall accuracy compared with the state-of-the-art works in Zero-Shot Learning. As future work, we will extend our CRL framework to perform Zero-Shot Learning for multi-modal classification with a more substantial amount of mixed text and image data. The work presented in this chapter was published as part of Chandrashekar et al.[183].

CHAPTER 6

VISUAL CONTEXT LEARNING WITH BIG DATA ANALYTICS

6.1 Introduction

Understanding contextual information composed of both text and images is beneficial for multimedia information processing. We face challenges in dynamically capturing such contexts from real datasets. This challenge of capturing context is strongly related to big-data issues (i.e., volume, variety, velocity). Existing models such as ontologies and dictionary (e.g., WordNet) are mainly interested in individual terms/concepts. Still, they do not support identifying a group of words that describe a specific context. We assume that the association will define the relationship's consistent context among entities in the images. In our contextual model, the association among entities in visual settings (i.e., images) will be dynamically computed as relations in such contexts are extracted atomically from their annotated text. These are used to generate a contextual graph that describes a specific context of images.

We have relatively limited capacities to support for processing big data for extracting contextual information from heterogeneous sources and building contextual models for such big data. Furthermore, existing solutions are not scalable due to the computationally intensive tasks and prone to data sparsity.

Big data analytics have made significant progress in analyzing heterogeneous,

more massive datasets using big data processing frameworks like Hadoop and Spark. Processing, mapping, and integrating different data from multiple sources can be efficiently conducted for data-processing for real-world applications.

Our work is inspired by the concept of *end-to-end machine learning pipelines* in Spark MLlib that was proposed for scalable and efficient large-scale data processing in big data applications [184]. They have demonstrated the improvement of the speed, scalability, and continuous extension in Spark MLlib. Specifically, Spark provides a new computational paradigm for parallel and pipeline programming for big data analytics. Although Spark has shown the improvement of high performance in data analytics with big data compared to Hadoop [184], some challenges should be overcome for big data analytics. Specifically, there is still much room for improvement in the learning context from big data. Understanding the nature of what is required to build a context model is imperative. Providing context models for data processing is extremely time-consuming and complex but often highly demanding.

In this chapter, we have presented a novel framework based on a parallel and pipeline architecture that will support big data analytics for context learning from large scale images and their annotations. We present the VisContext framework (shown in Figure 41) that aims to support contextual learning from multiple sources by constructing clusters of contexts in multimedia. In the clusters, contexts are analyzed for preserving neighboring information that is relevant in a given context. The context models based on the associated entities and their neighbors are defined as context graphs in clusters of associated features. Effectiveness in the discovery of the contextual association of terms

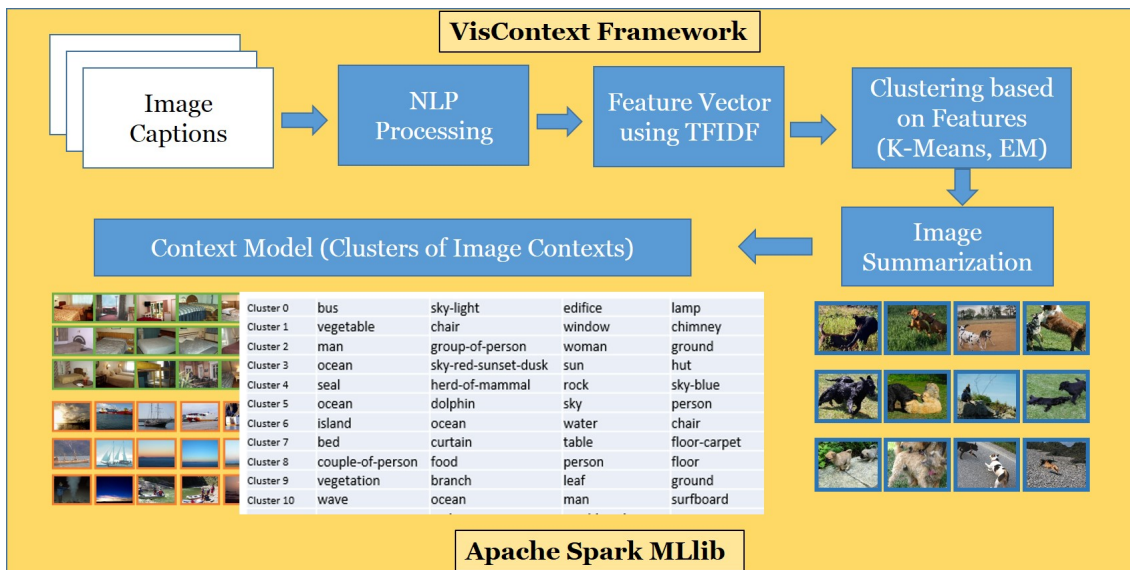


Figure 41: VisContext Framework on Apache Spark

and images, visual context symmetrization, and image classification based on context, have been evaluated using various well-known datasets.

We first rationalize the *Visual Context Model* (VisContext) that specifies contextual association for visual context learning from heterogeneous sources. Second, we define the association measurement between features and their clustering. Third, we evaluate the robust context learning approaches using state-of-the-art clustering and classification techniques for recognizing the most suitable method for building the VisContext model.

6.2 Related Work

There are ongoing efforts in annotating the contents of images. Correctly, for automatic annotation of images, a corresponding mapping schema between words and

images [185] and the spatial mapping between scene type and objects [186, 187] were presented. These works focus on correct image classification in terms of correctly labeling scenes or objects with a fixed set of categories. However, our work is different as we are interested in extracting contextual information from a collection of images and building a contextual model based on evidence from the derived contexts.

The following are some works that explore the description generation for images: Machine learning algorithms were applied to generate the most compatible annotation models [188], breaking annotations into smaller pieces and binding them together into new annotations [189], image caption generation based on predefined templates for the context [190], generative grammars [191], and a fixed window context [192]. Recurrent Neural Networks [193] were used to generate image descriptions depending upon the probability distribution over the next word [194], but they suffered in performance. Our work is different from these works since we focus on building context models and collecting contexts extracted from images for these context models.

Image annotation and description datasets [195] and sentence description for semantically similar images [196] establish image description generation through semantic scene understanding. Semantic tagging focuses on recognizing words of semantic importance and on a contextual understanding of a natural language description or query. Common approaches in semantic taggers include recurrent neural networks [193], sentiment analysis [197], named entity recognition [198], entity-relation extraction [199], and supervised word learning with word2vec [200].

These works depend on large amounts of manually annotated data or well defined

categorized data to achieve excellent performance. Besides, they are limited in terms of the context learning with associated conditions and presenting the semantic relations of these contextual terms. In [201], the visual context the structure was constructed by analyzing visual link graphs and latent semantics using Singular Value Decomposition (SVD), which is not applicable for a large number of images due to computational complexity.

Unlike these works, we focus on a pipeline approach that started from unstructured data to contextual knowledge using natural language processing, information retrieval technologies, and unsupervised learning approaches. Finally, the context models will be evaluated using supervised learning algorithms. This approach supports understanding the semantics of context models built from evidence extracted from images by analyzing the association of features (their concurrence and frequency in specific contexts) and their attributes.

6.3 Visual Context Model

Zitnick et al., [202] described the semantic meaning of images that can be captured through the presence of objects, their attributes, and their relations to other objects. It is not easy to extract such complex visual information from a group of images. In this chapter, we propose a new approach to building a model for a visual context model that depicts the visual contexts and the semantic information of images.

Definition 1: Visual Context - The *visual context* describes bounded contexts through the association of entities and their relations in a visual context. Different settings may

have completely different associations among any everyday objects and their relationships in multimedia domains. It is extended by the collaboration of entities with each other through contextual association model.

Definition 2: Visual Context Graph - The *visual context graph* describes bounded contexts through contextual association model. A visual context is represented by the context graph representing an association of entities (features) and their relations. Different entities may have different associations dependent upon a specific context.

Definition 3: Context Boundary - The *context boundary* defines the scope of context in which the information can be associated, related, and connected in a visual context graph. The association of data is given by boundary on the context graph based on terms described as sets of entities and relations

Various factors can determine boundaries between contexts. Usually, the dominant one is strongly associated with others so that this can be measured by high in-degree/out-degree and distance in a visual context graph. This boundary can be set differently depending on the context. Multiple contexts can be found within the same domain. Similarly, a single context can be found across various domains.

The visual context clusters are discovered with the bounded contexts that are a central concept in visual context learning. The clustering technique is applied to partition a large and complex context graph into multiple smaller settings. The bounded contexts are specifically tailored for a set of visual contexts. The boundary B is determined based on the distance L (without considering direction) between any two contextual features from different images. In this framework, the context boundary will be determined using

clustering techniques. The distance L can be measured differently depending upon the clustering techniques used. For example, the Euclidean distance was used for K-Means clustering and maximum likelihood estimation for Expectation Maximization clustering.

Definition 4: Degree of Association - The *degree of association* is defined to measure the degree of the association between features from different sets of images. The associate degree is determined with a weight assigned to links between contextual features from different categories.

The weight will be computed to measure the degree of the association between contextual features from different categories. In this chapter, TF-IDF was used to calculate the degree of associated visual contexts using Equation 6.1 for TF, Equation 6.2 for IDF and Equation 6.3 for TF-IDF . The rationale is to capture association relations between contextual features from multiple images by giving a higher weight to the ties for frequent evidence and contextual information distribution.

6.4 VisContext Framework: Image Context

The visual context model is a graph with the vertex set that corresponds to the features and the association among the features. In our context learning model, the association among features in images will be identified to describe a specific context. We assume that the association among features in the images will be defined in a consistent context for the relationship to hold. The proposed VisContext framework for building the visual context model is composed of (i) Natural Language Processing, (ii) Feature Extraction using TF-IDF, (iii) Visual Context Learning using clustering techniques, and (iv)

Validation using classification techniques.

6.4.1 Pruning of Annotation Terms using NLP

The first part of the VisContext framework is to prune annotation terms using Natural Language Processing composed of two primary operations: (i) Lemmatization. It is the algorithmic process of determining the base form by grouping together the various forms for a given word. It is required to understand the context of associated words and identify the part of speech of a word in a sentence. For example, 'run', 'ran', 'runs', 'running' will be mapped to the base form, 'run'. (ii) Stop Word Detection - It is the algorithmic process of filtering out some of the most commonly chosen stop words to improve the performance of pruning the annotation terms. For example, some of the most common and function words include 'the', 'is', 'with', 'that', and on.

6.4.1.1 Feature Extraction using TF-IDF

Each image consists of several annotated terms that can be used as features. This is used to identify the critical features implemented by each of the images. The input for this technique could be the annotated terms detected from the images. Visual features of images are identified by applying Term Frequency-Inverse Document Frequency (TF-IDF) [203] to their annotated terms. The high-level flow of feature extraction from the images is shown in Figure 41.

The feature extraction has the following four steps. In the first step, the features from the images are aggregated to form the feature matrix. The representative terms for

feature extraction are selected using TF-IDF, which is the product of term frequency and inverse document frequency. The term frequency is the number of annotated terms that appear in a specific image. Document frequency is the frequency of the terms in all the images. The Inverse Document Frequency intends to reduce the word's importance that occurs most frequently in all the images. It is mainly used to eliminate the common terms across all the images. The IDF value is computed by dividing the number of images with the number of images containing the given term t and then applying a logarithm to the resultant value. If the term appears in more images, it is more likely to be a common term that is not specific to any given image. Hence, the log value of the word reduces to zero, ensuring that the IDF value and thereby the TF-IDF value, is less for this term.

$$TF(t, d) = 1 + \log(f_{t,d}) \quad (6.1)$$

$$IDF(t, D) = \log \frac{N}{1 + |\{d \in D : t \in d\}|} \quad (6.2)$$

where N is the total number of images in the corpus, i.e., $N = |D|$ and $|\{d \in D : t \in d\}|$ is the number of images where the term t appears (i.e., $TF(t, d) \neq 0$).

TF-IDF value is high if the term has high term frequency and a low document frequency in the whole collection of images. Hence by considering the TF-IDF value, we can eliminate the common terms in determining features.

$$TF - IDF(t, D) = TF(t, d) \cdot IDF(t, D) \quad (6.3)$$

A feature may be implemented in different images. The feature variants implemented by different images could be different. Hence, we need to aggregate the (feature, <List of feature variants>) tuples generated from all the images to obtain the final feature to feature variant mapping.

6.4.2 Feature Association using Clustering Techniques

A context association model can be created for any given list of contextual features concerning any collection of any image corpus. Contextual sets of features can be called 'neighbors', and these often group into 'neighborhoods' based on their similarity or co-occurrence (interconnections) in images. Individual features may have several neighbors. Neighborhoods may relate to one another through at least one unique feature or may remain unrelated.

We designed the context association model using two different clustering algorithms: K-Means and EM.

6.4.2.1 K-Means Clustering for Associating Contexts

For constructing the visual context model, the *K-Means clustering* (KM) algorithm was used to group visible entities based on their contextual closeness properties to discover contextual information in images. In this chapter, we designed the KM algorithm for the discovery of relevant contexts from integrated multiple sources to form a context graph. The KM algorithm is an unsupervised learning technique for partitioning entities (annotated terms extracted from images) into K different contexts by clustering them with

the nearest mean.

Given a set of visual annotated terms (T_1, T_2, \dots, T_n) , where each image can be represented as a vector, KM clustering aims to partition the n terms into K , which is $\leq n$ sets $C = C_1, C_2, \dots, C_k$ to minimize the within-cluster sum of squares (amount of distance functions of each point in the cluster to the K center).

$$[\operatorname{argmin}_S \sum_{i=1}^k \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \mu_i\|^2] \quad (6.4)$$

where μ_i is the mean of points in S_i .

The KMeans algorithm is an excellent way to discover contexts with multiple features from images and summarize with an integrated view of the provided features.

6.4.2.2 EM Clustering for Associating Contexts

We also used an Expectation-Maximization (EM) algorithm [204] for building a visual context model. EM is an iterative method that aims to create a model for maximum likelihood of the features by exploring unobserved latent variables. The EM algorithm iteratively performs two steps: (1) Expectation step (E) creates the expectation function of the log-likelihood evaluated by using the current estimate of the parameters (2) Maximization step (M) computes parameters maximizing the expected log-likelihood computed from the E step (i.e., by determining the distribution of the latent variables in the next E step).

The EM algorithm seeks to find the maximum likelihood estimate (MLE) of the marginal likelihood by iteratively applying the steps. Given the statistical model which

generates a set \mathbf{X} of observed data, a set of unobserved latent data \mathbf{Z} , and a vector of unknown parameters θ , along with a likelihood function $L(\theta; \mathbf{X}, \mathbf{Z}) = p(\mathbf{X}, \mathbf{Z}|\theta)$, the MLE of the unknown parameters is determined by the marginal likelihood of the observed data

$$L(\theta; \mathbf{X}) = p(\mathbf{X}|\theta) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\theta)$$

Expectation step (E step): Calculates the expected value of the log-likelihood function, concerning the conditional distribution of \mathbf{Z} given \mathbf{X} under the current estimate of the parameters $\theta^{(t)}$:

$$Q(\theta|\theta^{(t)}) = E_{\mathbf{Z}|\mathbf{X},\theta^{(t)}} [\log L(\theta; \mathbf{X}, \mathbf{Z})]$$

Maximization step (M step): Finds the parameter that maximizes this quantity:

$$\theta^{(t+1)} = \operatorname{argmax}_{\theta} Q(\theta|\theta^{(t)})$$

6.4.3 Validation using Supervised Learning Algorithms

To validate the visual context models constructed from the clustering, we have utilized supervised learning algorithms (i.e., Naive Bayes, Decision Tree, and Random Forest) using Spark MLlib.

6.4.3.1 Naive Bayes classifier

It is a probabilistic classification algorithm based on applying Bayes' theorem with naive independence assumptions between the features. Given a problem instance to

be classified, represented by a vector $\mathbf{x} = (x_1, \dots, x_n)$ that represents some n features, it assigns to this instance probabilities $p(C_k|x_1, \dots, x_n)$, for each of K possible classes C_k . The probability can be formulated as follows:

$$p(C_k|\mathbf{x}) = \frac{p(C_k) p(\mathbf{x}|C_k)}{p(\mathbf{x})}$$

6.4.3.2 Decision Tree

A predictive model that maps observations about an item to build a model is called the decision tree. Tree models, where the target variable can take a finite set of values, are called classification trees. The decision tree is composed of nodes representing the features and edges representing the feature valuesâtraversing the tree to reach a leaf node representing a class label by matching branches representing the conjunctions of features. In a decision tree, information gain is the measure used for selecting features in the construction of a tree by computing information entropy H for the given information T . The information gain for an attribute a is defined in entropy $H()$ as follows:

$$IG(T, a) = H(T) - H(T|a) = H(T) - \sum_{v \in vals(a)} \frac{|\{\mathbf{x} \in T|x_a = v\}|}{|T|} \cdot H(\{\mathbf{x} \in T|x_a = v\}) \quad (6.5)$$

where T denotes a set of data, each of the forms $(\mathbf{x}, y) = (x_1, x_2, x_3, \dots, x_k, y)$ where $x_a \in vals(a)$ is the value of the a_{th} feature of \mathbf{x} and y is the corresponding class label.

6.4.3.3 Random Forest

Random Forest is an ensemble learning method that constructs a multitude of decision trees and classifies the input data by taking the majority voting within the individual trees. The training learner of random forests applies bootstrap aggregating. Given a training set $X = x_1, \dots, x_n$ with responses $Y = y_1, \dots, y_n$, bagging by repeatedly selecting a random data with the replacement of the training set and fits trees to these data. After training, predictions for unseen samples x' can be made by averaging the individual regression trees' predictions on x' .

For this validation, with the above three algorithms (Naive Bayes Classifier, Decision Tree, Random Forest), the datasets are randomly divided into 60% training and 40% testing datasets. The validation is designed as part of the VisContext framework. The accuracy performance of these three different algorithms (Naive Bayes Classifier, Decision Tree, Random Forest) are reported in Section 6.5.3.5.

6.5 Results & Evaluation

6.5.1 Datasets

We used three different datasets for validating our approach. All the datasets are of image annotations or images with captions. Table 39 shows the datasets used for evaluating our framework: SAIAPR TC-12 Benchmark(IAPR) [205], 1 million captioned images from Stony Brook University (SBU) [206], and Flickr30k dataset collected from Flickr (social media data) (Flickr30k) [207]

The SAIAPR TC-12 Benchmark(IAPR) describes the segmented and annotated

Table 39: Dataset Description

	#images	#terms	#terms after NLP	#unique terms
IAPR	20000	99527	99527	261
Flickr30k	158915	2127783	2124534	13273
SBU	1000000	13319299	13128289	177921

IAPR-TC12 benchmark designed for multimedia information retrieval. The IAPR TC-12 collection includes (i) Segmentation masks and segmented images for the 20,000 pictures, (ii) Features extracted from the regions and labels assigned to them, (iii) Region-level annotations according to an annotation hierarchy, (iv) Spatial relationship information. The SBU dataset and Flickr30k consist of Flickr images shared by different users on social media. Flickr is a hosting service for more than 6 billion images in blogs and social media, and the number of images is continuously growing. SBU is a pruned dataset compared to Flickr30k, as discussed in [206].

6.5.2 Implementation

The implementation was based on a 12 GB Ubuntu 14.04 operating system and was implemented using parallel processing based on Apache Spark. The Apache Spark is scalable and can be extended to larger sized datasets. For the VisContext framework’s scalability, all of the algorithms and implementation discussed in this chapter are entirely implemented in Spark. The algorithms were performed with the Spark MLlib, Spark NLP, and CoreNLP libraries.

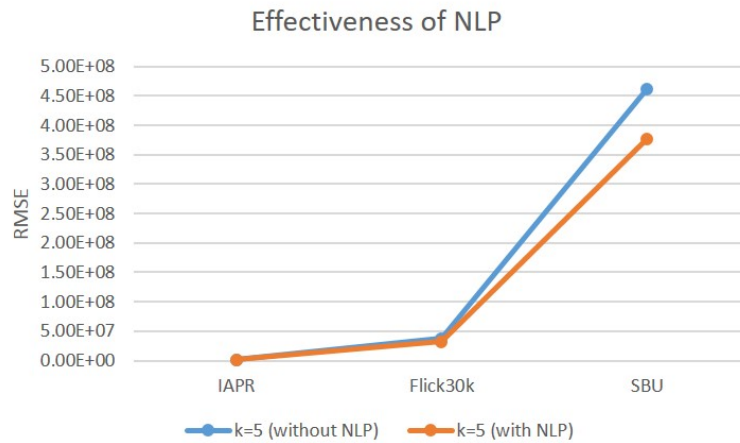


Figure 42: Evaluation: Effectiveness of NLP with RMSE

6.5.3 Evaluations

6.5.3.1 Effectiveness of Natural Language Processing

The Natural Language Processing (NLP), i.e., Lemmatization, Stop Word Detection, were conducted in the process of context discovery. We measured the effectiveness of NLP in context learning using Root Mean Square Error for two clustering groups ($K = 5$ with NLP and $K = 5$ without NLP, where K is the number of clusters). The effectiveness of the NLP based clustering (the error rates decreased after applying NLP to the data in the clustering case, $K = 5$ with NLP) is shown in Figure 42. The other advantage is the reduction in the number of terms, as shown in Table 39. We observed that the number of words after the NLP operations was lesser than the number of original words in all three datasets.

6.5.3.2 K-Means Clustering Validation: K Value vs. RMSE

We used Root-Mean-Square Error (RMSE) to validate cluster outcomes by minimizing the forecasting errors computed by the standard deviation of the differences between predicted and observed values. The RMSE of predicted values \hat{x}_t for times t of a regression's dependent variable x_t is computed for n different predictions as the square root of the mean of the squares of the deviations as follows:

$$RMSE = \sqrt{\frac{\sum_{t=1}^n (\hat{x}_t - x_t)^2}{n}}.$$

For the K-Means clustering algorithm, three different values of K ($K=5, 10, 20$) were considered for comparison, as shown in Figure 43. We observed that the best value was at $k=20$. Another observation was that the RMSE value was higher for social network data such as Flickr30k and SBU compared to the IAPR human-defined annotation data, with a limited vocabulary.

6.5.3.3 TF-IDF Validation: Vector Size and Accuracy

The comparison of TF-IDF vector sizes (50, 100, 150, 200) was performed to observe the influence of the scale over the clusters formed. The Naive Bayes classification algorithm was run for validating the clusters. Figures 44 - 46 shows the Precision, Recall, and F-Measure, respectively. The highest Precision was 85% for vector size=100 in Flickr30K, and the lowest one was 75% for vector size = 150 for IAPR. The highest recall was 82% for vector size = 50 in Flickr30K, and the lowest one was 68% for vector size =150 for IAPR. The highest F-Measure was 82% for vector size=100 in Flickr30K, and

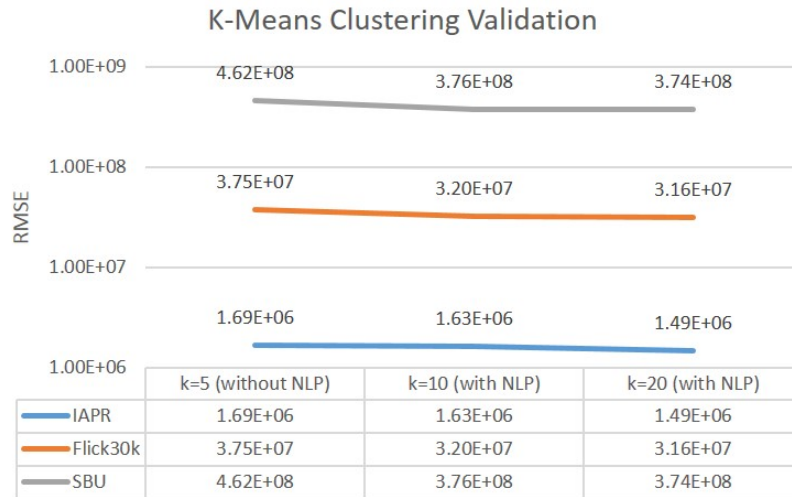


Figure 43: Evaluation: Root Mean Square Error(RMSE) vs. K Value

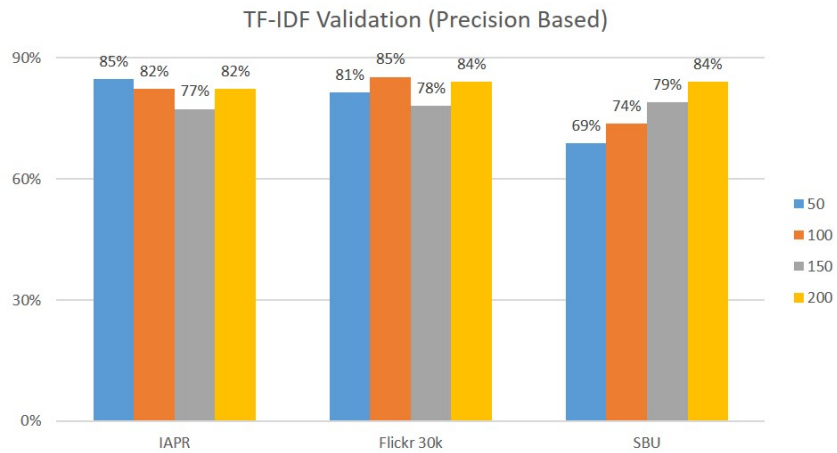


Figure 44: Evaluation: Feature Vector Size vs Precision

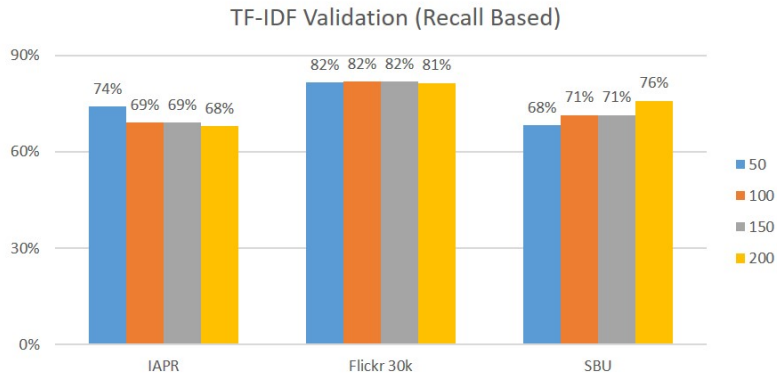


Figure 45: Evaluation: Feature Vector Size vs Recall

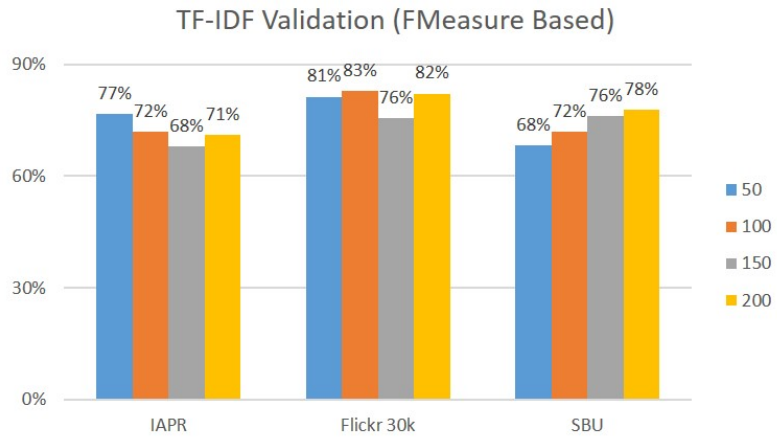


Figure 46: Evaluation: Feature Vector Size vs F-Measure

the lowest one was 68% for vector size = 150 for IAPR.

6.5.3.4 Latent Dirichlet Allocation on Datasets

Our context association model is different from Latent Dirichlet Allocation (LDA), a generative statistical model in which each document is a mixture of a small number of topics. Each word's creation is attributable to one of the document's topics. In our experiments, the LDA performed poorly on the three different datasets compared to our

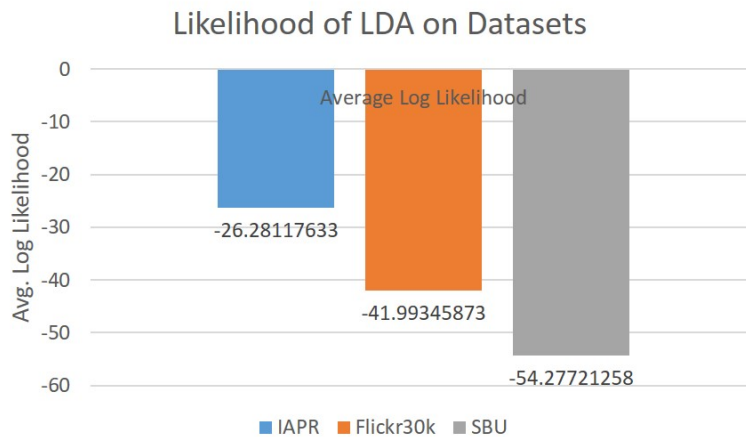


Figure 47: Evaluation: Average Log Likelihood of LDA Models

clustering-based approach against the LDA topic discovery model. Figure 47 shows the average log-likelihood for each dataset presented. One of the disadvantages of using these datasets for LDA is that there is no distribution available for the datasets in general, which makes the likelihood much lower.

6.5.3.5 Comparison of Clustering and Classification Algorithms

The two clustering algorithms, EM and K-Means, were compared with two different vector sizes (50, 100). Figure 48 shows the comparison using Precision. The observation from this comparison is that EM performed very well and formed unique clusters. The contexts discovered were verified using the three classification algorithms; Naive Bayes, Decision Tree, and Random Forest. Figures 49a- 49c shows the Precision, Recall, and F-Measure for different datasets and algorithms. Observation of these figures was that the overall performance was better with Naive Bayes. The highest F-Measures (by

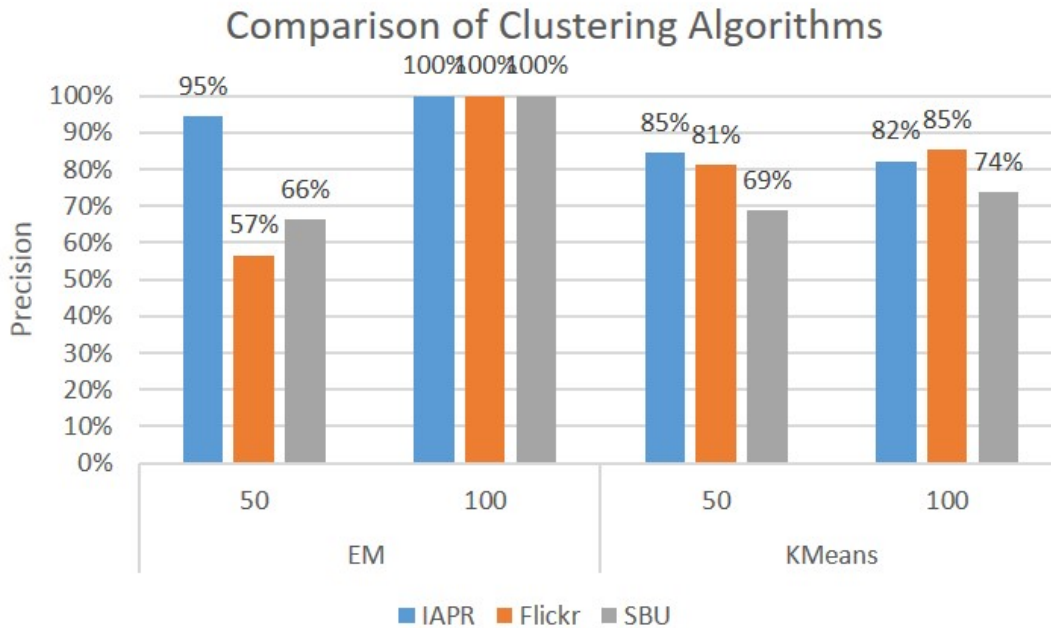


Figure 48: Comparison between EM and K-Means

Naive Bayes) were 72%, 83%, 69% for IAPR, Flickr30K, SBU, respectively. The lowest F-Measures (by Random Forests) were 67%, 22%, 24% for IAPR, Flickr30K, SBU, respectively.

6.5.3.6 Visual Context Model

The top five words representing each cluster for 20 clusters in the IAPR dataset and Flickr dataset are shown in Figures 50- 51, respectively. The images grouped under four unique contexts for IAPR and Flickr30k domains gives a visual perspective to the context-based clusters formed through the process. The contexts discovered are from a vector size of 100 with $K = 20$ with the K-Means clustering algorithm. The framework based images are shown in Figures 52- 53.

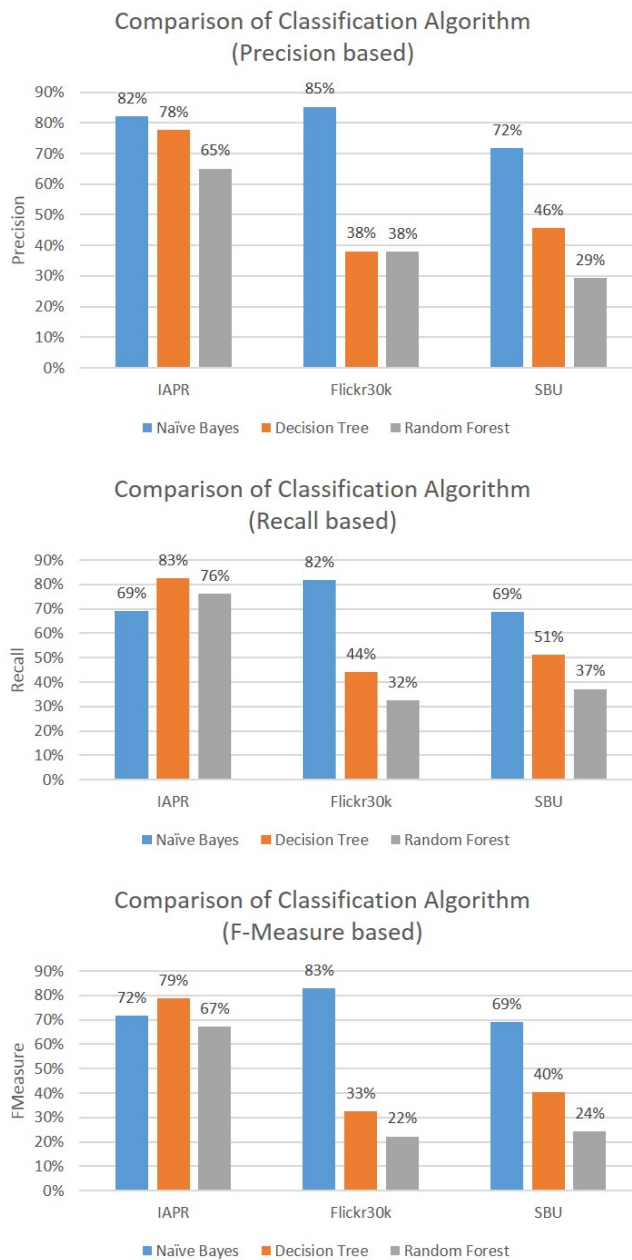


Figure 49: Classification Algorithms: (a) Precision (b) Recall (c) F-Measure

Cluster 0	bus	sky-light	edifice	lamp	public-sign
Cluster 1	vegetable	chair	window	chimney	wall
Cluster 2	man	group-of-person	woman	ground	non-wooden-furniture
Cluster 3	ocean	sky-red-sunset-dusk	sun	hut	palm
Cluster 4	seal	herd-of-mammal	rock	sky-blue	wood
Cluster 5	ocean	dolphin	sky	person	man
Cluster 6	island	ocean	water	chair	floor-other
Cluster 7	bed	curtain	table	floor-carpet	wall
Cluster 8	couple-of-person	food	person	floor	group-of-person
Cluster 9	vegetation	branch	leaf	ground	rock
Cluster 10	wave	ocean	man	surfboard	sky
Cluster 11	man	crab	sand-beach	ocean	vegetation
Cluster 12	cloud	sky-blue	ground	hill	flower
Cluster 13	sky-blue	hut	sand-beach	vegetation	palm
Cluster 14	flower	leaf	butterfly	plant	water
Cluster 15	man	woman	paper	fabric	wood
Cluster 16	roof	wall	cactus	sky-blue	bed
Cluster 17	vegetation	waterfall	rock	water	sky-blue
Cluster 18	public-sign	swimming-pool	plant	door	window
Cluster 19	couple-of-person	face-of-person	child-girl	sky-blue	child-boy

Figure 50: 20 Context Clusters of IAPR Dataset

Cluster 0	kid	play	seesaw	sit	man
Cluster 1	white	out	condition	snow	ground
Cluster 2	young	White	male	outside	bush
Cluster 3	black	dog	tri-colored	each	road
Cluster 4	baseball	cap	black	jacket	stand
Cluster 5	girl	cellphone	skating	Woman	talk
Cluster 6	young	guy	shaggy	hair	hand
Cluster 7	black	dog	white	brown	spot
Cluster 8	man	cook	bean	grill	basket
Cluster 9	people	photo	play	guitar	poke
Cluster 10	man	Germany	jump	rail	shirt
Cluster 11	man	green	guitar	stuff	lion
Cluster 12	guy	joke	man	finishing	Touch
Cluster 13	girl	grass	play	Finger paint	canvas
Cluster 14	man	kitchen	cooking	Food	trendy
Cluster 15	girl	pink	dress	wooden	climb
Cluster 16	worker	piece	equipment	Asian	man
Cluster 17	man	gray	Black	stand	stove
Cluster 18	youth	jump	roadside	Railing	night
Cluster 19	child	pink	dress	climb	stair

Figure 51: 20 Context Clusters of Flickr30K Dataset

Contexts in IAPR

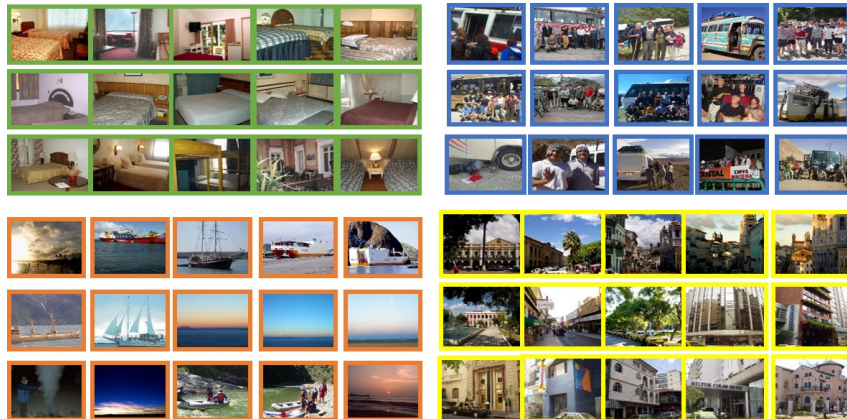


Figure 52: Image Context Clusters of IAPR Dataset

Contexts in Flickr30K



Figure 53: Image Context Clusters of Flickr30K Dataset

6.6 Conclusion

In this chapter, we presented the VisContext framework for context learning from large scale multimedia data. The implementation of the VisContext framework was introduced. The evaluation of the framework was conducted with three large datasets (IAPR, Flickr30k, SBU) in the effectiveness of visual understanding in terms of accuracy. The results confirm the effectiveness in discovering the contextual association of terms and images, visual context clustering, and image classification based on context. The work presented in this chapter was published as part of Chandrashekar et al.[208].

CHAPTER 7

TRANSFORMATION FROM PUBLICATIONS TO ONTOLOGY USING TOPIC-BASED ASSERTION DISCOVERY

7.1 Introduction

In recent years, there has been an explosive growth in the amount of biomedical data being generated, with the majority being unstructured. The majority of this growth can be observed and tracked using the publication databases. These publications represent the novel findings and hypotheses from the research. The dissemination and sharing of biomedical findings to translational medicine are slow, even though most of them are open-source and available to all through publications. The most up-to-date findings of the diagnosis, interventions, and treatments would be critically important. They can be life-changing when used to back critical decisions for their patients by physicians and researchers in health care.

Topic modeling is a frequently used technique to discover latent topics and topical structures in document collections. Latent Dirichlet Allocation (LDA) [209] that is widely used allows documents to have a mixture of topics. We will explore the patterns in the diabetes publication and identify assertions by mapping the topical probability. In this study, we will demonstrate the effectiveness of the ontology generation by integrating topic distribution and assertion discovery.

There is an increasing demand for ontologies that will contribute most to expediting the discovery of new diagnostic treatments and interventions for medical applications such as knowledge retrieval, summarization, medical question answering system. Dynamic and relevant ontologies will be more useful for evidence-based medicine or personalized treatment than general ontologies and significant ontologies.

Most of the existing ontologies are designed extensively by domain experts. Some domain expert generated ontologies are OntoDiabetic [210], Diabetes Diagnosis Ontology (DDO) [211] and Diabetes Mellitus Treatment Ontology(DMTO) [212]. Most of these ontologies are proposed in general domains, with different techniques to automate the extraction tasks. These studies have shown promising results. Nevertheless, experts are always needed because ontology construction and enrichment require a considerable amount of domain knowledge. Text2Onto [213], OntoLearn [214], and Sprat [215] are semi-automatic methods for ontology construction from textual data.

Unlike these ontologies, we may need an automatic approach that can generate an ontology by discovering assertions from free-text sources, such as scientific publications in PubMed. To determine ontological assertions from a free-text corpus, Lossio-Ventura et al.,[216] detected subject or object from a sentence using named-entity recognition (NER), PoS tagging, or Information Extraction techniques. The relationship between subject and object was also detected and classified.

In this chapter, we proposed the Assertion Discovery framework (as shown in Figure 54) that aims to discover bio-medical assertions from free-text sources (like PubMed publications), mapped them to the existing diabetes ontologies, and integrated them with

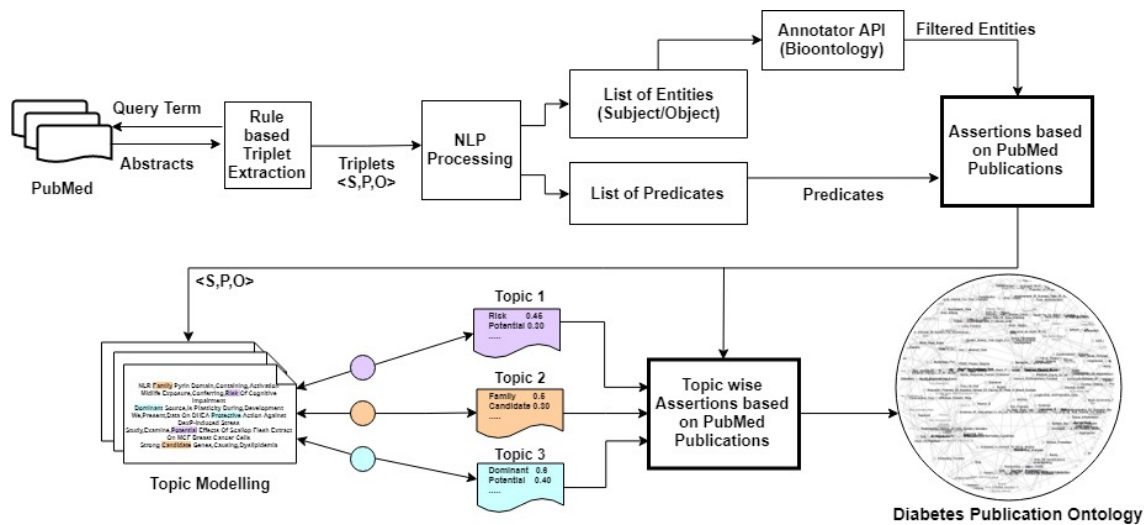


Figure 54: Assertion Discovery Framework

newly found assertions from diabetes publication. We designed a pipeline approach of (i) Natural Language Processing (NLP) [217], (ii) Information Extraction (IE) [218], (iii) Topic Discovery using Latent Dirichlet allocation (LDA) [209] and (iv) Ontology Generation using OWL API [219].

In this chapter, we choose *diabetes* as a case study and use diabetes publications as datasets for topic discovery and assertion refinement. As shown in Table 40, we compared our ontology, Diabetes Publication Ontology (DPO) against the existing diabetes ontologies including Ontology for Genetic Disease Investigations(OGDI) [220], Ontology of Glucose Metabolism Disorder (OGMD) [221], BioMedBridges Diabetes Ontology (DIAB) [222], Diabetes Mellitus Diagnosis Ontology (DDO) [211] and Diabetes Mellitus Treatment Ontology (DMTO) [212].

7.2 Related Work

Many projects [223] have been proposed to process natural language data and create a meaningful ontology that can be used in an information retrieval system. In BioBroker [224], Shen and Lee clustered entities to facilitate the process of biomedical knowledge discovery. A work by Omura [225] is similar to our approach in which the focus is on finding the disease similarity by constructing a graph representing anatomical features. However, the work does not focus on those features that are unique to a disease. In our work, we are fetching the common topics and relationships among ontologies and extracting the unique topics and relationships. Ma et al., [226] aimed to classify the diseases based on their related term. This work is similar to our approach, where the relationships are extracted between three categories of diseases.

Another similar work, Lossio-Ventura [216], focused on automatic knowledge base construction from heterogeneous information sources on Obesity. In this work, they used biomedical entity detection, which is very useful to determine the meaning of the various medical terms present in the chapter. Also, the help of domain experts was used to annotate the medical words manually. For a large text corpus, manual annotation of a corpus would be a tedious task. Also, they explored binary classification for relationship extraction. The binary classification does not discover topics for a particular text corpus if a word belongs to a specific entity or relation.

Fred [227] is a tool that automatically produces RDF/OWL formatted ontologies from natural language sentences. This tool uses multiple natural language processing components to formalize the output to a visual knowledge graph. The generated output

graph is designed according to frame semantics, where each frame is expressed by verbs or other linguistic constructions formalized as OWL n-ary relations. Fred is a domain and task-independent tool suitable for task-specific applications. It changes the input from discourse representation structures to RDF/OWL n-ary relations. It can represent modality, tense, and negation of the sentences. It is capable of handling compositional semantics, taxonomy induction, and quality representation. However, Fred is incapable of handling large datasets for generated ontology visualization. The results are not uniform if both facts and entities are expressed by natural language text. Fred's frame construction based on real-world facts was not what we expected.

Topic modeling techniques were used in biomedical domains. Topic modeling techniques such as Latent Dirichlet Allocation (LDA) [209] and semantic group-based model were used for recommending patient education materials [228]. In this study, the LDA topic model outperformed the vector space model (VSM) and semantic group-based model in the context of patients' question-answering in recommending diabetic education materials.

In this chapter, we use LDA for topic discovery to determine the topic that the authors discuss in each cluster of the triplets. Previous studies identified knowledge structures or topics by adopting the LDA model. There are many attempts [229], [230], in which the LDA model was used to incorporate content analysis into metadata generation. The limitation of these works mainly explains topical trends by using topical terms but failed to extend it to understand bio-medical publications' assertions. Applying the proposed model, we aim to integrate the topics from the LDA model to assertions from

bio-medical publications. We further extended the topic-oriented assertions to the ontology to represent the significant assertions.

Chan et al. [231] aim to provide the structural and digital form of patient records and helps patient care, advice, and clinical decision. The terms related to liver cancer are extracted and mapped to ontological features using the Systemized Nomenclature of Medicine (SNOMED). In our work, we are assigning our terms to several ontologies including Ontology for Genetic Disease Investigations(OGDI) [220], Ontology of Glucose Metabolism Disorder (OGMD) [221], BioMedBridges Diabetes Ontology (DIAB) [222], Diabetes Mellitus Diagnosis Ontology (DDO) [211] and Diabetes Mellitus Treatment Ontology (DMTO) [212].

7.3 Assertion Discovery Framework

In this chapter, we propose the Assertion Discovery framework that aims to construct the Diabetes Publication Ontology (DPO) through a topic-driven approach. In this framework, the PubMed abstracts in diabetes are transformed into the Diabetes Publication Ontology (DPO). The sentences in the publication datasets will be converted into an assertion in a triplet format <Subject, Predicate, Object>. We used state-of-the-art technologies to process and discover significant entities and predicates. In this chapter, an entity is defined as a subject and an object in a triplet. The entity might be later identified as either a concept or an individual in an ontology. A predicate is defined as a relationship between entities. The Assertion Discovery framework consists of three main components: (i) Assertion Discovery, (ii) Assertion Alignment, and (iii) Assertion Integration.

Figure 55 shows the transformation process of the assertion discovery framework, being changed from publications to the DPO ontology (highlighted the assertions associated with *diabetes* and *cardiovascular_risk_factor*). We will discuss each of these steps below.

Table 40: Diabetes Ontologies

Ontology	Publication	Description	#Class	#Individual	#Property
OGMD	(Lin & Sakamoto, 2009)[221]	Ontology of Glucose Metabolism Disorder: Including the disease names	134	0	0
OGDI	(Lin & He, 2014) [220]	Ontology for Genetic Disease Investigations: model the scientific investigation, Genome-Wide Association Study (GWAS), to find out genetic susceptibility factor to diabetes	386	363	109
DIAB	(Vasant et al., 2015) [222]	BioMedBridges Diabetes Ontology	375	0	4
DDO	(El-Sappagh & Ali, 2016) [211]	Diabetes diagnosis ontology: A standardized ontology that collects all of the classes, properties, and axioms of diabetes diagnosis process.	6,444	18	48
DMTO	(El-Sappagh et al., 2018) [212]	Diabetes Mellitus Treatment Ontology: creating customized treatment plans for diabetic patients	10,700	63	315
DPO	Our work	Diabetes Publication Ontology: dynamic ontology generation from diabetes publications	1,802	1,802	870

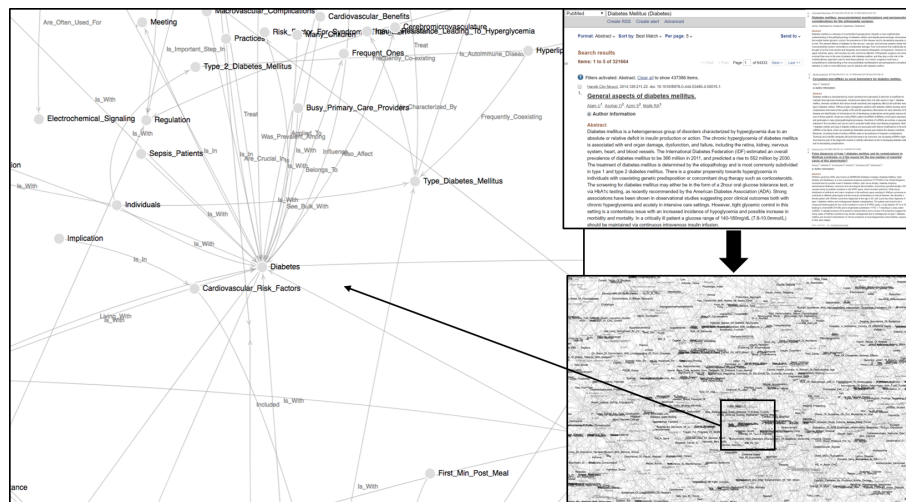


Figure 55: Transformation from Publications to Diabetes Publication Ontology

7.3.1 Assertion Discovery

The assertion is defined as textual facts or textual findings from biomedical research that will be described as a form of $\langle \text{Subject(S)}, \text{Predicate(P)}, \text{Object(S)} \rangle$. The assertion discovery (Figure 54) is based on the integration process of the following two steps: (i) Rule-based Triplet Extraction using OpenIE [218]; (ii) Natural Language Processing (NLP).

7.3.1.1 Rule Based Triplet Extraction

First, the Open Information Extraction (OpenIE) [218] is performed to the PubMed abstract dataset. Open information extraction (OpenIE) refers to the extraction of relation tuples, typically binary relations, from plain text. The OpenIE delivers its result in the form of Quadruple $\langle S, P, O, C \rangle$, where the triplet of Subject(S), Predicate(P), Object(O) is determined when its confidence-score (C) is a higher than a threshold δ . The triplets

extracted from the publication text are based on the rules defined in terms of syntactic, lexical, and semantic patterns in the linguistic form of the English language. The rules act very effectively on the publication dataset, as the sentences are well structured.

Some rules inspired by the existing works [218],[232] were used to extract the triples (subjects, predicates, objects). The rules written in terms on Parts of Speech (PoS) Tagging [233] and Universal Dependencies Representation [234].

7.3.1.2 Natural Language Processing (NLP)

The Natural Language Processing (NLP) was conducted to separate entities and predicates from the triplets. The NLP steps were performed using Stanford CoreNLP Library [217] as follows: (1) Tokenization is the process of breaking sentences into tokens which are the smallest constructs of a huge text data; (2) Lemmatization is the process of separating words into individual morphemes and identify the class of the morphemes; (3) Stopword Removal is the process of removing stopwords from the corpus. For example, the stopwords in English include *able, about, above, according, accordingly*, etc. After performing the NLP operations, the triplets (Subject, Predicate, Object) will be separated into two parts: Entities (Subjects or Objects) and Predicates.

7.3.1.3 Topic Modeling

The topic modeling for the assertion discovery is based on Latent Dirichlet Allocation (LDA) [209]. In Latent Dirichlet Allocation (LDA), topics are discovered through

the probability distribution of topics over documents that are associated with word distribution. Topics are inferred by mapping words into topics associated with documents.

LDA is a Bayesian model where the distributions over the parameters θ and ϕ are modeled, given by Dirichlet distributions with hyperparameters α and β . More specifically, each word n in document d has a variable w_{dn} that represents the observed word type and a latent topic variable δ_{dn} . A word is generated by randomly sampling a value $z_{dn} = k$ from the topic distribution of document θ_d , then sampling a word type $w_{dn} = v$ from the topic (k) distribution ϕ_k . Each word is conditionally independent, given the parameters. Given the parameters θ and ϕ , the probability of a word in the LDA model is:

$$P(w_{dn} = v | \theta_d, \phi) = \sum_k \theta_{dk} \phi_{kv} \quad (7.1)$$

We computed overall term frequency and estimated term frequency within the selected topic according to Termite [235]. Top-30 most salient terms were computed for topic t and term w as follows:

$$sal(w) = freq(w) * \sum_t P(t|w) * \log \frac{P(t|w)}{P(t)} \quad (7.2)$$

The relevance metric was computed to adjust to given a weight parameter λ (where $0 \leq \lambda \leq 1$) according to LDAvis [236] as follows:

$$rel(w|t) = \lambda * (w|t) + (1 - \lambda) * \frac{P(w|t)}{P(w)} \quad (7.3)$$

In this chapter, we extended the LDA model [209] to map topic terms to the triplets

extracted from diabetes publications. The PyLDA [236] was used to discover the hidden topics in the triplets and visualized the inter-topic distance map with PyLDA (as shown in Figure 56).

7.3.1.4 Topic Wise Assertion Discovery

We now select only the important triplets $\langle \text{Subject}(S), \text{Predicate}(P), \text{Object}(O) \rangle$ by matching the topic terms from the LDA-based topic discovery with the entities that were extracted from the OpenIE triplets. As in, we retain only those triplets in which a subject S or an object O matches one of the topic terms of a topic $T_i = \{t_{i1}, t_{i2}, \dots, t_{in}\}$.

In the Topic Wise Assertion Discovery, for any given triplets from the previous step, if Subject(S) or Object(O) from $\langle \text{Subject}(S), \text{Predicate}(P), \text{Object}(O) \rangle$ contain any topic terms $T_i = \{t_{i1}, t_{i2}, \dots, t_{in}\}$ discovered from the LDA topic discovery, the assertion is selected as a topic wise assertion (i.e., $S^{ti} = S \in T_i$ or $O^{ti} = O \in T_i$).

Medical terms were not captured using the rule-based linguistic approach For example, for the given triplet $\langle S: 1680 \text{ mg NaHCO}_3, P: \text{followed by}, O: \text{acid load meal in a double-blind} \rangle$, $S: \text{mg NaHCO}$ was captured while ignoring a number or a special character in the text. The entities from the triplets are mapped to the existing diabetes ontologies such as Diabetes Diagnosis Ontology (DDO) [211] or Diabetes Mellitus Treatment Ontology(DMTO) [212] using the Bioportal Annotator API [237].

7.3.2 Assertion Alignment

For Assertion Alignment, the following steps are designed:

Selected Topic:



Figure 56: Inter-topic Distance Map for Diabetes Publications

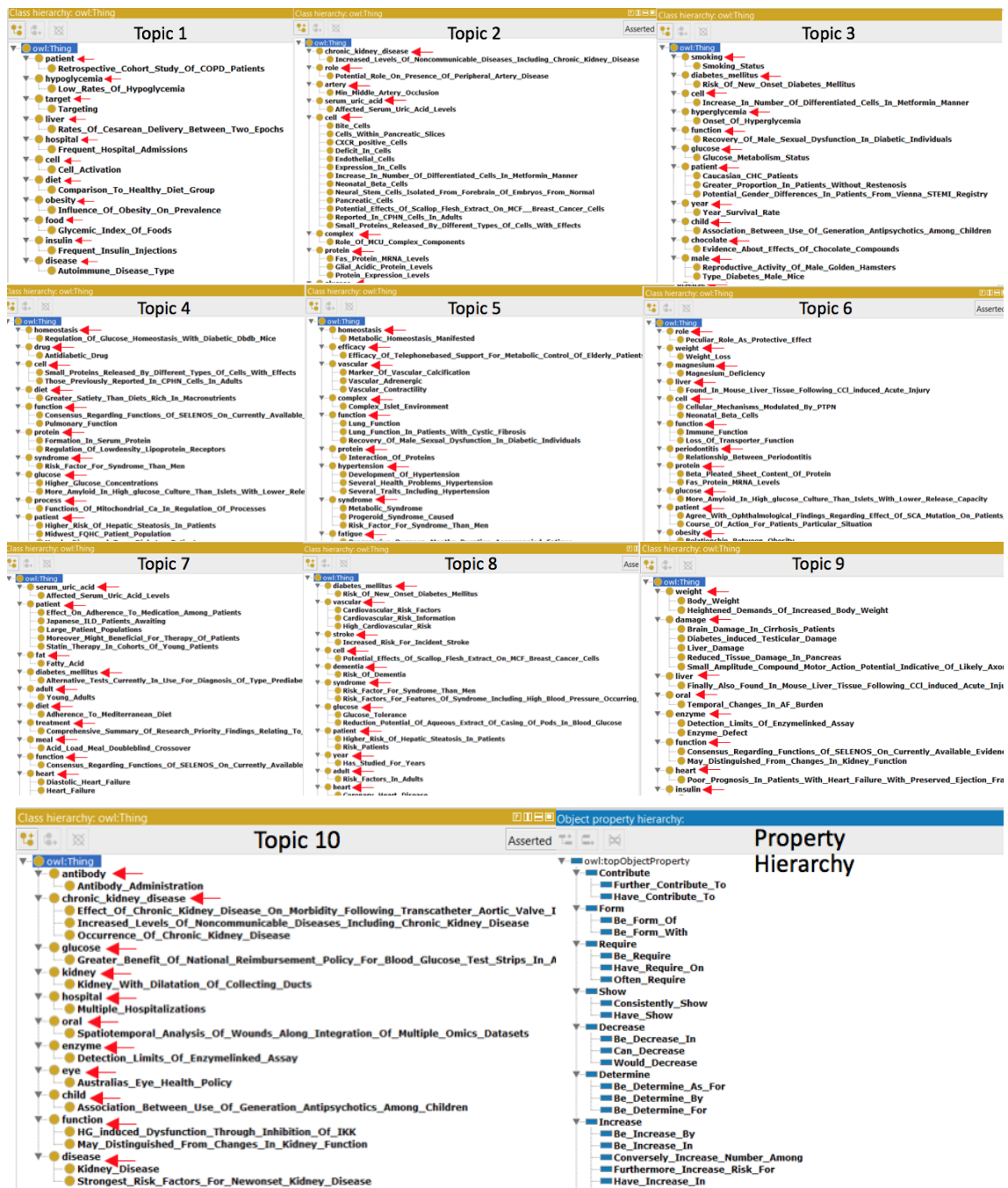


Figure 57: Diabetes Publication Ontology (DPO): Topic Entity Hierarchy (Topics 1- 10) & Property Hierarchy

Entity Learning: The classes and their individuals will be identified in this step. A medical term captured by the Annotator [237] or topic terms discovered by the LDA topic modeling are considered as a class. Individuals for these classes are also generated. Regarding the entity learning, we have not fully covered the schema learning here. In this chapter, we have simplified the schema learning as follows: any entities containing the types (S or O) by using the Annotator [237], these entities, S or O , will be represented as URI_S and URI_O . The entities are mapped to the Diabetes Mellitus Treatment Ontology (DMTO), as shown in Figure 57. The annotated entities from DMTO are shown with a red arrow mark. The entities are a child of the DMTO entity since they partially contains, in its context, that have the same meaning as the formal entity in DMTO.

Property Learning: We are now checking if the objects O in the triplet belong to any of the Classes or Individuals. If they do, the type of property P is Object Property. If not, it is Data Property.

Triplet Learning: Once the entities and properties are identified, the triplets $\langle \text{Subject}, \text{Predicate}, \text{Object} \rangle$ will be generated with S and O from step i) and P from step ii). These triplets $\langle S, P, O \rangle$ will be defined as the assertions in both schema (TBOX) and data (ABOX). $\langle S, P, O \rangle$ will be represented as $\langle URI_S, URI_P, URI_O \rangle$ where URI_S , URI_P , URI_O are defined from the previous steps. Figure 57 shows the topic wise entity hierarchy and property hierarchy in the DPO ontology. The entities were mapped to the entities in DMTO [212] ontology using Annotator API [237]. The topics considered to display here are the topics which had the highest number of entities that could possibly be grouped under the DMTO entity. A very interesting discovery was made among the

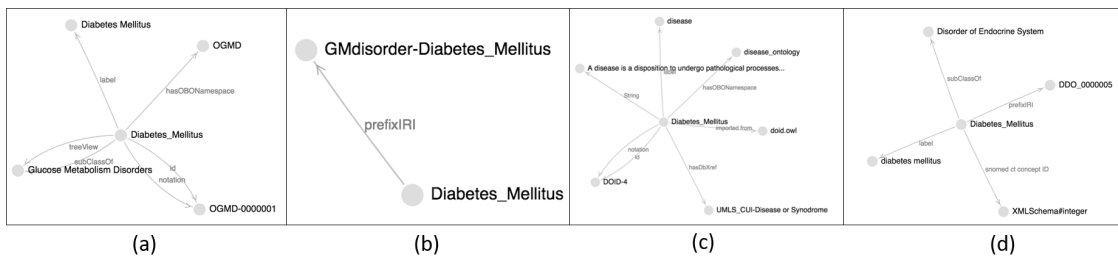


Figure 58: *Diabetes Mellitus* and Associated Assertions in (a) OGMD (b) OGDI (c) DIAB (d) DDO

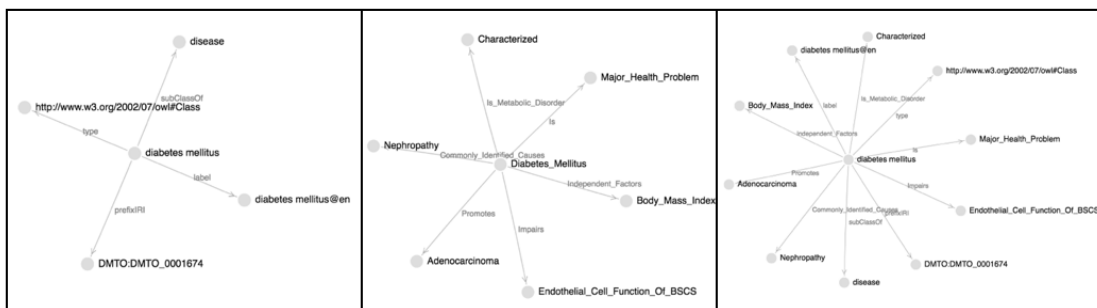


Figure 59: *Diabetes Mellitus* in (a) DMTO, (b) DPO, (c) Integration of DPO and DMTO

entities across the shown topics, the entities were uniquely grouped under each topic. For example, *protein*, (a DMTO Class Entity) exists in Topic 2 as well as Topic 5 but the subclasses in Topic 2: *protein* discusses protein levels and Topic 5: *protein* discusses about the interaction between proteins.

7.3.3 Assertion Integration

In the Assertion Integration step, the newly discovered assertions that were not present in the existing diabetes ontologies will be integrated into existing ontologies. Figure 58 shows the *Diabetes Mellitus* as an entity from (a) OGMD, (b) OGDI, (c) DIAB and (d) DDO, respectively. OGMD is specifically designed from *Glucose Metabolism*

Disorders, while OGDI is designed for *Genetic Disorders* concerning *Diabetes Mellitus*. DIAB is cross-referencing to other famous ontologies such as SNOMED CT, UMLS. DIAB also provides a brief definition of diseases. DDO shows that *Diabetes Mellitus* is a specific case of diabetes where a *disorder of Endocrine System* has occurred. Figure 59a shows the DMTO Ontology's triplets associated with *Diabetes Mellitus*. These are the top five diabetes ontologies, providing excellent structural inferences, but they fail to cover the semantic aspect of diabetes; also, there is a lack of predicates connecting them to potential risks and treatments.

We consider DMTO, the latest ontology (published in Feb 2018), to guide us with structural and schematic information built by experts. Figure 59a shows the entity *Diabetes Mellitus* in DMTO, Figure 59b the entity *Diabetes Mellitus* with newly discovered assertions in DPO. Once we conduct the assertion integration, then we have the integration of assertions in Figure 59a and Figure 59b in DPO as shown in Figure 59c. After integrating the DPO assertions and DMTO structural assertions, we try to construct a holistic ontology.

Some of the predicates discovered are $\langle S: \text{Diabetes Mellitus}, P: \text{Independent Factors}, O: \text{Body Mass Index} \rangle$, $\langle S: \text{Diabetes Mellitus}, P: \text{Commonly Identified Causes}, O: \text{Nephropathy} \rangle$, $\langle S: \text{Diabetes Mellitus}, P: \text{Impairs}, O: \text{Endothelial Cell Function of BSCS} \rangle$. In the last-mentioned triplet, it was clearly determined that the object entity *Endothelial Cell Function of BSCS* was an error when it was processed in the context of the assertion. It is supposed to BSCB (Blood Spinal-Cord Barrier). This chapter shows that NLP and OpenIE techniques could be used in detecting errors by authors of publications.

Table 41: DPO Dataset and Assertions

D	A	W	T	TN	E	P	AE
Dataset 1	100	11,700	490	324	582	229	126
Dataset 2	500	56,000	2,417	1,557	1,802	870	457

D: Dataset; A: Abstract; W: Word; T: Triplet; TN: Triplet after NLP; E: Entity (Concept/Individual); P: Predicate (Object/Data Property); AE: Annotated Entities

7.4 Results and Evaluation

7.4.1 Dataset

We used the PubMed API to access publications from more than 27 million citations for biomedical literature from MEDLINE, life science journals, and online books. We have randomly collected two corpora 100 and 500 publications in diabetes as shown in Table 41 through the PubMed API [238]. For the 100 abstracts and 500 abstracts datasets, there were 11,700 and 56,000 words, respectively.

7.4.2 Results from Topic-based Assertion Discovery

From the Rule-based triplet Extraction and NLP steps of the Assertion discovery (as shown in Table 41), we obtained 490 and 2,417 triplets and 324 and 1,557 triplets after the NLP treatment for the two abstract datasets (100 and 500 abstracts), respectively. The number of entities (Subjects or Objects) discovered from the Assertion Discovery framework for the two datasets are 582 and 1802, respectively. The numbers of predicates are 229 and 870, respectively.

Figure 56 shows 10 topics from each domain with their topic name discovered

from LDA topic modeling. Table 42 shows the entities, the annotated entities, and assertion examples for each LDA topic. We discovered 10 topics from the 1802 entities of the 1,557 triples in the 500 abstract dataset. The 30 topic terms were generated for each topic. Table 42 shows the entities and triplets in each topic of the DPO ontology (10 topics) as well as the annotated entities that were mapped to the other five diabetes ontologies (OGMD, OGDI, DIAB, DDO, DMTO). Topic 2 has the largest number of entities (88 entities), while Topic 9 has the smallest entities (39 entities). Topic 5 has the highest number of the annotated entities (31 entities) mapped to the diabetes ontologies. The entity hierarchy with newly discovered entities and existing annotated entities for each topic are shown in Figure 57. Topics 6, 8, 9, 10 are closely related, as shown in Figure 56. Topics 6, 8, 9, 10 are about Weight Loss, Risk Factors, Body Weight, and Kidney Disease, respectively.

Table 42: DPO Ontology: Entities/Annotated Entities and Assertions per Topic

Topic	#Ent	#Ann.	Example of Annotated Entities	Assertion Example
Topic 1	43	8	Autoimmune Disease Type	<Extensions, Progression_To, Autoimmune_Disease_Type>
Topic 2	88	19	Affected Serum Uric Acid Levels	<Similar_Effects, Were_Observed_In, Human_Brain_Microvascular_Cells>
Topic 3	63	12	Disease Onset	<Measure, Collect, Information_About_Various_Dimensions>
Topic 4	71	14	Risk Factor For Syndrome Than Men	<Antidiabetic_Medications, more_Increase_Risk_For, Heart_Failure>
Topic 5	69	31	Duration Of Disease	<Metabolic_Syndrome, Leading_To, Severe_Health_Problems_Hypertension>
Topic 6	69	17	Peculiar Role As Protective Effect	<Comorbidities, Contribute_To, Increasing_Prevalence_In_Developed_Countries>
Topic 7	54	14	Adherence To Mediterranean Diet	<Intervention_Thresholds, Identified_Women_In, Particularly_Elderly>
Topic 8	77	20	Strongest Risk Factors	<Midlife_Exposure, Conferring, Risk_Of_Cognitive_Impairment>
Topic 9	39	9	Percentage of Body Fat In ZDF Rats	<Strong_Candidate_Genes, Causing, Dyslipidemia>
Topic 10	40	11	Antibody Administration	<Aerobic_Exercise_Training, Be_Effective_Therapy_In, Management_Of_Type>

7.4.3 Results from Ontology Mapping and Integration

In the Ontology Mapping, we used the Biportal API [237] to identify the types of these entities. The annotated entities are detected by the Annotator [237] based on the five diabetes ontologies. As shown in Table 41, the number of entities (annotated entities) that were mapped to existing ontologies is 21.6% (126 out of 582 entities) and 25.3% (457 out of 1802 entities), respectively.

Table 43 shows the comparative analysis among six diabetes including OGMD [221], OGD I[220], DIAB[222], DDO[211], DMTO[212] and our ontology DPO. DMTO has the largest number of triplets, 97,203, while DDO has the largest number of entities, 25,480. OGD I has the largest number of predicates, 1,274 and DPO has the second largest number of predicates, 870. In terms of coverage for both entities and predicates, DPO shows the highest coverage (100%) of entities (terms) or predicates (relations) discovered from publications while most of these diabetes ontologies have an insufficient coverage (0%-13%). This indicates a big gap between ontologies designed by domain experts and terminologies used by researchers in publications. Thus, it would be difficult for ontologies to be readily used in applications such as question answering systems or information retrieval systems.

Figure 60a shows the results from the mapping between the entities in the DPO ontology (DPO_E) and ones in the DMTO ontology ($DMTO_E$). The number of the entities in DMTO ($DMTO_E$) is 10,700 while the number of the entities in DPO (DPO_E) is

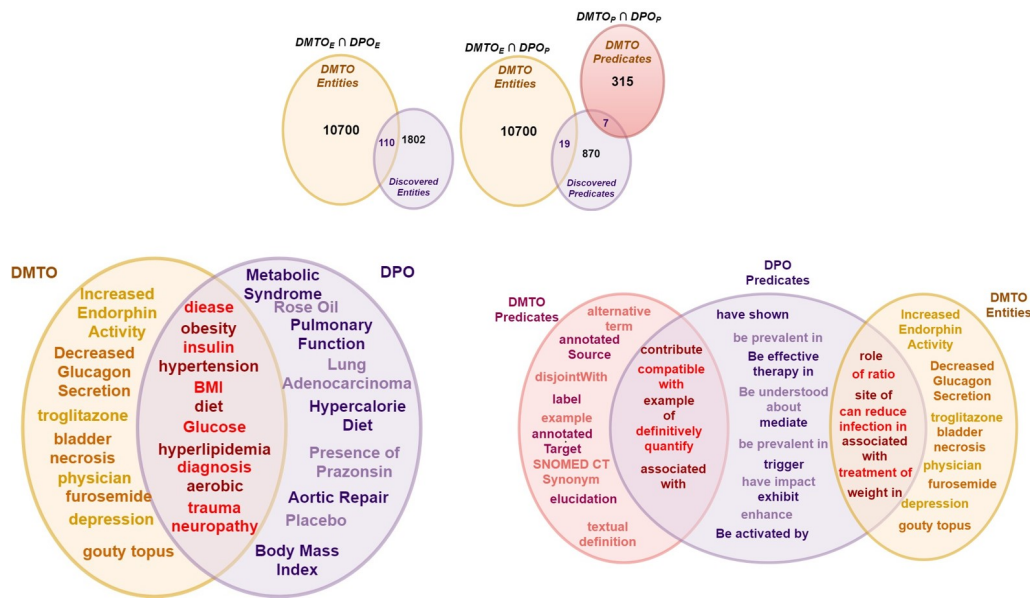


Figure 60: Mapping between DMTO and DPO

(a) Entities in DMTO and DPO ($DMTO_E \cap DPO_E$) (b) Predicates in DMTO and DPO ($DMTO_P \cap DPO_P$) & Entities in DMTO and Predicates in DPO ($DMTO_E \cap DPO_P$)

1,802. The common entities of these two ontologies are 110. The entities in the DPO ontology are very specific and informative. Figure 60b shows the results from mapping between the predicates of the DMTO and DPO ontologies. There are two kinds of mapping: 1) Predicates in DMTO mapped to Predicates in DPO ($DMTO_P \cap DPO_P$) and 2) Entities in DMTO mapped to DPO's Predicates ($DMTO_E \cap DPO_P$). From this figure, the common predication number ($DMTO_P \cap DPO_P$) is 6 while the common entity/predicate ($DMTO_E \cap DPO_P$) is 19. The unique predicates in DMTO ($DMTO_P$) is 315 while the unique predicates in DPO (DPO_P) is 870. Interestingly in the DMTO Ontology when queried, there are only 95 Unique Predicates that contain a possible subject and object associated with it. The rest of predicates (about 315 predicates) are merely declared in the ontology without usage.

Table 43: Mapping between Publications and Ontologies

T: Triplets; E: Entities; P: Predicates; EC: Entity Coverage%; PC Predicate Coverage%

	T	E	P	EC(%)	PC(%)
OGMD[221]	976	435	14	0.66%	0%
OGDI[220]	4,137	4,131	1,274	0.66%	0.68%
DIAB[222]	11,944	4,327	39	0.66%	0.68%
DDO[211]	34,423	25,480	56	0.66%	0%
DMTO[212]	97,203	10,700	315	13%	3%
DPO (ours)	2,417	1,802	870	100%	100%

7.5 Diabetes Publication Ontology Application

The web-based application for the proposed framework has been implemented on the Spark parallel engine [239]. For the dataset collection, we have used the PubMed API [238]. The NLP and OpenIE triplet extraction have been implemented using Stanford CoreNLP [217] and Annotator [237]. The Annotation has been implemented using AngularJS with NCBO BioPortal Annotator API [237]. The Ontology construction has been implemented using OWL API [219] on the Spark. The OWL API is a Java API and reference implementation for creating, manipulating and serializing OWL Ontologies. We are visualizing the generated ontology using the online WebVOWL tool [240]. WebVOWL is a web application for ontology visualization. It implements the Visual Notation for OWL Ontologies (VOWL) by providing graphical depictions for elements of the Web Ontology Language (OWL) that are combined to a force-directed graph layout representing the ontology.

We have built a web-based application for searching diabetes publications as well as retrieving the assertions from the diabetes publications through dynamically generating the DPO ontology. Protégé [241], an open-source ontology editor and framework for

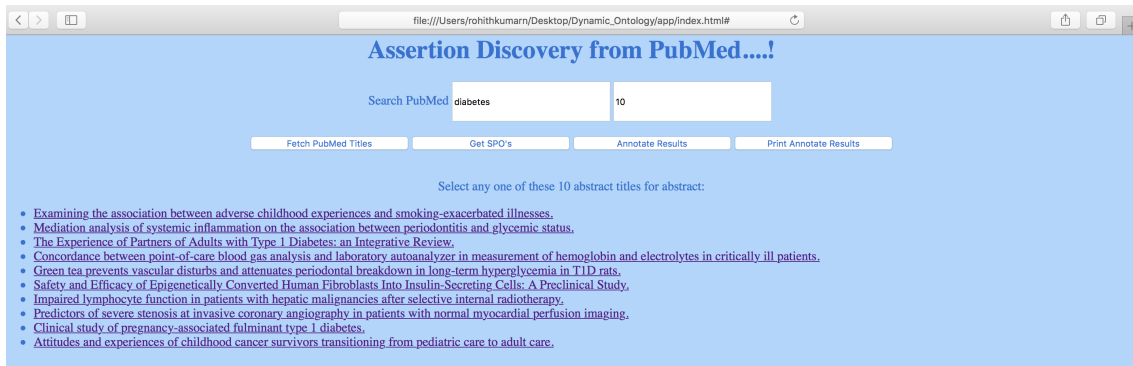


Figure 61: Web Application: Publication Searching

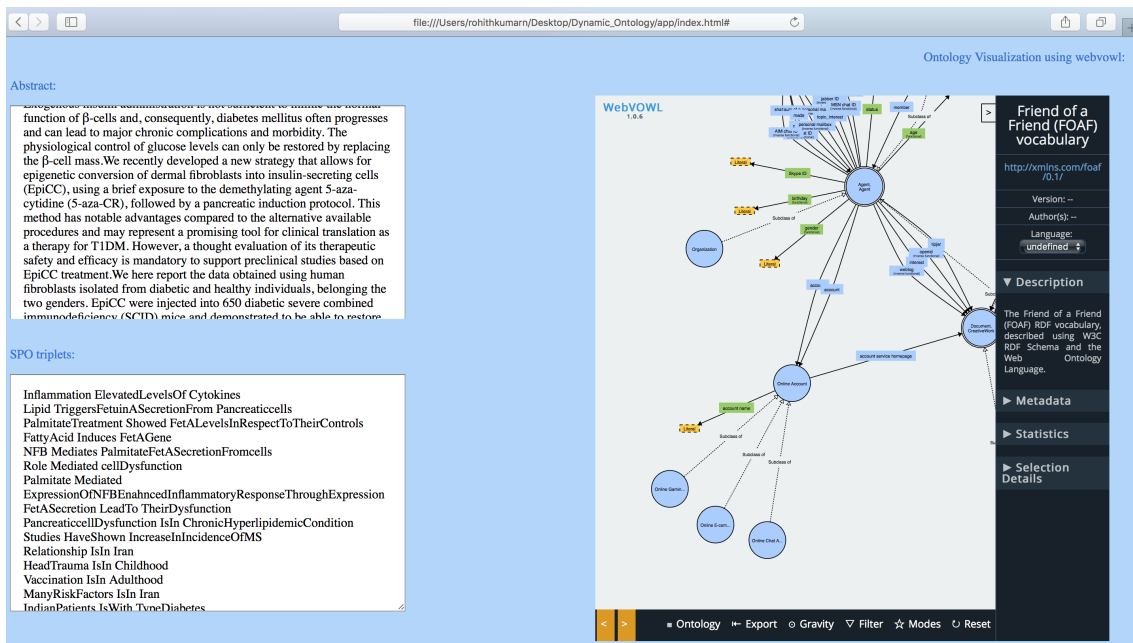


Figure 62: Web Application: Ontology Visualization

Annotated Subjects	List of Predicates	Annotated Objects
Inflammation -> http://data.bioontology.org/ontologies/LOINC Lipid -> http://data.bioontology.org/ontologies/LOINC Palmitate Treatment -> Fatty Acid -> http://data.bioontology.org/ontologies/GRO NFB -> http://data.bioontology.org/ontologies/BIOMODELS Role -> http://data.bioontology.org/ontologies/PR Palmitate -> http://data.bioontology.org/ontologies/LOINC FetA Secretion -> http://data.bioontology.org/ontologies/OBOE-SE Pancreatic-cell Dysfunction -> http://data.bioontology.org/ontologies/VDOT Studies -> http://data.bioontology.org/ontologies/ONTOAD Relationship -> http://data.bioontology.org/ontologies/MESH Head Trauma -> http://data.bioontology.org/ontologies/RH-MESH Vaccination -> http://data.bioontology.org/ontologies/ONTOAD Many Risk Factors -> http://data.bioontology.org/ontologies/SNOMEDCT Indian Patients -> http://data.bioontology.org/ontologies/CSEO Adipose Tissue -> http://data.bioontology.org/ontologies/FLU Whose Accumulation -> http://data.bioontology.org/ontologies/SNOMEDCT VEGFB -> http://data.bioontology.org/ontologies/ONTOPSYCHL Present Investigation -> http://data.bioontology.org/ontologies/GE Administration -> http://data.bioontology.org/ontologies/NCIT Blood Levels -> http://data.bioontology.org/ontologies/AURA Aqueous Extract -> http://data.bioontology.org/ontologies/OGG	ElevatedLevelsOf Triggers Fetuin A Secretion From Showed Induces Mediates Mediated LeadTo IsIn HaveShown IsWith RespondTo AgreeOver Evaluate ResultedIn AqueousExtractOf WasMoreEffectiveIn CaredFor Found ReportedIntercourseAt Reporting PersistAfter SecBulkWith CanAdverselyAffect IsEssentialFor ManagingGlucoseControlIn	Cytokines -> Pancreatic-cells -> http://data.bioontology.org/ontologies/PR Fet A Levels In Respect To Their Controls -> http://data.bioontology.org/ontologies/MESH Fet A Gene -> http://data.bioontology.org/ontologies/PR Palmitate Fet A Secretion From-cells -> http://data.bioontology.org/ontologies/LOINC Cell Dysfunction -> http://data.bioontology.org/ontologies/AURA Expression Of NFB Enhanced Inflammatory Response Through Expression -> http://data.bioontology.org/ontologies/VDOT Their Dysfunction -> http://data.bioontology.org/ontologies/RADLEX Chronic Hyperlipidemic Condition -> http://data.bioontology.org/ontologies/ICNP Increase In Incidence Of MS -> http://data.bioontology.org/ontologies/SNOMEDCT Iran -> http://data.bioontology.org/ontologies/SNOMEDCT Childhood -> http://data.bioontology.org/ontologies/SNMI Adulthood -> http://data.bioontology.org/ontologies/CVAO Type Diabetes -> http://data.bioontology.org/ontologies/HL7 Hyper insulinemia -> http://data.bioontology.org/ontologies/EFO Skeletal Muscles -> http://data.bioontology.org/ontologies/SSE Two Metabolic Effects Of VEGFB -> http://data.bioontology.org/ontologies/RETO Obese Patients -> http://data.bioontology.org/ontologies/GRO Reduction Potential Of Aqueous Extract Of Casing Of Pods In Blood

Figure 63: Web Application: List of Discovered Subject, Predicate, and Object from PubMed Abstracts

Annotated Subjects	List of Predicates	Annotated Objects
Inflammation -> http://data.bioontology.org/ontologies/LOINC Lipid -> http://data.bioontology.org/ontologies/LOINC Palmitate Treatment -> Fatty Acid -> http://data.bioontology.org/ontologies/GRO NFB -> http://data.bioontology.org/ontologies/BIOMODELS Role -> http://data.bioontology.org/ontologies/PR Palmitate -> http://data.bioontology.org/ontologies/LOINC FetA Secretion -> http://data.bioontology.org/ontologies/OBOE-SE Pancreatic-cell Dysfunction -> http://data.bioontology.org/ontologies/VDOT Studies -> http://data.bioontology.org/ontologies/ONTOAD Relationship -> http://data.bioontology.org/ontologies/MESH Head Trauma -> http://data.bioontology.org/ontologies/RH-MESH Vaccination -> http://data.bioontology.org/ontologies/ONTOAD Many Risk Factors -> http://data.bioontology.org/ontologies/SNOMEDCT Indian Patients -> http://data.bioontology.org/ontologies/CSEO Adipose Tissue -> http://data.bioontology.org/ontologies/FLU Whose Accumulation -> http://data.bioontology.org/ontologies/SNOMEDCT VEGFB -> http://data.bioontology.org/ontologies/ONTOPSYCHL Present Investigation -> http://data.bioontology.org/ontologies/GE Administration -> http://data.bioontology.org/ontologies/NCIT Blood Levels -> http://data.bioontology.org/ontologies/AURA Aqueous Extract -> http://data.bioontology.org/ontologies/OGG	ElevatedLevelsOf Triggers Fetuin A Secretion From Showed Induces Mediates Mediated LeadTo IsIn HaveShown IsWith RespondTo AgreeOver Evaluate ResultedIn AqueousExtractOf WasMoreEffectiveIn CaredFor Found ReportedIntercourseAt Reporting PersistAfter SecBulkWith CanAdverselyAffect IsEssentialFor ManagingGlucoseControlIn	Cytokines -> Pancreatic-cells -> http://data.bioontology.org/ontologies/PR Fet A Levels In Respect To Their Controls -> http://data.bioontology.org/ontologies/MESH Fet A Gene -> http://data.bioontology.org/ontologies/PR Palmitate Fet A Secretion From-cells -> http://data.bioontology.org/ontologies/LOINC Cell Dysfunction -> http://data.bioontology.org/ontologies/AURA Expression Of NFB Enhanced Inflammatory Response Through Expression -> http://data.bioontology.org/ontologies/VDOT Their Dysfunction -> http://data.bioontology.org/ontologies/RADLEX Chronic Hyperlipidemic Condition -> http://data.bioontology.org/ontologies/ICNP Increase In Incidence Of MS -> http://data.bioontology.org/ontologies/SNOMEDCT Iran -> http://data.bioontology.org/ontologies/SNOMEDCT Childhood -> http://data.bioontology.org/ontologies/SNMI Adulthood -> http://data.bioontology.org/ontologies/CVAO Type Diabetes -> http://data.bioontology.org/ontologies/HL7 Hyper insulinemia -> http://data.bioontology.org/ontologies/EFO Skeletal Muscles -> http://data.bioontology.org/ontologies/SSE Two Metabolic Effects Of VEGFB -> http://data.bioontology.org/ontologies/RETO Obese Patients -> http://data.bioontology.org/ontologies/GRO Reduction Potential Of Aqueous Extract Of Casing Of Pods In Blood

Figure 64: Web Application: List of Filtered Entities (Subject and Object) using Annotator API [237]

building intelligent systems, can be used to query the ontology and retrieve papers and assertions from the DPO ontology.

The Web application contains the following parts: (i) *Publication Search Widget*: The users can input their query in terms of keywords and the number of papers to be retrieved (Figure 61); (ii) *Abstract Display Widget*: This displays the abstract obtained using the PubMed API. The abstract is shown in the display with a list of assertions (Figure 62); (iii) *Assertion Widget*: This widget will display all the assertions that were dynamically generated after the Assertion discovery step with all the abstracts selected above (as shown in Figure 63). Assertion Discovery step of our model is incorporated into this widget; (iv) *Ontology Widget*: This is the widget where the WebVOWL visualization [240] of the ontology dynamically generated for the given abstract will be displayed (as shown in Figure 64). (v) *Annotation Widget*: This is the widget where the extracted $\langle \text{Subject(S)}, \text{Predicate(P)}, \text{Object(S)} \rangle$ are annotated with bio-portal API [237] to validate in any existing ontologies. The Assertion Alignment process of the Diabetes Publication Ontology (DPO) model is shown in this widget.

7.6 Conclusion

In this chapter, we proposed a semantic framework for dynamically generating a diabetes publication ontology from a large corpus of diabetes publications. The popular topic modeling method LDA was very effective in finding latent topics. These topic terms were mapped to ontological assertions extracted from the scientific publications in

PubMed. The ontological assertions were used to enhance the existing diabetes ontologies and new relations and entities between topics were introduced to existing diabetes ontologies. The proposed framework has been implemented in a parallel and distributed computing engine, Apache Spark for providing a scalable solution for a large amount of data. The parallel and distributed pipeline approach includes CoreNLP for Natural Language Processing, OpenIE (Open Information Extraction) for relation extraction, and LDA (Latent Dirichlet Allocation) for topic discovery and OWL API for ontology generation. We presented a web-based application that aims to provide enriched information to healthcare providers in diabetes patient care and treatments through the interface for searching publications as well as retrieving the assertions from the publications. The application can be used in many practical cases, providing knowledge and evidence to support decisions in healthcare. The work presented in this chapter was published as part of Nagulapati et al. [242]. Nagabhushan et al. [223] and Chandrashekar et al. [243] are some related publications.

CHAPTER 8

ONTOLOGY MAPPING FRAMEWORK WITH FEATURE EXTRACTION AND SEMANTIC EMBEDDINGS

8.1 Introduction

There is an increasing demand for ontologies that will contribute most to expediting the discovery of new diagnostic treatments and interventions for medical applications such as knowledge retrieval, summarization, medical question answering system. Understanding the relationships among the classes from multiple ontologies will be more useful for evidence-based medicine or personalized treatment than general ontologies or more prominent ontologies.

The National Center for Biomedical Ontology (NCBO) BioPortal [244] introduced about 690 ontologies for diverse applications such as data integration [245], data annotation [246], data interoperability [247], and data encoding in electronic health records (EHRs) [248]. Some of these ontologies are large and complex. Due to the size and complexity of ontologies, understanding or extracting relations between classes and axioms either within an ontology or across ontologies is not feasible.

Shen and Lee [224], [249] presented algorithms to find relations across multiple ontologies that were significant in biomedical research. The contribution of these works is in using a graph clustering method and an indexing technique to discover knowledge patterns over a set of interlinked data sources and query execution with the Bio2RDF data.

SAMBO [250] was proposed for matching and merging biomedical ontologies through (i) lexical similarity based on n-gram, edit distance, (ii) structural similarity based on is-a, part-of hierarchies, (iii) knowledge-based similarity using WordNet and UMLS. Falcon [251] proposed (i) partitioning ontologies using clustering, (ii) matching structural blocks (GMO), and (iii) discovering structural proximities and alignments.

Most of the existing ontologies are designed extensively by domain experts. Some of the domain experts generated ontologies are OntoDiabetic [210], Diabetes Diagnosis Ontology (DDO) [211] and Diabetes Mellitus Treatment Ontology(DMTO) [212]. Most of these ontologies are proposed in general domains, with different techniques to automate the extraction tasks. These studies have shown promising results. Nevertheless, experts are always needed because ontology construction and enrichment require a considerable amount of domain knowledge. Text2Onto [213], OntoLearn [214], and Sprat [215] are semi-automatic methods for ontology construction from textual data.

In this chapter, we proposed an ontology mapping framework (as shown in Figure 65) that aims to discover the relationships (mapping and connectivity) from the analysis of semantic features across multiple ontologies and identify the abstractions of the ontological relationships through mapping between features to the ontologies. We designed a pipeline approach of machine learning with the ontological properties (concept terms and other related terms) from multiple ontologies in a domain as follows:

- Ontology search and selection
- Feature extraction based on ontological properties (concept terms and other related terms),

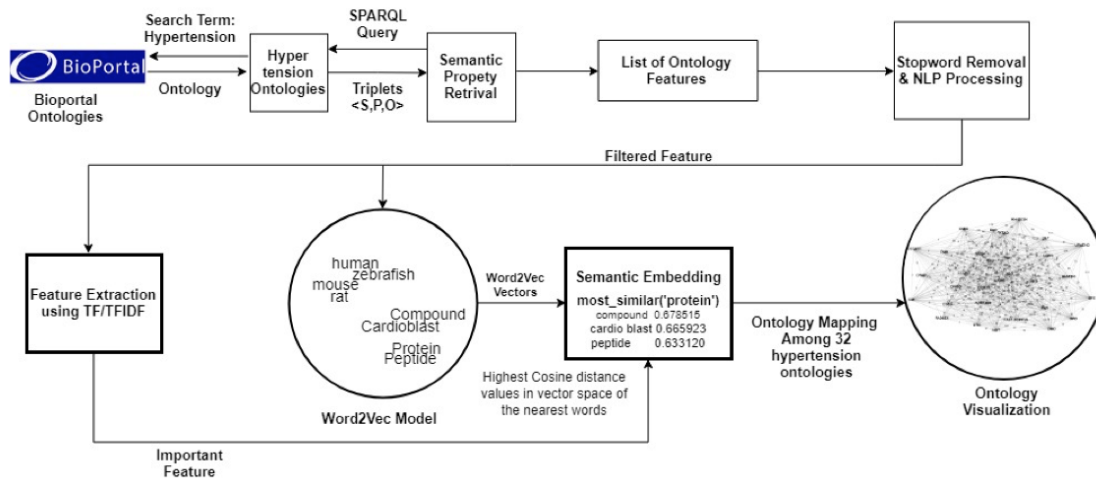


Figure 65: Ontology Mapping Framework

- Natural Language Processing with ontological features,
- Feature Extraction using Term Frequency (TF) and Term Frequency and Inverse Document Frequency (TF-IDF)
- Word Embeddings using ontological properties,
- Semantic Embeddings by combining the feature extracting techniques and Word Embeddings,
- Ontology Mapping and Connectivity across ontologies using Ontology Mapping Graphs.

As a case study, we choose 32 ontologies in the hypertension domain for the analysis of ontology mapping and connectivity, as shown in Table 44.

8.2 Related Work

The previous works developed Abstraction Networks for ontologies [252], [253], in which the structures and contents of various ontologies are identified and classified. In Ochs et al. [254], more than 300 NCBO BioPortal ontologies were analyzed and classified into the abstractions of ontologies defined by the structural meta-ontology. In Mortensen et al. [255], ontology design patterns (ODPs) were proposed for utilizing reoccurring models and best practices in ontology development and maintenance for biomedical ontologies.

Similar to our work, in Noy et al. [256], more than 30,000 mappings were extracted from 20 ontologies in BioPortal. However, this work is limited since some of these mappings were created manually by developers of biomedical ontologies as follows:

- The connects of Gene Ontology, ICD-9, FMA, and the NCI Thesaurus are mapped through the concepts in UMLS Metathesaurus,
- Preferred names and synonyms for concepts in the domain ontologies,
- Relate concepts through `rdfs:seeAlso` or `obo:xref`,
- Manual mappings between concepts in the NCI Thesaurus and the Mouse anatomy ontology
- mapping based on the lexical comparison in the PROMPT mapping algorithm for the mappings between the NCI Thesaurus and the Galen ontology

Many projects have been proposed to process natural language data and create a

meaningful ontology for building an information retrieval system. The work by Omura [225] is similar to our approach in which the focus is on finding the disease similarity by constructing a graph representing anatomical features, and its local information. But the work doesn't focus on those features that are unique to a disease. In our work, we are fetching the common topics and relationships among ontologies and the unique topics and relationships.

Lossio-Ventura [216] focused on automatic knowledge base construction from heterogeneous information sources on Obesity. In this work, they used biomedical entity detection, which is very useful to determine the meaning of the various medical terms present in the paper. Also, the help of domain experts was used to annotate the medical words manually. For a large text corpus, manually annotating the words would be a tedious task. Also, they explored binary classification for relationship extraction. The binary classification doesn't discover topics for a particular text corpus instead of classifies based on if a word belongs to a distinct entity or relation. In our work, we applied NLP, information extraction techniques to ontologies to generate semantic embeddings. We will further extend the semantic embeddings to ontology annotation and relation extraction.

Chan et al. [231] provided a structural and digital form of patient records and helped inpatient care, advice, and clinical decision. The terms related to liver cancer are extracted and mapped to ontological features using SNOMED (Systemized Nomenclature of Medicine). In our work, we mapped semantic embeddings we generated from NLP and information retrieval to the hypertension domain's ontologies.

8.3 Ontology Mapping Framework

In this chapter, we propose the Ontology Mapping framework (OMF) that aims to construct a dynamic ontology based on users' interests through the assertion discovery approach of Natural Language Processing (NLP), information retrieval (IR), and machine learning (ML). In this framework, the ontological properties (concept terms and other related terms) from base ontologies are used to discover new assertions dynamically. The Ontology Mapping framework (OMF) consists of three main components: We designed the pipeline approach in OMF as follows: (i) Ontology search and selection using the NCBO BioPortal API [237] with a keyword *hypertension*, (ii) Feature extraction based on ontological properties (concept terms and other related terms), (iii) Feature vector generation using word embedding technique, Word2Vec, (iv) Topic discovery with the semantic feature vector, (v) Assertion discovery through dynamic ontology modeling and alignment.

Figure 65 shows the assertion discovery process in the OMF framework, being changed from a list of assertions from the base ontologies to new types of assertions. We will discuss each of these steps below.

8.3.1 Ontology Search and Selection

First, we have collected ontologies through the NCBO BioPortal API [237] for the *hypertension* domain. Ontologies were searched using the search keyword *hypertension*. Second, related ontologies were further searched through the *Reuses in Other Ontologies* option (e.g., MeSH and RH-MeSH). Finally, a refined set of ontologies were selected

based on (i) ontology format: OWL, RDF, TTL, N-TRIPLE were supported. While OBO, OFN, OWL2, and xrdf were not supported in our framework. (ii) public availability: some of them were private and not downloadable, and (iii) semantic annotation: ontologies contain the following OWL or RDF properties that would be used to extract ontological properties such as *label*, *hasExactSynonym*, *hasRelatedSynonym*, *hasNarrowSynonym*, and *prefLabel*.

8.3.2 Semantic Property Retrieval

For ontology mapping and understanding of their relationships, we have started to extract the key features considering ontological features (*label*, *hasExactSynonym*, *hasRelatedSynonym*, and *prefLabel*) for the given ontologies. We further extract more important features from the base ontologies' ontological properties that provide more structured features such as concept names and synonyms, etc.

8.3.2.1 Natural Language Processing

The Natural Language Processing (NLP) was conducted to separate concepts and predicates from the triplets. The NLP steps were conducted using the Stanford CoreNLP library [217] as follows: (1) Tokenization is the process of breaking sentences into tokens which are the smallest constructs of a huge text data; (2) Lemmatization is the process of separating words into individual morphemes and identify the class of the morphemes; (3) Stopword Removal is the process of removing stopwords from the corpus. For example, the stopwords in English include *able*, *about*, *above*, *according*, *accordingly*, etc. After

performing the NLP operations, the triplets (Subject, Predicate, Object) will be separated into two parts: Concepts (Subjects or Objects) and Predicates.

8.3.2.2 Feature Extraction

For the purpose, we used two feature extraction techniques: (i) Term Frequency (TF) and (ii) Term Frequency and Inverse Document Frequency (TF-IDF). We have extended the term frequency (TF) and the term frequency-inverse document frequency (TF-IDF) that are a numerical statistic (a term is an ontological feature and a document is an ontology). Thus, the term frequency (TF) as the number of annotated terms that appear in a specific ontology. The representative terms for feature extraction are selected using TF-IDF, which is the product of term frequency (TF) and inverse document frequency (IDF). We redefined the Document frequency (DF) to the frequency of the terms in all the ontologies. The Inverse Document Frequency (IDF) intends to reduce the importance of the word that occurs most frequently in all the ontologies. It is mainly used to eliminate the common terms across all the ontologies. The IDF value is computed by dividing the number of ontologies with the number of ontologies containing the given term t and then applying a logarithm to the resultant value. If the term appears in more ontologies, it is more likely to be a common term that is not specific to any given ontology. Hence, the log value of the word reduces to zero, ensuring that the IDF value and thereby the TF-IDF value, is less for this term.

$$TF(t, o) = 1 + \log(f_{t,o}) \quad (8.1)$$

$$IDF(t, O) = \log \frac{N}{1 + |\{o \in O : t \in o\}|} \quad (8.2)$$

where N is the total number of the ontologies in the corpus, i.e., $N = |O|$ and $|\{o \in O : t \in o\}|$ is the number of ontologies where the term t appears (i.e., $TF(t, o) \neq 0$).

TF-IDF value is high if the term has high term frequency and a low document frequency in the corpus. Hence by considering the TF-IDF value, we can eliminate the common terms in determining features.

$$TF - IDF(t, D) = TF(t, d) \cdot IDF(t, D) \quad (8.3)$$

The TF-IDF score reflects how important an ontological feature is to an ontology in a corpus of ontologies. The top X important words can be extracted based on the weights of the TF-IDF terms (e.g., $X=50$).

8.3.3 Semantic Embeddings using Word2Vec

For the feature vector generation, we used Word2Vec [257], which was designed to produce word embeddings. Word2Vec is a two-layer neural network model for finding contexts of words in the corpus. Word2Vec can utilize two types of model architectures either continuous bag-of-words (CBOW) or continuous skip-gram. In the CBOW model, the current word is predicted from surrounding context words (bag-of-words assumption). In the skip-gram model, the surrounding context words will be predicted from the current

word by weighing more to nearby context words. [Note: *word* and *term* are interchangeably used in this chapter.]

In this chapter, we used the skip-gram model of Word2Vec with negative sampling [54] in which a window is defined to learn word embeddings. The model defines a context based on the frequently co-occurred terms in the same dimensional space of an ontology corpus. The model builds similar embeddings for target words t that co-occur with similar contexts c . The objective function is defined to optimize embeddings as follows:

$$L = \sum_{(t,c) \in P} L_{t,c} \quad (8.4)$$

$$L_{t,c} = \log \alpha(v'_c \cdot v_t) + \sum_{r \in R_{t,c}} \log \alpha(v'_r \cdot v_t) \quad (8.5)$$

where v_t and v'_c are the vector representations of target word t and context word c . P is a set of co-occurring target-context pairs (t, c) within a defined size of window, and $R(t, c)$ is a set of randomly sampled context words c used with the pair (t, c) .

In this chapter, we have uniquely applied the Word2vec model to a large corpus of ontological properties (*label*, *hasExactSynonym*, *hasRelatedSynonym*, and *prefLabel*) that were extracted from the ontologies to produce a vector space with k dimensions. Each unique word in the corpus was assigned a corresponding vector in the space. In the vector space, if an ontological property shares common contexts with the other ontology, they will be located near one another in the space.

We have constructed the semantic embeddings by mapping significant features

(TF or TF-IDF) to Word Embeddings (Word2Vec). We have outlined the notable features X ($X = 200$) using TF and TF-IDF measurements to the base terms in the word embeddings using Word2Vec with window size W ($W = 20$) and cosine similarity. The Semantic Embeddings are representative embeddings selected by integrating Word Embedding (Word2Vec) with Feature Extraction (TF or TF-IDF).

8.3.4 Ontology Mapping Graph

In this step, we have mapped the terms in the semantic embeddings to concepts of the 32 hypertension ontologies. We have applied partial matching for this mapping by using the *contained* operation (e.g., $(\text{contains}(\text{hypertension}, c))$, where *hypertension* is contained in the concept c). The ontology-concept and embedding matrix is designed to identify the matched concepts from this ontology mapping. From this analysis, we also find the degree of connectivity based on the matched concepts of the ontologies. In the final step of the ontology mapping process, the connectivity between ontologies is defined as an edge in a graph, called Ontology Mapping Graph. More specifically, an ontology O_x is defined as a node in the graph G and two ontologies is connected (O_x, O_y) if a concept in the semantic embeddings in an ontology (O_x) is contained any concepts (S or O) of another ontology (O_y) in the hypertension domain. This represents the local edge-connectivity of the ontologies that are symmetric.

The ontology mapping graph represents the degree of connectivity. Two ontologies are k -edge-connected if the edge between the two ontologies is connected with the degree of k . A graph is called k -edge-connected if its edge connectivity is k or greater.

The ontology mapping graph is maximally connected if its connectivity equals its minimum degree ($k = \# \text{Ontology} - 1$). In the hypertension context of the 32 ontologies, an ontology can be mapped to most 31 ontologies (i.e., $k=31$). The ontology mapping graph is minimally edge-connected if its edge-connectivity equals its minimum degree ($k = 0$). Furthermore, we extended the connectivity concept to the concepts across ontologies. The concept graph across ontologies are t -edge-connected if the edge between the two concepts across ontologies are connected with the degree of t . The ontology mapping graph is strongly connected if its edge-connectivity $t > 100$.

8.4 Results and Evaluation

The OMF implementation and experiments were conducted on a 16GB RAM Machine. Ontologies were extracted from the NCBO BioPortal using Biportal Rest API [237]. The queries on the ontologies were performed with Apache Jena [258] with the SPARQL Wrapper. TF-IDF and Word2Vec Model were implemented with Apache Spark 2.2.0. Visualization of the Ontology Mapping Graph was done using Gephi 0.9.2 [259]. Pandas 0.22.0 [260] was used for the implementation of a pipeline for data analysis such as cleaning data, analyzing and modeling, then organizing the results in the tabular display or by plotting.

8.4.1 Dataset

For the experiment, 32 ontologies (shown in Table 44) were searched and selected from the NCBO BioPortal that contains more than 690 ontologies in biomedical domains.

Initially, we had 41 ontologies using the search keyword *hypertension* through the BioPortal API [237]. This dataset was extended to 55 ontologies through the *Reuses in Other Ontologies* option (e.g., MeSH and RH-MeSH) (shown in Table 45). From the 55 ontologies, we have a refined dataset of 32 ontologies (shown in Table 44) that are selected based on (i) ontology format: OWL, RDF, Turtle, N-Triples were supported while OBO, OFN, OWL2, and xrdf were not supported in our framework (as shown in Table 46) and (ii) publicly available: some of them were private and not downloadable.

8.4.2 Results of Semantic Property Extraction

From the 32 ontologies, we extract features by querying semantic properties: the following OWL or RDF properties were used to extract ontological features including *label*, *prefLabel*, *hasExactSynonym*, *hasRelatedSynonym*, *hasNarrowSynonym*, as shown in Table 47. In most of the ontologies gave useful semantic concepts from the *label* and *prefLabel*. Two major patterns were observed during the domain expert-designed ontologies. In the first pattern, Concepts were named based on the complete semantic meaning, such as ADO URI for NSAID medication is named based on the same while the second type, Concepts were mainly had URI based on Identification Numbers such as CRISP (as shown in Table 48). Ontological features were extracted from the *medication* concept in the 32 hypertension ontologies, as shown in Table 48. In this table, some interesting features are extracted, for example, *Studying medication history*, *medication adherence behavior*, and *time-release medication*.

Table 44: Ontology Dataset in Hypertension Domain

Acronym	Ontology Name	#Class	#Individual	#Property
ACGT-MO	Cancer Research and Management ACGT Master Ontology	1770	61	260
ADO	Alzheimer's disease ontology	1565	0	12
BAO	BioAssay Ontology	7192	1	223
BDO	Bone Dysplasia Ontology	3668	0	19
CCONT	Cell Culture Ontology	20844	0	61
COSTART	Coding Symbols for a Thesaurus of Adverse Reaction Terms	1707	0	1
CRISP	Computer Retrieval of Information on Scientific Projects Thesaurus	9039	0	7
CSEO	Cigarette Smoke Exposure Ontology	20085	0	91
DIAB	BioMedBridges Diabetes Ontology	375	0	4
DMTO	Diabetes Mellitus Treatment Ontology	10700	63	315
DOID	Human Disease Ontology	12694	0	15
DTO	Drug Target Ontology	10075	0	0
EFO	Experimental Factor Ontology	20637	0	0
FAST-TOPICAL	FAST (Faceted Application of Subject Terminology) Topical Facet	3	583314	0
HFO	Heart Failure Ontology	1652	0	0
HL7	Health Level Seven Reference Implementation Model	9866	0	63
ICD9CM	International Classification of Diseases, Version 9 - Clinical Modification	22533	0	8
IFAR	Fanconi Anemia Ontology	4530	0	0
KTAO	Kidney Tissue Atlas Ontology	2201	13	163
LOINC	Logical Observation Identifier Names and Codes	2000674	0	0
MESH	Medical Subject Headings	268289	0	38
NATPRO	Natural Products Ontology	9465	22012	64
NCIT	National Cancer Institute Thesaurus	138291	0	97
OMIM	Online Mendelian Inheritance in Man	98642	0	15
ONS	Ontology for Nutritional Studies	3442	104	66
PMA	Portfolio Management Application	2084	0	9
RADLEX	Radiology Lexicon	46656	47155	94
RCTV2	Read Clinical Terminology Version 2	88854	0	0
RH-MESH	Robert Hoehndorf Version of MeSH	305349	0	1
RPO	Resource of Asian Primary Immunodeficiency Diseases (RAPID) Phenotype Ontology	1545	0	166
SSE	Surgical Secondary Events	244	1323	19
UPHENO	Combined Phenotype Ontology	86448	4	142

Table 45: Reuses in Other Ontologies

Base Ontology	Reuses in Other Ontologies
MeSH	RH-MeSH
DOID	BAO, NATPRO, DTO
HP	NIFSTD, BDO, DIAB, KTAO
EFO	CCONT, ONS
DDO	DMTO
MP	UPHENO
APAONTO	APADISORDERS
OF	APO

Table 46: Ontology Formats of the NCBI BioPortal [244]

Format	Description	OMF Support
OWL	is a family of knowledge representation languages for authoring ontologies. The OWL languages are characterized by formal semantics.	Yes
RDF	RDF Schema provides a data-modelling vocabulary for RDF data. RDF Schema is an extension of the basic RDF vocabulary that is a standard model for data interchange on the Web.	Yes
Turtle	Turtle (Terse RDF Triple Language) is a format for expressing data in the RDF data model	Yes
N-Triples	N-Triples is a format for storing and transmitting data. It is a line-based, plain text serialisation format for RDF graphs, and a subset of the Turtle format	Yes
OBO	The OBO format is a format for sharing the use of controlled vocabularies across different biological and medical domains.	No
OFN	OWL functional notation	No
OWL2	Version 2 of the Web Ontology Language	No
xref/Dbref	Xref and Dbxref are properties for support OBO mappings by referring it to an analogous term in another vocabulary	No

Table 47: Ontology Properties for Extracting Ontological Features

Property	Description	URL
<i>label</i>	Instance of <code>rdf:Property</code> to provide a human-readable version of a resource's name.	< http://www.w3.org/2000/01/rdf-schema#label >
<i>prefLabel</i>	The preferred lexical label for a resource, in a given language. The range of <code>skos:prefLabel</code> is the class of RDF plain literals. A resource has no more than one value of <code>skos:prefLabel</code> per language tag.	< http://www.w3.org/2004/02/skos/core#prefLabel >
<i>hasExactSynonym</i>	Instance of <code>owl:AnnotationProperty</code> to describe the exact synonyms. This property was defined in PRO	< http://www.geneontology.org/formats/oboInOwl#hasExactSynonym >
<i>hasRelatedSynonym</i>	Instance of <code>owl:AnnotationProperty</code> to describe the related synonyms. This property was defined in PRO	< http://www.geneontology.org/formats/oboInOwl#hasRelatedSynonym >
<i>hasNarrowSynonym</i>	Instance of <code>owl:AnnotationProperty</code> to describe the specific synonyms. This property was defined in PRO	< http://www.geneontology.org/formats/oboInOwl#hasNarrowSynonym >

Table 48: Ontology Feature Match Example: Medication

Ontology	URI	Feature
ADO	http://scai.fraunhofer.de/AlzheimerOntology#Studying_medication_history	Studying medication history
ADO	http://scai.fraunhofer.de/AlzheimerOntology#NSAID_medication	NSAID medication
ADO	http://scai.fraunhofer.de/AlzheimerOntology#Experienced_side_effects_from_past_medications	Experienced side effects from past medications
CCONT	http://www.ebi.ac.uk/efo/EFO_0006344	medication adherence behavior
CRISP	http://purl.bioontology.org/ontology/CSP/0962-7240	time release medication
CRISP	http://purl.bioontology.org/ontology/CSP/1385-2694	self medication prescribed
CRISP	http://purl.bioontology.org/ontology/CSP/0962-5947	medication error
CRISP	http://purl.bioontology.org/ontology/CSP/0962-5804	self medication nonprescribed
DMTO	https://biportal.bioontology.org/ontologies/DMTO.owl#DMTO_0000911	medication history
EFO	http://www.ebi.ac.uk/efo/EFO_0006344	medication adherence behavior
FAST-TOPICAL	http://id.worldcat.org/fast/1111464	Self medication

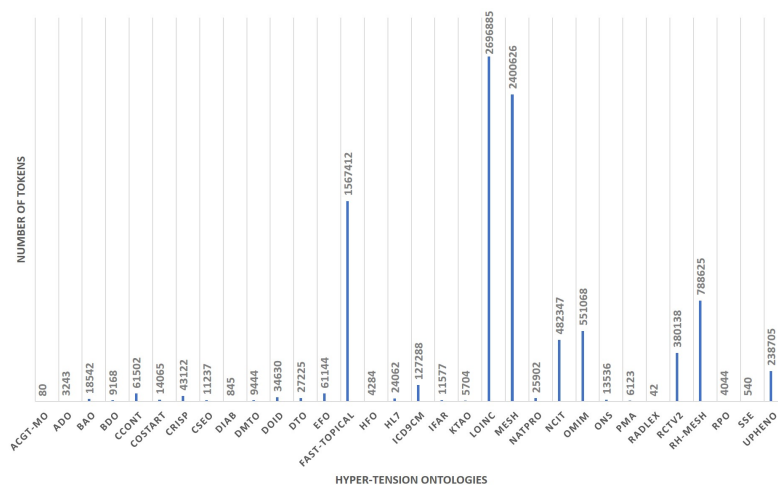


Figure 66: Tokens from Hypertension Ontologies

8.4.3 Results of Natural Language Processing

Figure 66 shows the number of tokens generated from the NLP processing with the ontological properties in hypertension ontologies dataset. An average of 300,724 tokens are extracted from 32 ontologies. The extracted tokens are at a maximum of 2,696,885 in LOINC ontology and at a minimum of 42 in RADLEX ontology. Top three ontologies with the largest tokens are LOINC, MESH, and FAST-TOPICAL. These tokens are used to generate the word embeddings.

8.4.4 Results of Word Embeddings

Figure 67 shows the word embeddings generated from ontology property corpus using Word2Vec with window size 20. In this embeddings, the first column in each row is the base word and the words from the second column to the 20th column are the words that share common contexts with the base word in the corpus.

1	AALPHA.	VLISINGE	TOCHIGI	MATSUMI	FLORXENII	PRDX1	ISGS6	IFNA14	MILANO	IFNAR2	SIMO	ALDIMINE	AF2	HPPA
2	ACCORD.	HEALEY	CHARGER	CORVETTE	HOLDEN	CRUISER	LEGEND	THUNDER	STUDEBAK	CADILLAC	CITROËN	AF	CIRRUS	RENAULT
3	ALLOILE.	ADAMANT	DLEU	PLACTIN	HISTOGRA	OET	DTRP	BZL	MEASP	MEARG	ARGININY	NHOH	ENKEPHAL	NHCH3
4	ANALYST.	TRACER	GENERATI	SELAGINEI	FOCK	VOYAGER	CHEMIST	MICROCEI	HOMA	ASCENT	ELISA	IMMUNOC	COCCIDIO	LYMPHAN
5	ANCYLOST	ENTOPTER	CYSTOISO:	EUROTIUM	PHAEOCHI	PROTHOR,	DENTARY	CALDICEI	EPIBOLY	BARTONEI	OPERCLE	RADICLE	KENNEDY	BRAZILIEI
6	ANNIVERS	UNBORN	CUSTARD	ARENAS	KAW	LIMINALIT	ALTIMETE	OILSEEDS	MASERATI	DACIANS	GRAYLING	WITCHES	VULNERAE	FILMSTRIP
7	APOD.	SYNEMBR	PARAXIS	NOELIN	HYAL2	HRK	MKRN2	BID3	RNF13	MTSSB	CHIBBY	FOX2	FOXH1	DRP1
8	ARTHROS	PERONOSI	PERIPLOC	NEUROPH	BARI	SCYPHOZC	PEARLS	TIGGER	RUFOUS	DICHOTOM	CHARACID	WALTON	CROTIN	BREWSTEF
9	AUREUS.M	MRSA	AUREUS.G	SP.OXACIL	HAEMOLY	SP.COAGU	SAU3AI	SEB	TOX1	SPECIES.V	AERUGINC	ETX	TSST	SUSCEPTIE
10	AVOCADO	CLAM	ENDIVE	SCALLOP	PERSIMM	PARSNIP	SWORDFIS	GINGER	CHICORY	ASPEN	NECTARIN	OREGANO	WATTLE	ALLSPICE
11	BARX2.	GSH2	GEC1	ANGPTL6	SEC6L1	PINCH1	DYNLT1	COTL1	DMRT2	PNAS4	WNTLESS	TSNAXIP1	ARHGAP1E	WBSCR14
12	BCL2.	E1B	ATHANOG	BCL	IGH	CCND1	MYC	TRANSCRI	Q21.3	NIP2	Q32	MYEOV	BAX	MAF
13	BIOSEQUE	TAUSCHII	PLASMABI	HYPCHO	CHORDO	TELEOPSIS	NISHIMUR	LNCAP	KARPAS	QUADRATI	PITUICYTC	UACC	SCLC	CAKI
14	BOTULISV	BOTULISM	FOODBOR	BRAWNY	MEGACOL	MICROCEI	MUCORMI	ENDOPHTI	ARTERIOI	NEURONI	KARYOME	PARAPHAF	MYXOEDE	HYPGAST
15	BRUCEI.	TRYPANOS	RHODESIE	CRUZI	GAMBIEN	PROCYCLI	VSG	RHODESIE	ESAG	CONGOLE	TRYPANOS	FURANOEI	PHYTASE	PA26
16	CEBPA.	NEFL	APTX	FGF23	PTCH	RPGR	ABCD1	MEN1	SFTPB	HEXB	TYROBP	CSTB	AHCY	AARS2
17	CEREBROS	BOREDON	NGBAKA	ALMSHOU	CHEETAH	HUANCA	BERBERS	DANES	PURIM	DAFLA	PALACES	NOSOLOG	ARCADES	JUKEBOXE
18	CLARITY.	FLD.SPUN	UNPREDIC	WOUND.B	ANNUL	EXHIBITED	WOUND.E	OXYMETR	TOPICALS	RECAPITU	LANGUAG	IRREVOCA	REINTUBA	REFRACTO
19	CLIC2.	CLIC3	KCNF1	KCNT2	CLCN4	TNK1	STYK1	PDIK1L	NIM1	YSK1	MARK4	GRK1	MAP4K5	CDKL1
20	CLONAZEF	ETHYLFLU	PENTOBAL	NORPROP	DESALKYLI	CHLORDIA	MDMA	EME	PROPOXYF	PHENIDAT	BZE	MDEA	METHAQU	NORTRAM

Figure 67: Word2Vec Embeddings

8.4.5 Results of Semantic Embeddings

The feature extraction techniques are applied to the 5 ontological properties (shown in Table 47) extracted from 32 ontologies. We have identified the top 10 features and also selected 500 significant features based on the weight of each feature. The window size for Word2Vec was defined as 20 using TF and TF-IDF Terms and the similarity measure is cosine similarity. Table 49 shows the results (top 10 features) from two feature extraction approaches: (i) Term Frequency (TF) and (ii) Term Frequency and Inverse Document Frequency (TF-IDF).

Considering 500 significant features, only 109 terms were found in TF-IDF/Word2Vec Model, while 363 terms were found in TF/Word2Vec Model. As the terms of the TF/Word2Vec Semantic Embeddings were higher than TF-IDF/Word2Vec Semantic Embeddings, the TF/Word2Vec Model is more appropriate than TF-IDF/Word2Vec for semantic embeddings.

Table 49: Semantic Embeddings: Integration of TF-IDF and Word2Vec

TF Terms	Word2Vec	TF-IDF Terms	Word2Vec
Protein	Compound, Peptide, Car- dioblast	Hexane	Heptane, Methylthiazole, Dimethoxybenzylidene
Time	Date, Day, Planned	Avocado	Clam, Endive, Scallop
Point	Timepoint, Endpoint, Titra- tion	Temafloracin	Troleandomycin, Roxithromycin, Talampicillin
Human	Mouse, Rat, Zebrafish	Analyst	Tracer, Generational, Chemist
Piano	Guitar, Choir, Cymbals	Prescribed	Refills, Presc, Medication
DNA	RNA, Glycolase, Methylase	Distracted	Shocked, Helpful, Across
Acid	Salt, Monosodium, Disodium	Nordeoxycorticosterone	Cholestatrien, Ketovitamin, Thiaes- tra
Food	Source, Plant, Mix	Hebrew	Aramaic, Tagalog, Konkani
Serum	Plasma, Test, Level	Torture	Novices, Umbanda, Polygamy
Breast	Oesophagus, Uretur, Pros- trate	Maculo	Blackheads, Herpetiform, Papulovesicular

Table 50: Ontology Mapping

Type	Original Mapping		OMF Mapping	
	#Ontologies	#Connectivity	#Ontologies	#Connectivity
Regular Connectivity	27	113	32	23029
Strong Connectivity	19	39	13	54
Extraction	OMF: TF-IDF		OMF: TF	
Regular Connectivity	29	367	32	23029

8.4.6 Results of Ontology Mapping

Table 50 shows the increase in the degree of connectivity among ontologies in the hypertension context. After the OMF mapping, we found the degree of connectivity among ontologies has been significantly increased from 23% to 98%. Specifically, ontology mapping with no connectivity was 16% in original mapping, while all 32 ontologies were connected to other ontologies in the OMF mapping. In the OMF mapping, the degree of connectivity for the full connectivity (i.e., 31 ontologies) was 68%.

Figure 68 shows the connectivity among ontologies in hypertension without and with OMF. As seen in the figure, most of the ontologies are fully connected to other ontologies in the OMF framework. Figure 69a shows the ontology mapping graph for the 27 ontologies, 113 concept connectivity among 27 ontologies, 23% degree of connectivity with the original ontology mapping. Figure 69b shows the ontology mapping graph for the 32 ontologies and 23029 concept connectivity among 32 ontologies, 98% degree of connectivity after applying the OMF framework (TF-Word2Vec). These results show that the OMF framework is very effective in increasing the ontology number and the degree of concept connectivity among ontologies.

Table 51: Ontology Mapping and Connectivity in Hypertension

Type Ontology	Original		OMF	
	#Con	#Con > 100	#Con	#Con > 100
ACGT-MO	9	0	30	0
ADO	9	0	30	2
BAO	13	5	31	0
BDO	9	0	31	9
CCONT	13	4	31	0
COSTART	8	6	30	9
CRISP	8	6	31	0
CSEO	10	2	30	0
DIAB	9	1	30	0
DMTO	0	0	31	0
DOID	7	2	31	0
DTO	8	4	31	8
EFO	13	4	31	9
FAST-TOPICAL	8	0	31	0
HFO	0	0	31	0
HL7	8	6	30	9
ICD9CM	8	6	31	0
IFAR	8	1	31	0
KTAO	14	3	31	9
LOINC	8	4	31	3
MESH	8	6	31	0
NATPRO	1	0	31	9
NCIT	1	0	30	9
OMIM	9	5	31	0
ONS	12	6	30	0
PMA	1	0	30	4
RADLEX	6	0	21	4
RCTV2	0	0	31	0
RH-MESH	0	0	31	0
RPO	0	0	31	10
SSE	8	1	30	0
UPHENO	10	4	31	0

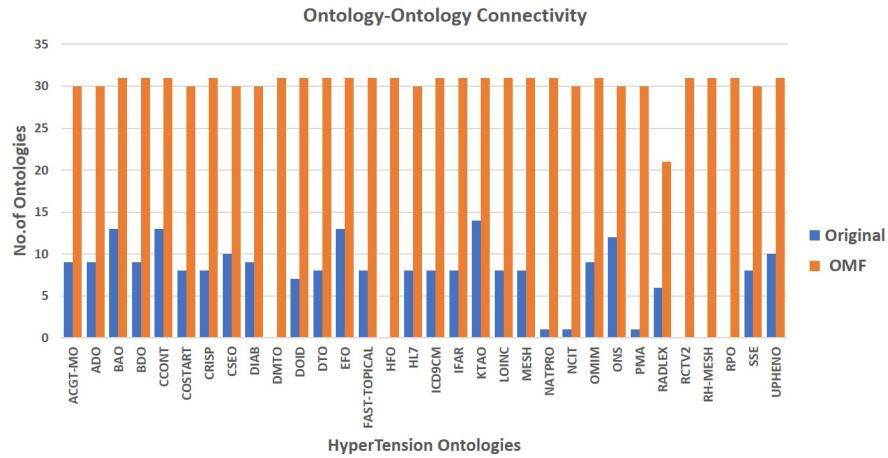


Figure 68: Ontology Connectivity in Hypertension

Table 52: Ontology Mapping (Source/Target) in Hypertension

Original			OMF		
Source	Target	#Con	Source	Target	#Con
EFO	CCONT	20636	RH-MESH	MESH	202
DOID	BAO	3210	EFO	CCONT	148
DTO	DOID	2089	OMIM	MESH	146
DTO	BAO	1770	RCTV2	MESH	145
UPHENO	CCONT	1767	MESH	LOINC	141
UPHENO	EFO	1756	MESH	CCONT	135
UPHENO	KTAO	1462	MESH	EFO	135
NCIT	IFAR	1048	MESH	FAST-TOPICAL	135
UPHENO	DIAB	359	RCTV2	LOINC	134
KTAO	CCONT	301	RCTV2	OMIM	132

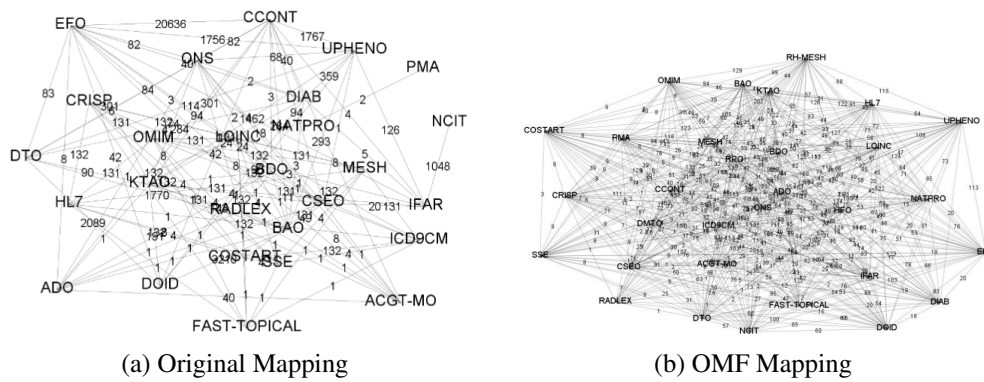


Figure 69: Ontology Mapping in Hypertension (Original Mapping vs. OMF Mapping)

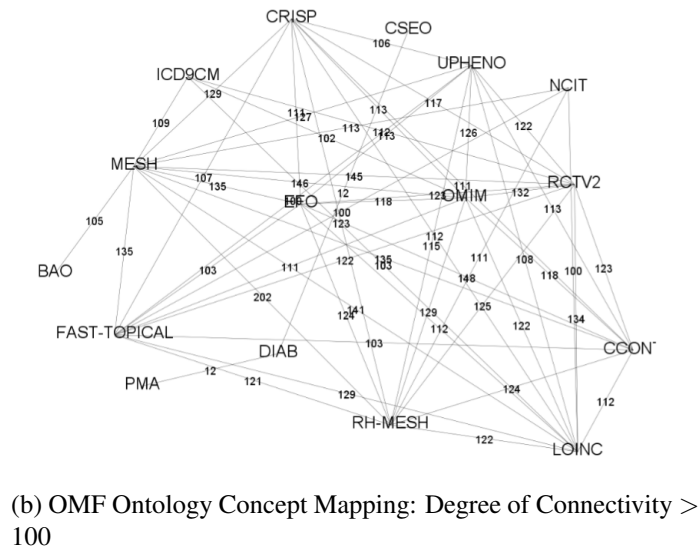
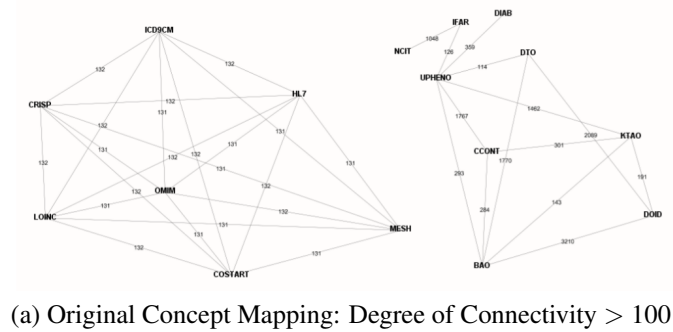
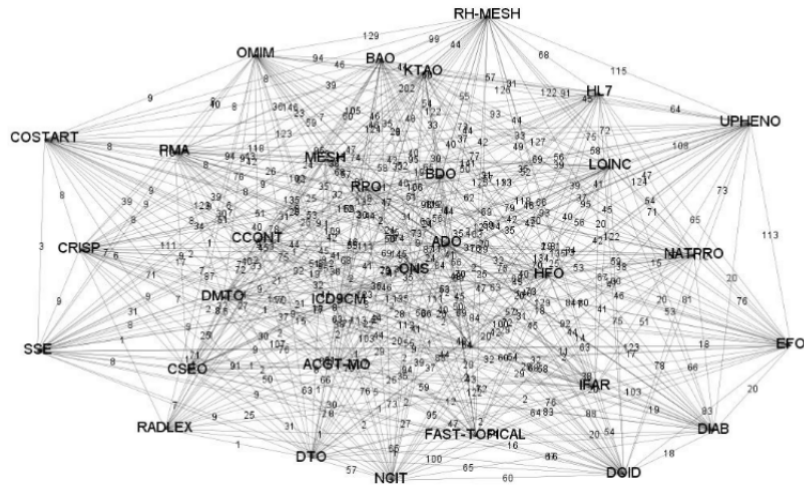
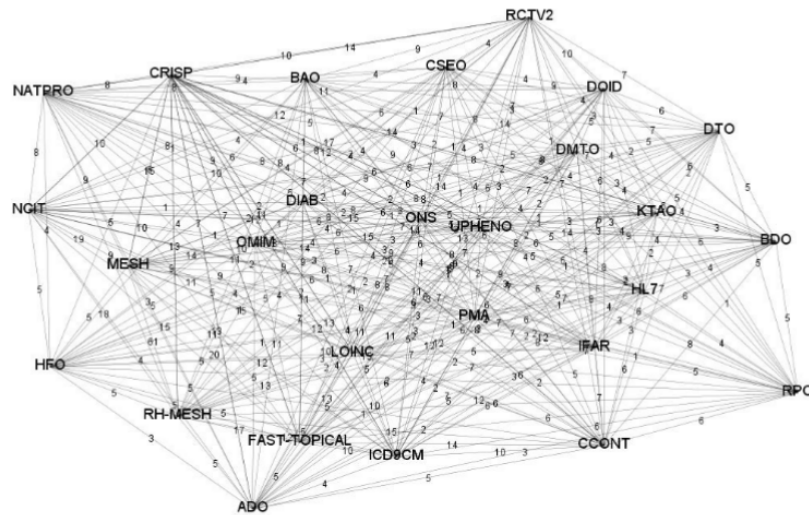


Figure 70: Ontology Mapping in Hypertension (Strong Degree of Connectivity: Original Mapping vs. OMF Mapping)



(a) OMF Ontology Mapping with TF-IDF Feature Extraction



(b) OMF Ontology Mapping with TF Feature Extraction

Figure 71: OMF with TF & TF-IDF Feature Extraction: Ontology Mapping in Hypertension

Figure 70a shows the ontology mapping graph with 19 ontologies, 39 concept connectivity, showing a strong relationship (the number of common concepts > 100) with the original ontology mapping. Figure 70b shows the ontology mapping graph with 13 ontologies, 54 concept connectivity showing a strong relationship (the number of common concepts > 100) after applying the OMF framework (TF-Word2Vec). These results show that the OMF framework effectively increases the degree of concept connectivity (for the strong connectivity of the number of common concepts > 100) among ontologies.

Figure 71a shows the ontology mapping graph with 32 ontologies and 367 concept connectivity showing strong relations (the number of common concepts > 100) after applying the OMF framework (TF-IDF-Word2Vec). Figure 71b shows the ontology mapping graph for the 29 ontologies and 23029 concept connectivity after applying the OMF framework (TF-Word2Vec). These results show that the TF feature extraction is more effective than the TF-IDF extraction method in the OMF framework.

Table 51 shows the original and OMF mappings for normal and strong connectivity relations between the concepts for the 32 ontologies. Table 52 shows the original and OMF mappings between source and target ontologies, together with the number of connectivity.

8.5 Conclusion

In this chapter, we proposed a semantic framework for automatic ontology mapping through ontology search, feature extraction, and word embeddings. This is a new

way to discover semantic mapping between concepts across multiple ontologies. The ontologies were mapped to semantic features extracted from multiple ontologies selected from the NCBI BioPortal [244]. From the comparative analysis, we confirmed that the proposed approach is effective in discovering relationships between ontologies.

The OMF framework was implemented in a parallel and distributed computing engine, Apache Spark, for providing a scalable solution for a large number of features from multiple ontologies. The parallel and distributed pipeline approach includes (i) Ontology searching using BioPortal API [237], (ii) Ontology query with the ontology properties using Apache Jena (iii) TF-IDF (Term Frequency Inverse Document Frequency) for feature extraction, (iv) Word2Vec for Word Embeddings, (v) Ontology mapping through concept matching, (vi) Gephi for Ontology mapping graph visualization. We confirmed that the OMF framework is effective in enhancing the existing ontologies mapping and discovering new relations across ontologies beyond the boundary of ontologies. The work presented in this chapter was published as part of Chandrashekar et al. [261]. Nagabhushan et al.[223] and Chandrashekar et al. [243] are some related publications.

CHAPTER 9

CONCLUSION

We proposed a new learning method, the deep open representative learning for image and text classification. The open representative learning consists of class representative for text and image data. Class Representatives are designed to project the abstract features extracted from a deep learning environment to the high-dimensional feature space. The contributions can be summarized into image and text categories.

9.1 Class Representative Learning for Image

- **Image Classification Setting:** As shown in Chapter 2, the class representatives were independently generated using pre-trained convolution networks for image classification. The CRL model showed slightly improved performance compared to existing mobile-based image classification with significant in execution time. The Class Representative. Such showing that the class representative learning model was an effective and efficient image classification model.
- **Zero-Shot Learning Setting:** As shown in Chapter 3, the class representative showed an effective model for seen data and unseen data. Compared with state-of-the-art zero-shot learning models, CRL proved to out-performed significantly for unseen data and at par in seen accuracy. The class representative model was powerful,

with very small and highly imbalanced datasets. In this chapter, the class representative's shows the ability to transfer knowledge from seen dataset to completely unseen dataset with no learning or relearning on the unseen dataset.

- **Discriminant Distribution Model:** As shown in Chapter 4, a discriminant distribution model was proposed using an optimal distribution of classes by computing a misclassification cost (i.e., confusion factor). The classification hierarchical deep neural network model was built by learning an optimal distribution of classes with a higher accuracy performance of the learning process.

9.2 Class Representative Learning for Text

- **Zero-Shot Learning Setting:** As shown in Chapter 5, the class representative learning was proposed for zero-shot learning text classification. The class representative learning was generated by aggregating projected features using sentence encoder and feed-forward networks. In this chapter, the class representative's shows the ability to transfer knowledge from seen text dataset to completely unseen text dataset with no learning or relearning on the unseen dataset. The class representative learning model was very effective in the text classification compared to state-of-art zero-shot learning text classification models.
- **Context Discovery:** In Chapter 6, the VisContext framework was proposed to capture visual context through text caption data using unsupervised approaches. The results confirm the effectiveness in discovering the contextual association of terms

and images, visual context clustering, and image classification based on context.

- **Text to Ontology:** In Chapter 7, a semantic framework was proposed for dynamically generating a diabetes publication ontology from a large corpus of diabetes publications. The topic modeling methods were employed in finding latent topics. These topic terms were mapped to ontological assertions extracted from the scientific publications in PubMed. The ontological assertions were used to enhance the existing diabetes ontologies, and new relations and entities between topics were introduced to existing diabetes ontologies. In Chapter 8, describes an automatic ontology mapping through ontology search, feature extraction, and word embeddings, a new way to discover semantic mapping between concepts across multiple ontologies. The ontologies were mapped to semantic features extracted from multiple ontologies selected from the NCBI BioPortal [244].

CHAPTER 10

FUTURE WORK

As future work, the class representative learning (CRL) model can be extended to represent multi-modal data, particularly by combining image and text class representatives of a particular class. It will be shown that the extension of the CRL model needs further enhancement for multi-label classification and even zero-shot object detection. The CRL-based zero-shot framework can be easily extended to incorporate multi-label text classification with the minimum modification in the CRL's inferring. It is similar to the object detection problem. The same success can be achievable with the change of zero-shot classification by extracting features from multiple objects instead of from entire images. By adding a region detection layer to the CRL based image classification model, we can create class representatives for object detection.

The CRL-based zero-shot learning shows the superior ability to transfer knowledge from a source domain to a target domain. CRL can be considered as an effective domain adaptation technique due to its superiority in performance, even if no learning is needed for a target domain. The existing domain adaptation methods rely on rich prior knowledge on the relationships of source and target domains. The bonds can be analyzed in different abstraction levels, such as class representatives (lowest abstraction within a single domain), category gaps (intra-domain relationships), and domain gaps (inter-domain relationships). Currently, CRL relies on a single source and multiple target

model. The CRL-based domain adaptation can be extended for open-set transfer learning for various sources and various targets. The various targets in the transfer learning in CRL will be supported for an open-set recognition framework.

We have described the ability of smaller models and contextual models in CRL in Chapter 2. This ability can be further explored for lightweight edge or mobile applications for specific contexts. The CRL framework will be extended for context-aware dynamic modeling.

We will apply the CRL model to a bio-medical domain. The CRL models for image and text can be integrated with medical ontologies. The integrated semantic model will contribute to creating a bio-medical application for clinical decision support systems. This integrated semantic model can be exploited for personalized detection and treatment, which is crucial to precision medicine.

Finally, the integrated semantic model with CRL (the CRL ontologies) will support the explainable artificial intelligence. Machine learning and ontologies can be interoperability (from ontologies to machine learning as well as machine learning to ontologies). The CRL ontologies will provide actionable interpretable models to applications. It will lead to reasoning and learning capabilities that will be available in the CRL ontologies.

APPENDIX A

A COMPARATIVE EVALUATION WITH DIFFERENT SIMILARITY MEASURES

In this appendix, we compare the four different similarity measures namely cosine similarity, Minkowski distance, Euclidean distance and Manhattan distance for evaluation of class representatives.

Cosine Similarity: Cosine Similarity is measured by the cosine of angle between two class representatives projected in N -dimensional space (Equation A.1).

Cosine Similarity :

$$\begin{aligned}\cos(CR_1, CR_2) &= \frac{CR_1 \cdot CR_2}{\|CR_1\| \|CR_2\|} \\ &= \frac{\sum_{i=1}^N x_{i,1}x_{i,2}}{\sqrt{\sum_{i=1}^N x_{i,1}^2} \sqrt{\sum_{i=1}^N x_{i,2}^2}}\end{aligned}\tag{A.1}$$

Euclidean Distance Euclidean Distance is the distance between two points in Cartesian coordinates in N -dimensional space (Equation A.2).

Euclidean Distance :

$$E(CR_1, CR_2) = \sqrt{\sum_{i=1}^N (x_{i,1} - x_{i,2})^2}\tag{A.2}$$

Manhattan Distance Manhattan Distance is the fixed distance between two points in Cartesian coordinates in N -dimensional space (Equation A.3).

Mahanttan Distance :

$$M(CR_1, CR_2) = \sum_{i=1}^N (x_{i,1} - x_{i,2}) \quad (\text{A.3})$$

Minkowski Distance The Minkowski distance is a metric in a normed vector space which can be considered as a generalization of both the Euclidean distance and the Manhattan distance.

Minkowski Distance of order of p :

$$\text{Mk}(CR_1, CR_2) = \left(\sum_{i=1}^N (x_{i,1} - x_{i,2})^p \right)^{\frac{1}{p}} \quad (\text{A.4})$$

Table 53 shows the comparison between Similarity Measures using class representatives from two different datasets: Caltech-101 and CIFAR100. Caltech-101 is one of the best performing and CIFAR100 is one of the worst-performing datasets. The class representative image classification task was considered for this evaluation.

Table 53: Comparison between Similarity Measures in CRL Image Classification Task

Similarity Measures	Caltech-101 Accuracy	CIFAR-100 Accuracy
Cosine Similarity	93.9	57.96
Euclidean Distance	93.0	57.07
Manhattan Distance	92.5	56.17
Minkowski Distance ($p = 0.5$)	92.3	54.09

Bibliography

- [1] Davide Castelvechi. “Can we open the black box of AI?” In: *Nature News* 538.7623 (2016), p. 20.
- [2] David Gunning. “Explainable artificial intelligence (xai)”. In: *Defense Advanced Research Projects Agency (DARPA), nd Web 2* (2017), p. 2.
- [3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. “Representation learning: A review and new perspectives”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828.
- [4] Wei Wang et al. “A survey of zero-shot learning: Settings, methods, and applications”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 10.2 (2019), p. 13.
- [5] Abhijit Bendale and Terrance Boulton. “Towards open world recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 1893–1902.
- [6] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. “Learning to detect unseen object classes by between-class attribute transfer”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 951–958.
- [7] Brenden Lake et al. “One shot learning of simple visual concepts”. In: *Proceedings of the Annual Meeting of the Cognitive Science Society*. Vol. 33. 33. 2011.

- [8] Oriol Vinyals et al. “Matching networks for one shot learning”. In: *Advances in neural information processing systems*. 2016, pp. 3630–3638.
- [9] Durk P Kingma et al. “Semi-supervised learning with deep generative models”. In: *Advances in neural information processing systems*. 2014, pp. 3581–3589.
- [10] P Kingma Diederik, Max Welling, et al. “Auto-encoding variational bayes”. In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2014.
- [11] Danilo Jimenez Rezende et al. “One-shot generalization in deep generative models”. In: *arXiv preprint arXiv:1603.05106* (2016).
- [12] Xiaofeng Xu, Ivor W Tsang, and Chuancai Liu. “Complementary Attributes: A New Clue to Zero-Shot Learning”. In: *IEEE transactions on cybernetics* (2019).
- [13] Zeynep Akata et al. “Multi-cue zero-shot learning with strong supervision”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 59–68.
- [14] Aravindh Mahendran and Andrea Vedaldi. “Understanding deep image representations by inverting them”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 5188–5196.
- [15] Bolei Zhou et al. “Learning deep features for discriminative localization”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2921–2929.
- [16] Sachin Ravi and Hugo Larochelle. “Optimization as a model for few-shot learning”. In: *International Conference on Learning Representations*. 2016.

- [17] Yunlong Yu, Zhongfei Zhang, and Jungong Han. “Meta-Transfer Networks for Zero-Shot Learning”. In: *arXiv preprint arXiv:1909.03360* (2019).
- [18] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.
- [19] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. “An analysis of deep neural network models for practical applications”. In: *arXiv preprint arXiv:1605.07678* (2016).
- [20] Mingxing Tan and Quoc V Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: *arXiv preprint arXiv:1905.11946* (2019).
- [21] Mark Sandler et al. “Mobilenetv2: Inverted residuals and linear bottlenecks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4510–4520.
- [22] Barret Zoph et al. “Learning transferable architectures for scalable image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 8697–8710.
- [23] Xiaojun Chang et al. “Dynamic concept composition for zero-example event detection”. In: *Thirtieth AAAI Conference on Artificial Intelligence*. 2016.
- [24] Mark Palatucci et al. “Zero-shot learning with semantic output codes”. In: *Advances in neural information processing systems*. 2009, pp. 1410–1418.
- [25] Zafer Aydin et al. “Learning sparse models for a dynamic Bayesian network classifier of protein secondary structure”. In: *BMC bioinformatics* 12.1 (2011), p. 154.

- [26] Y-Lan Boureau et al. “Learning mid-level features for recognition”. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Cite-seer. 2010, pp. 2559–2566.
- [27] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. “Improving the fisher kernel for large-scale image classification”. In: *European conference on computer vision*. Springer. 2010, pp. 143–156.
- [28] Weixin Li et al. “Dynamic pooling for complex event recognition”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2013, pp. 2728–2735.
- [29] Yangqing Jia, Chang Huang, and Trevor Darrell. “Beyond spatial pyramids: Receptive field learning for pooled image features”. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2012, pp. 3370–3377.
- [30] Weixin Li and Nuno Vasconcelos. “Recognizing activities by attribute dynamics”. In: *Advances in Neural Information Processing Systems*. 2012, pp. 1106–1114.
- [31] Weixin Li et al. “Recognizing activities via bag of words for attribute dynamics”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 2587–2594.
- [32] Eric Tzeng et al. “Simultaneous deep transfer across domains and tasks”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 4068–4076.
- [33] Simon Kornblith, Jonathon Shlens, and Quoc V Le. “Do better imagenet models transfer better?” In: *arXiv preprint arXiv:1805.08974* (2018).

- [34] Kaiming He, Ross Girshick, and Piotr Dollár. “Rethinking imagenet pre-training”. In: *arXiv preprint arXiv:1811.08883* (2018).
- [35] Deepak Pathak et al. “Learning features by watching objects move”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 2701–2710.
- [36] Jeff Donahue et al. “Decaf: A deep convolutional activation feature for generic visual recognition”. In: *International conference on machine learning*. 2014, pp. 647–655.
- [37] Sinno Jialin Pan, Qiang Yang, et al. “A survey on transfer learning”. In: *IEEE Transactions on knowledge and data engineering* 22.10 (2010), pp. 1345–1359.
- [38] Iasonas Kokkinos. “Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6129–6138.
- [39] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. “Learning multiple visual domains with residual adapters”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 506–516.
- [40] Hakan Bilen and Andrea Vedaldi. “Universal representations: The missing link between faces, text, planktons, and cat breeds”. In: *arXiv preprint arXiv:1701.07275* (2017).
- [41] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. “Efficient parametrization of multi-domain deep neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8119–8127.

- [42] Ashish Kapoor et al. “Learning to learn: Algorithmic inspirations from human problem solving”. In: *Twenty-Sixth AAAI Conference on Artificial Intelligence*. 2012.
- [43] Li Shen et al. “Multi-level discriminative dictionary learning with application to large scale image classification”. In: *IEEE Transactions on Image Processing* 24.10 (2015), pp. 3109–3123.
- [44] Kaiming Nan et al. “Deep model compression for mobile platforms: A survey”. In: *Tsinghua Science and Technology* 24.6 (2019), pp. 677–693.
- [45] Andrew G Howard et al. “Mobilenets: Efficient convolutional neural networks for mobile vision applications”. In: *arXiv preprint arXiv:1704.04861* (2017).
- [46] Xiangyu Zhang et al. “Shufflenet: An extremely efficient convolutional neural network for mobile devices”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6848–6856.
- [47] Jake Snell, Kevin Swersky, and Richard Zemel. “Prototypical networks for few-shot learning”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 4077–4087.
- [48] Wenlin Wang et al. “Zero-shot learning via class-conditioned deep generative models”. In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.
- [49] Lior Wolf and Ian Martin. “Robust boosting for learning from few examples”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. IEEE. 2005, pp. 359–364.

- [50] Antonio Torralba, Kevin P Murphy, and William T Freeman. “Sharing visual features for multiclass and multiview object detection”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 5 (2007), pp. 854–869.
- [51] Francois Fleuret and Gilles Blanchard. “Pattern recognition from one example by chopping”. In: *Advances in Neural Information Processing Systems*. 2006, pp. 371–378.
- [52] Evgeniy Bart and Shimon Ullman. “Cross-generalization: Learning novel classes from a single example by feature replacement”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. IEEE. 2005, pp. 672–679.
- [53] Yonatan Amit et al. “Uncovering shared structures in multiclass classification”. In: *Proceedings of the 24th international conference on Machine learning*. ACM. 2007, pp. 17–24.
- [54] Tomas Mikolov et al. “Distributed representations of words and phrases and their compositionality”. In: *Advances in neural information processing systems*. 2013, pp. 3111–3119.
- [55] Jeffrey Pennington, Richard Socher, and Christopher Manning. “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.
- [56] Zeynep Akata et al. “Evaluation of output embeddings for fine-grained image classification”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 2927–2936.

- [57] Andrea Frome et al. “Devise: A deep visual-semantic embedding model”. In: *Advances in neural information processing systems*. 2013, pp. 2121–2129.
- [58] Zhenyong Fu et al. “Zero-Shot Object Recognition by Semantic Manifold Distance”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015.
- [59] Bernardino Romera-Paredes and Philip Torr. “An embarrassingly simple approach to zero-shot learning”. In: *International Conference on Machine Learning*. 2015, pp. 2152–2161.
- [60] Yongqin Xian et al. “Latent embeddings for zero-shot classification”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 69–77.
- [61] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. “Attribute-based classification for zero-shot visual object categorization”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.3 (2013), pp. 453–465.
- [62] Mohammad Norouzi et al. “Zero-shot learning by convex combination of semantic embeddings”. In: *arXiv preprint arXiv:1312.5650* (2013).
- [63] Qian Wang and Ke Chen. “Alternative semantic representations for zero-shot human action recognition”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2017, pp. 87–102.
- [64] Qian Wang, Penghui Bu, and Toby P Breckon. “Unifying Unsupervised Domain Adaptation and Zero-Shot Visual Recognition”. In: *arXiv preprint arXiv:1903.10601* (2019).

- [65] Yi Zhu et al. “Towards universal representation for unseen action recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 9436–9445.
- [66] Youssef Tamaazousti et al. “Learning more universal representations for transfer-learning”. In: *IEEE transactions on pattern analysis and machine intelligence* (2019).
- [67] Olga Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.
- [68] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press, 2016.
- [69] Yanwei Fu et al. “Vocabulary-informed Zero-shot and Open-set Learning”. In: *IEEE transactions on pattern analysis and machine intelligence* (2019).
- [70] Yanwei Fu and Leonid Sigal. “Semi-supervised vocabulary-informed learning”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 5337–5346.
- [71] Rajat Raina et al. “Self-taught learning: transfer learning from unlabeled data”. In: *Proceedings of the 24th international conference on Machine learning*. ACM. 2007, pp. 759–766.
- [72] Ruochun Jin et al. “Confusion Graph: Detecting Confusion Communities in Large Scale Image Classification.” In: *IJCAI*. 2017, pp. 1980–1986.
- [73] Chi Jin et al. “A community detection approach to cleaning extremely large face database”. In: *Computational intelligence and neuroscience 2018* (2018).

- [74] Yuntao Liu et al. “Visual Tree Convolutional Neural Network in Image Classification”. In: *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE. 2018, pp. 758–763.
- [75] Mayanka Chandrashekar and Yugyung Lee. “MCDD: Multi-class Distribution Model for Large Scale Classification”. In: *2018 IEEE International Conference on Big Data (Big Data)*. IEEE. 2018, pp. 4906–4914.
- [76] Christian Szegedy et al. “Rethinking the inception architecture for computer vision”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2818–2826.
- [77] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.
- [78] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).
- [79] Apache Software Foundation. *Apache Spark*. Version 2.4.3. (Accessed on 05/26/2019). URL: <https://spark.apache.org/>.
- [80] *Pretrained Inception-v3 convolutional neural network - MATLAB inceptionv3 - MathWorks India*. <https://in.mathworks.com/help/deeplearning/ref/inceptionv3.html>. (Accessed on 05/13/2019).
- [81] Oscar Day and Taghi M Khoshgoftaar. “A survey on heterogeneous transfer learning”. In: *Journal of Big Data* 4.1 (2017), p. 29.
- [82] Laurens van der Maaten and Geoffrey Hinton. “Visualizing data using t-SNE”. In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605.

- [83] Sergey Zagoruyko and Nikos Komodakis. “Wide residual networks”. In: *arXiv preprint arXiv:1605.07146* (2016).
- [84] Youssef Tamaazousti, Hervé Le Borgne, and Céline Hudelot. “Mucale-net: Multi categorical-level networks to generate more discriminating features”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6711–6720.
- [85] Anish Shah et al. “Deep residual networks with exponential linear unit”. In: *arXiv preprint arXiv:1604.04112* (2016).
- [86] N. Silberman and S. Guadarrama. *TensorFlow-Slim image classification model library*. <https://github.com/tensorflow/models/tree/master/research/slim>. (Accessed on 08/19/2019). 2016.
- [87] *Pretrained Deep Neural Networks - MATLAB & Simulink - MathWorks India*. <https://in.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html>. (Accessed on 05/13/2019).
- [88] Walter J Scheirer et al. “Toward open set recognition”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.7 (2012), pp. 1757–1772.
- [89] Yongqin Xian, Bernt Schiele, and Zeynep Akata. “Zero-shot learning-the good, the bad and the ugly”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 4582–4591.
- [90] Mayanka Chandrashekar and Yugyung Lee. “CRL: Class Representative Learning for Image Classification”. In: *arXiv preprint arXiv:2002.06619* (2020).
- [91] Zeynep Akata et al. “Label-embedding for image classification”. In: *IEEE transactions on pattern analysis and machine intelligence* 38.7 (2015), pp. 1425–1438.

- [92] Yongqin Xian et al. “Zero-shot learning A comprehensive evaluation of the good, the bad and the ugly”. In: *IEEE transactions on pattern analysis and machine intelligence* 41.9 (2018), pp. 2251–2265.
- [93] Ilkay Ulusoy and Christopher M Bishop. “Comparison of generative and discriminative techniques for object detection and classification”. In: *Toward Category-Level Object Recognition*. Springer, 2006, pp. 173–195.
- [94] Soravit Changpinyo, Wei-Lun Chao, and Fei Sha. “Predicting visual exemplars of unseen classes for zero-shot learning”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 3476–3485.
- [95] Xingxing Zhang et al. “Hierarchical Prototype Learning for Zero-Shot Recognition”. In: *IEEE Transactions on Multimedia* (2019).
- [96] Zhenyong Fu et al. “Zero-shot learning on semantic class prototype graph”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.8 (2017), pp. 2009–2022.
- [97] Ali Farhadi et al. “Describing objects by their attributes”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 1778–1785.
- [98] Richard Socher et al. “Zero-shot learning through cross-modal transfer”. In: *Advances in neural information processing systems*. 2013, pp. 935–943.
- [99] Elyor Kodirov, Tao Xiang, and Shaogang Gong. “Semantic autoencoder for zero-shot learning”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 3174–3183.

- [100] Li Zhang, Tao Xiang, and Shaogang Gong. “Learning a deep embedding model for zero-shot learning”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 2021–2030.
- [101] Yongqin Xian et al. “Feature generating networks for zero-shot learning”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 5542–5551.
- [102] Edgar Schonfeld et al. “Generalized zero-and few-shot learning via aligned variational autoencoders”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8247–8255.
- [103] Rafael Felix et al. “Multi-modal cycle-consistent generalized zero-shot learning”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 21–37.
- [104] Jiechao Guan et al. “Zero and few shot learning with semantic feature synthesis and competitive learning”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).
- [105] Ziming Zhang and Venkatesh Saligrama. “Zero-shot learning via semantic similarity embedding”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 4166–4174.
- [106] Soravit Changpinyo et al. “Synthesized classifiers for zero-shot learning”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 5327–5336.

- [107] Long Chen et al. “Zero-shot visual recognition using semantics-preserving adversarial embedding networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1043–1052.
- [108] Zeynep Akata et al. “Label-embedding for attribute-based classification”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 819–826.
- [109] Jimmy Lei Ba, Kevin Swersky, Sanja Fidler, et al. “Predicting deep zero-shot convolutional neural networks using textual descriptions”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 4247–4255.
- [110] Majid Yazdani and James Henderson. “A model of zero-shot learning of spoken language understanding”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2015, pp. 244–249.
- [111] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. “Exploiting similarities among languages for machine translation”. In: *arXiv preprint arXiv:1309.4168* (2013).
- [112] Yutaro Shigeto et al. “Ridge regression, hubness, and zero-shot learning”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2015, pp. 135–151.
- [113] Yang Long et al. “Zero-shot learning using synthesised unseen visual data with diffusion regularisation”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.10 (2017), pp. 2498–2512.
- [114] Xun Xu, Timothy Hospedales, and Shaogang Gong. “Transductive zero-shot action recognition by word-vector embedding”. In: *International Journal of Computer Vision* 123.3 (2017), pp. 309–333.

- [115] Elyor Kodirov et al. “Unsupervised domain adaptation for zero-shot learning”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2452–2460.
- [116] Yanwei Fu et al. “Learning multimodal latent attributes”. In: *IEEE transactions on pattern analysis and machine intelligence* 36.2 (2013), pp. 303–316.
- [117] Qian Wang and Ke Chen. “Zero-shot visual recognition via bidirectional latent embedding”. In: *International Journal of Computer Vision* 124.3 (2017), pp. 356–383.
- [118] Baohan Xu et al. “Video emotion recognition with transferred deep feature encodings”. In: *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*. 2016, pp. 15–22.
- [119] Xing Xu et al. “Matrix tri-factorization with manifold regularizations for zero-shot learning”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 3798–3807.
- [120] Shai Ben-David et al. “Analysis of representations for domain adaptation”. In: *Advances in neural information processing systems*. 2007, pp. 137–144.
- [121] John Blitzer et al. “Learning bounds for domain adaptation”. In: *Advances in neural information processing systems*. 2008, pp. 129–136.
- [122] Daniel Kifer, Shai Ben-David, and Johannes Gehrke. “Detecting change in data streams”. In: *VLDB*. Vol. 4. Toronto, Canada. 2004, pp. 180–191.

- [123] Li Fei-Fei, Rob Fergus, and Pietro Perona. “Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories”. In: *2004 conference on computer vision and pattern recognition workshop*. IEEE. 2004, pp. 178–178.
- [124] Gregory Griffin, Alex Holub, and Pietro Perona. “Caltech-256 object category dataset”. In: (2007).
- [125] P. Welinder et al. *Caltech-UCSD Birds 200*. Tech. rep. CNS-TR-2010-001. California Institute of Technology, 2010.
- [126] Genevieve Patterson et al. “The SUN Attribute Database: Beyond Categories for Deeper Scene Understanding”. In: *International Journal of Computer Vision* 108.1-2 (2014), pp. 59–81.
- [127] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.
- [128] Mayanka Chandrashekar and Yugyung Lee. “Class Representatives for Zero-shot Learning using Purely Visual Data”. In: *IEEE transactions on pattern analysis and machine intelligence (submitted)* (2020).
- [129] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.
- [130] Christian Szegedy et al. “Inception-v4, inception-resnet and the impact of residual connections on learning.” In: *AAAI*. Vol. 4. 2017, p. 12.

- [131] Cheng Ju, Aurélien Bibaut, and Mark van der Laan. “The relative performance of ensemble methods with deep convolutional neural networks for image classification”. In: *Journal of Applied Statistics* (2018), pp. 1–19.
- [132] Andy Liaw, Matthew Wiener, et al. “Classification and regression by randomForest”. In: *R news* 2.3 (2002), pp. 18–22.
- [133] Richard Maclin and David Opitz. “An empirical evaluation of bagging and boosting”. In: *AAAI/IAAI 1997* (1997), pp. 546–551.
- [134] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. “Cifar-10 and cifar-100 datasets”. In: *URL: <https://www.cs.toronto.edu/kriz/cifar.html>* 6 (2009).
- [135] Wonjoon Goo et al. “Taxonomy-regularized semantic deep convolutional neural networks”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 86–101.
- [136] Mayank Kabra, Alice Robie, and Kristin Branson. “Understanding classifier errors by examining influential neighbors”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3917–3925.
- [137] Matthew D Zeiler and Rob Fergus. “Visualizing and understanding convolutional networks”. In: *European conference on computer vision*. Springer. 2014, pp. 818–833.
- [138] Carl Vondrick et al. “Hoggles: Visualizing object detection features”. In: *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE. 2013, pp. 1–8.

- [139] Jack W Stokes, Ashish Kapoor, and Debajyoti Ray. “Asking for a second opinion: Re-querying of noisy multi-class labels”. In: *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE. 2016, pp. 2329–2333.
- [140] Ruslan Salakhutdinov, Antonio Torralba, and Josh Tenenbaum. “Learning to share visual appearance for multiclass object detection”. In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE. 2011, pp. 1481–1488.
- [141] Ofer Dekel, Joseph Keshet, and Yoram Singer. “Large margin hierarchical classification”. In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, p. 27.
- [142] Siddharth Gopal et al. “Bayesian models for large-scale hierarchical classification”. In: *Advances in Neural Information Processing Systems*. 2012, pp. 2411–2419.
- [143] Denny Zhou, Lin Xiao, and Mingrui Wu. “Hierarchical classification via orthogonal transfer”. In: (2011).
- [144] Kristen Grauman, Fei Sha, and Sung Ju Hwang. “Learning a tree of metrics with disjoint visual features”. In: *Advances in neural information processing systems*. 2011, pp. 621–629.
- [145] Jia Deng et al. “What does classifying more than 10,000 image categories tell us?” In: *European conference on computer vision*. Springer. 2010, pp. 71–84.
- [146] Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. “Incremental algorithms for hierarchical classification”. In: *Journal of Machine Learning Research* 7.Jan (2006), pp. 31–54.

- [147] Bin Zhao, Fei Li, and Eric P Xing. “Large-scale category structure aware image categorization”. In: *Advances in Neural Information Processing Systems*. 2011, pp. 1251–1259.
- [148] David Warde-Farley, Andrew Rabinovich, and Dragomir Anguelov. “Self-informed neural network structure learning”. In: *arXiv preprint arXiv:1412.6563* (2014).
- [149] Zhicheng Yan et al. “HD-CNN: hierarchical deep convolutional neural networks for large scale visual recognition”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2740–2748.
- [150] Samy Bengio, Jason Weston, and David Grangier. “Label embedding trees for large multi-class tasks”. In: *Advances in Neural Information Processing Systems*. 2010, pp. 163–171.
- [151] Gregory Griffin and Pietro Perona. “Learning and using taxonomies for fast visual categorization”. In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE. 2008, pp. 1–8.
- [152] Baoyuan Liu et al. “Probabilistic label trees for efficient large scale image classification”. In: *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE. 2013, pp. 843–850.
- [153] Tianshi Gao and Daphne Koller. “Discriminative learning of relaxed hierarchy for large-scale visual recognition”. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE. 2011, pp. 2072–2079.
- [154] Peixiang Dong et al. “Training inter-related classifiers for automatic image classification and annotation”. In: *Pattern Recognition* 46.5 (2013), pp. 1382–1395.

- [155] Hao Lei et al. “Learning group-based dictionaries for discriminative image representation”. In: *Pattern Recognition* 47.2 (2014), pp. 899–913.
- [156] Marcin Marszałek and Cordelia Schmid. “Constructing category hierarchies for visual recognition”. In: *European Conference on Computer Vision*. Springer. 2008, pp. 479–491.
- [157] Ning Zhou and Jianping Fan. “Jointly learning visually correlated dictionaries for large-scale visual recognition applications”. In: *IEEE transactions on pattern analysis and machine intelligence* 36.4 (2014), pp. 715–730.
- [158] Jia Deng et al. “Fast and balanced: Efficient label tree learning for large scale object recognition”. In: *Advances in Neural Information Processing Systems*. 2011, pp. 567–575.
- [159] Jianping Fan et al. “Hierarchical learning of tree classifiers for large-scale plant species identification”. In: *IEEE Transactions on Image Processing* 24.11 (2015), pp. 4172–4184.
- [160] Juyong Kim et al. “SplitNet: Learning to semantically split deep networks for parameter reduction and model parallelization”. In: *International Conference on Machine Learning*. 2017, pp. 1866–1874.
- [161] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE. 2009, pp. 248–255.

- [162] Gang Wang, Ye Zhang, and Li Fei-Fei. “Using dependent regions for object categorization in a generative framework”. In: *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. Vol. 2. IEEE. 2006, pp. 1597–1604.
- [163] Shuai Zhao et al. “Real-time network anomaly detection system using machine learning”. In: *2015 11th International Conference on the Design of Reliable Communication Networks (DRCN)*. IEEE. 2015, pp. 267–270.
- [164] Prakash Vaka et al. “PEMAR: A pervasive middleware for activity recognition with smart phones”. In: *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. IEEE. 2015, pp. 409–414.
- [165] Wenpeng Yin, Jamaal Hay, and Dan Roth. “Benchmarking Zero-shot Text Classification: Datasets, Evaluation and Entailment Approach”. In: *arXiv preprint arXiv:1909.00161* (2019).
- [166] D Yogatama et al. “Generative and discriminative text classification with recurrent neural networks”. In: *Thirty-fourth International Conference on Machine Learning (ICML 2017)*. International Machine Learning Society. 2017.
- [167] Jingqing Zhang, Piyawat Lertvittayakumjorn, and Yike Guo. “Integrating semantic knowledge to tackle zero-shot text classification”. In: *arXiv preprint arXiv:1903.12626* (2019).

- [168] Anthony Rios and Ramakanth Kavuluru. “Few-shot and zero-shot multi-label learning for structured label spaces”. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*. Vol. 2018. NIH Public Access. 2018, p. 3132.
- [169] Pushpankar Kumar Pushp and Muktabh Mayank Srivastava. “Train once, test anywhere: Zero-shot learning for text classification”. In: *arXiv preprint arXiv:1712.05972* (2017).
- [170] Shashank Srivastava, Igor Labutov, and Tom Mitchell. “Zero-shot learning of classifiers from natural language quantification”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2018, pp. 306–316.
- [171] Yann N Dauphin et al. “Zero-shot learning for semantic utterance classification”. In: *arXiv preprint arXiv:1401.0509* (2013).
- [172] Congzheng Song et al. “Generalized Zero-shot ICD Coding”. In: *arXiv preprint arXiv:1909.13154* (2019).
- [173] Jinseok Nam, Eneldo Loza Mencía, and Johannes Fürnkranz. “All-in text: Learning document, label, and word representations jointly”. In: *Thirtieth AAAI Conference on Artificial Intelligence*. 2016.
- [174] Daniel Cer et al. “Universal sentence encoder”. In: *arXiv preprint arXiv:1803.11175* (2018).
- [175] Mohit Iyyer et al. “Deep unordered composition rivals syntactic methods for text classification”. In: *Proceedings of the 53rd Annual Meeting of the Association for*

Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). 2015, pp. 1681–1691.

- [176] Andrew L. Maas et al. “Learning Word Vectors for Sentiment Analysis”. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150. URL: <http://www.aclweb.org/anthology/P11-1015>.
- [177] Andrew L. Maas. *Large Movie Review Dataset*. <http://ai.stanford.edu/~amaas/data/sentiment/>. 2011 (accessed April 1, 2020).
- [178] Jens Lehmann et al. “DBpedia—a large-scale, multilingual knowledge base extracted from Wikipedia”. In: *Semantic Web 6.2* (2015), pp. 167–195.
- [179] Xiang Zhang and Yann LeCun. “Text understanding from scratch”. In: *arXiv preprint arXiv:1502.01710* (2015).
- [180] Tomas Mikolov et al. “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781* (2013).
- [181] Yoshua Bengio et al. “A neural probabilistic language model”. In: *Journal of machine learning research* 3.Feb (2003), pp. 1137–1155.
- [182] Prateek Veeranna Sappadla et al. “Using semantic similarity for multi-label zero-shot classification of text documents.” In: *ESANN*. 2016.
- [183] Mayanka Chandrashekar, Sai Sri Narne, and Yugyung Lee. “Sentence embeddings in NLI with iterative refinement encoders”. In: *Natural Language Engineering (submitted)* (2020).

- [184] Xiangrui Meng et al. “Mllib: Machine learning in apache spark”. In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 1235–1241.
- [185] Richard Socher and Li Fei-Fei. “Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora”. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, pp. 966–973.
- [186] Li-Jia Li, Richard Socher, and Li Fei-Fei. “Towards total scene understanding: Classification, annotation and segmentation in an automatic framework”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 2036–2043.
- [187] Sanja Fidler, Abhishek Sharma, and Raquel Urtasun. “A sentence is worth a thousand pixels”. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2013, pp. 1995–2002.
- [188] Micah Hodosh, Peter Young, and Julia Hockenmaier. “Framing image description as a ranking task: Data, models and evaluation metrics”. In: *Journal of Artificial Intelligence Research* 47 (2013), pp. 853–899.
- [189] Polina Kuznetsova et al. “Collective generation of natural image descriptions”. In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2012, pp. 359–368.
- [190] Ali Farhadi et al. “Every picture tells a story: Generating sentences from images”. In: *European conference on computer vision*. Springer. 2010, pp. 15–29.
- [191] Mark Yatskar et al. “See no evil, say no evil: Description generation from densely labeled images”. In: (2014).

- [192] Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. “Multimodal neural language models”. In: *International conference on machine learning*. 2014, pp. 595–603.
- [193] Jenny Rose Finkel, Christopher D Manning, and Andrew Y Ng. “Solving the problem of cascading errors: Approximate bayesian inference for linguistic annotation pipelines”. In: *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. 2006, pp. 618–626.
- [194] Andrej Karpathy and Li Fei-Fei. “Deep visual-semantic alignments for generating image descriptions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3128–3137.
- [195] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *European conference on computer vision*. Springer. 2014, pp. 740–755.
- [196] C Lawrence Zitnick, Ramakrishna Vedantam, and Devi Parikh. “Adopting abstract images for semantic scene understanding”. In: *IEEE transactions on pattern analysis and machine intelligence* 38.4 (2014), pp. 627–638.
- [197] Duyu Tang et al. “Learning sentiment-specific word embedding for twitter sentiment classification”. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2014, pp. 1555–1565.
- [198] Rémi Lebret, Joël Legrand, and Ronan Collobert. *Is deep learning really necessary for word embeddings?* Tech. rep. Idiap, 2013.
- [199] Jason Weston et al. “Connecting language and knowledge bases with embedding models for relation extraction”. In: *arXiv preprint arXiv:1307.7973* (2013).

- [200] Mo Yu and Mark Dredze. “Improving lexical embeddings with semantic knowledge”. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 2014, pp. 545–550.
- [201] Wengang Zhou et al. “Latent visual context learning for web image applications”. In: *Pattern Recognition* 44.10-11 (2011), pp. 2263–2273.
- [202] C Lawrence Zitnick and Devi Parikh. “Bringing semantics into focus using visual abstraction”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 3009–3016.
- [203] Akiko Aizawa. “An information-theoretic perspective of tf–idf measures”. In: *Information Processing & Management* 39.1 (2003), pp. 45–65.
- [204] Arthur P Dempster, Nan M Laird, and Donald B Rubin. “Maximum likelihood from incomplete data via the EM algorithm”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22.
- [205] Michael Grubinger et al. “The iapr tc-12 benchmark: A new evaluation resource for visual information systems”. In: *International workshop ontoImage*. Vol. 2. 2006.
- [206] Vicente Ordonez, Girish Kulkarni, and Tamara L Berg. “Im2text: Describing images using 1 million captioned photographs”. In: *Advances in neural information processing systems*. 2011, pp. 1143–1151.
- [207] Peter Young et al. “From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions”. In: *Transactions of the Association for Computational Linguistics* 2 (2014), pp. 67–78.

- [208] Mayanka Chandrashekar and Yugyung Lee. “Visual context learning with big data analytics”. In: *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*. IEEE. 2016, pp. 600–607.
- [209] David M Blei, Andrew Y Ng, and Michael I Jordan. “Latent dirichlet allocation”. In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.
- [210] PC Sherimon and Reshmy Krishnan. “OntoDiabetic: an ontology-based clinical decision support system for diabetic patients”. In: *Arabian Journal for Science and Engineering* 41.3 (2016), pp. 1145–1160.
- [211] Shaker El-Sappagh and Farman Ali. “DDO: a diabetes mellitus diagnosis ontology”. In: *Applied Informatics*. Vol. 3. 1. SpringerOpen. 2016, p. 5.
- [212] Shaker El-Sappagh et al. “DMTO: a realistic ontology for standard diabetes mellitus treatment”. In: *Journal of biomedical semantics* 9.1 (2018), p. 8.
- [213] P Cimiano and J Völker. “Text2Onto. Natural language processing and information systems”. In: *10th International Conference on Applications of Natural Language to Information Systems, NLDB*. 2005, pp. 15–17.
- [214] Paola Velardi, Alessandro Cucchiarelli, and Michael Petit. “A taxonomy learning method and its application to characterize a scientific web community”. In: *IEEE Transactions on Knowledge and Data Engineering* 19.2 (2007).
- [215] Diana Maynard, Adam Funk, and Wim Peters. “SPRAT: a tool for automatic semantic pattern-based ontology population”. In: *International conference for digital libraries and the semantic web, Trento, Italy*. 2009.

- [216] Juan Antonio Lossio-Ventura et al. “Towards an obesity-cancer knowledge base: Biomedical entity identification and relation detection”. In: *Bioinformatics and Biomedicine (BIBM), 2016 IEEE International Conference on*. IEEE. 2016, pp. 1081–1088.
- [217] Christopher Manning et al. “The Stanford CoreNLP natural language processing toolkit”. In: *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*. 2014, pp. 55–60.
- [218] Gabriel Stanovsky, Ido Dagan, et al. “Open IE as an intermediate structure for semantic tasks”. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Vol. 2. 2015, pp. 303–308.
- [219] Matthew Horridge and Sean Bechhofer. “The owl api: A java api for owl ontologies”. In: *Semantic Web 2.1 (2011)*, pp. 11–21.
- [220] Yu Lin and Yongqun He. “The ontology of genetic susceptibility factors (OGSF) and its application in modeling genetic susceptibility to vaccine adverse events”. In: *Journal of biomedical semantics* 5.1 (2014), p. 19.
- [221] Yu Lin and Norihiro Sakamoto. “Ontology driven modeling for the knowledge of genetic susceptibility to disease”. In: *Kobe J Med Sci* 55.3 (2009), E53–E66.
- [222] Drashti Vasant et al. “DIAB: an ontology of type 2 diabetes stages and associated phenotypes”. In: *Proceedings of Phenotype Day at ISMB 2015 (2015)*, pp. 24–27.
- [223] Megha Nagabhusan et al. “Constructing dynamic ontologies from biomedical publications”. In: *Bioinformatics and Biomedicine (BIBM), 2017 IEEE International Conference on*. IEEE. 2017, pp. 581–584.

- [224] Feichen Shen and Yugyung Lee. “BioBroker: Knowledge Discovery Framework for Heterogeneous Biomedical Ontologies and Data”. In: *Journal of Intelligent Learning Systems and Applications* 10.01 (2018), p. 1.
- [225] Mai Omura, Noboru Sonehara, and Takashi Okumura. “Practical approach for disease similarity calculation based on disease phenotype, etiology, and locational clues in disease names”. In: *Bioinformatics and Biomedicine (BIBM), 2016 IEEE International Conference on*. IEEE. 2016, pp. 1002–1009.
- [226] Shiwen Ma et al. “Similarity-based algorithms for disease terminology mapping”. In: *Bioinformatics and Biomedicine (BIBM), 2016 IEEE International Conference on*. IEEE. 2016, pp. 1378–1384.
- [227] Aldo Gangemi et al. “Semantic web machine reading with FRED”. In: *Semantic Web* 8.6 (2017), pp. 873–893.
- [228] Yuqun Zeng et al. “Recommending Education Materials for Diabetic Questions Using Information Retrieval Approaches”. In: *Journal of medical Internet research* 19.10 (2017).
- [229] Jonathan H Chen et al. “Predicting inpatient clinical order patterns with probabilistic topic models vs conventional order sets”. In: *Journal of the American Medical Informatics Association* 24.3 (2017), pp. 472–480.
- [230] Chong Wang and David M Blei. “Collaborative topic modeling for recommending scientific articles”. In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2011, pp. 448–456.

- [231] Lawrence WC Chan, SC Cesar Wong, and Keith WH Chiu. “Ontological features of electronic health records reveal distinct association patterns in liver cancer”. In: *Bioinformatics and Biomedicine (BIBM), 2016 IEEE International Conference on*. IEEE. 2016, pp. 1051–1053.
- [232] Michael Schmitz et al. “Open language learning for information extraction”. In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics. 2012, pp. 523–534.
- [233] Mitchell Marcus and RM Ann Taylor. “Alphabetical list of part-of-speech tags used in the penn treebank project”. In: (2012).
- [234] Marie-Catherine De Marneffe and Christopher D Manning. *Stanford typed dependencies manual*. Tech. rep. Technical report, Stanford University, 2008.
- [235] Jason Chuang, Christopher D Manning, and Jeffrey Heer. “Termite: Visualization techniques for assessing textual topic models”. In: *Proceedings of the international working conference on advanced visual interfaces*. ACM. 2012, pp. 74–77.
- [236] Carson Sievert and Kenneth Shirley. “LDavis: A method for visualizing and interpreting topics”. In: *Proceedings of the workshop on interactive language learning, visualization, and interfaces*. 2014, pp. 63–70.
- [237] *NCBO BioPortal Annotator API*. <https://bioportal.bioontology.org/annotator>.
- [238] *PubMed Central (PMC) APIs*. <https://www.ncbi.nlm.nih.gov/home/develop/api/>.

- [239] Matei Zaharia et al. “Spark: Cluster computing with working sets.” In: *HotCloud* 10.10-10 (2010), p. 95.
- [240] Steffen Lohmann et al. “WebVOWL: Web-based visualization of ontologies”. In: *International Conference on Knowledge Engineering and Knowledge Management*. Springer. 2014, pp. 154–158.
- [241] Natalya F Noy et al. “Creating semantic web contents with protege-2000”. In: *IEEE intelligent systems* 16.2 (2001), pp. 60–71.
- [242] Rohithkumar Nagulapati, Mayanka Chandrashekar, and Yugyung Lee. “Transformation from Publications to Diabetes Ontology using Topic-based Assertion Discovery”. In: *2018 IEEE International Conference on Healthcare Informatics Workshop (ICHI-W)*. IEEE. 2018, pp. 9–18.
- [243] Mayanka ChandraShekar and Joseph A Cottam. “Graph Generation with a Focusing Lexicon”. In: *2019 IEEE International Conference on Big Data (Big Data)*. IEEE. 2019, pp. 4928–4931.
- [244] Manuel Salvadores et al. “BioPortal as a dataset of linked biomedical ontologies and terminologies in RDF”. In: *Semantic web* 4.3 (2013), pp. 277–284.
- [245] Barry Smith et al. “The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration”. In: *Nature biotechnology* 25.11 (2007), p. 1251.
- [246] Gene Ontology Consortium. “Gene Ontology annotations and resources”. In: *Nucleic acids research* 41.D1 (2012), pp. D530–D535.
- [247] Daniel L Rubin, Nigam H Shah, and Natalya F Noy. “Biomedical ontologies: a functional perspective”. In: *Briefings in bioinformatics* 9.1 (2007), pp. 75–90.

- [248] Kathy Giannangelo and Susan H Fenton. “SNOMED CT survey: an assessment of implementation in EMR/EHR applications”. In: *Perspectives in Health Information Management/AHIMA, American Health Information Management Association* 5 (2008).
- [249] Feichen Shen and Yugyung Lee. “Knowledge discovery from biomedical ontologies in cross domains”. In: *PloS one* 11.8 (2016), e0160005.
- [250] Patrick Lambrix and He Tan. “SAMBO a system for aligning and merging biomedical ontologies”. In: *Web Semantics: Science, Services and Agents on the World Wide Web* 4.3 (2006), pp. 196–206.
- [251] Wei Hu, Yuzhong Qu, and Gong Cheng. “Matching large ontologies: A divide-and-conquer approach”. In: *Data & Knowledge Engineering* 67.1 (2008), pp. 140–160.
- [252] Michael Halper et al. “Abstraction networks for terminologies: supporting management of big knowledge”. In: *Artificial intelligence in medicine* 64.1 (2015), pp. 1–16.
- [253] Christopher Ochs et al. “Deriving an abstraction network to support quality assurance in OCRe”. In: *AMIA Annual Symposium Proceedings*. Vol. 2012. American Medical Informatics Association. 2012, p. 681.
- [254] Christopher Ochs et al. “Utilizing a structural meta-ontology for family-based quality assurance of the BioPortal ontologies”. In: *Journal of biomedical informatics* 61 (2016), pp. 63–76.

- [255] Jonathan M Mortensen et al. “Applications of ontology design patterns in biomedical ontologies”. In: *AMIA Annual Symposium Proceedings*. Vol. 2012. American Medical Informatics Association. 2012, p. 643.
- [256] Natalya F Noy, Nicholas Griffith, and Mark A Musen. “Collecting community-based mappings in an ontology repository”. In: *International Semantic Web Conference*. Springer. 2008, pp. 371–386.
- [257] Xin Rong. “word2vec parameter learning explained”. In: *arXiv preprint arXiv:1411.2738* (2014).
- [258] Apache Jena. “Apache jena”. In: *jena. apache. org [Online]*. Available: <http://jena.apache.org> [Accessed: Mar. 20, 2014] (2013), p. 14.
- [259] Sébastien Heymann. “Gephi”. In: *Encyclopedia of social network analysis and mining*. Springer, 2014, pp. 612–625. URL: <https://gephi.org/>.
- [260] W McKinney. “Python Data Analysis Libraryâpandas: Python Data Analysis Library [WWW Document]”. In: URL <http://pandas.pydata.org/>(accessed 7.22.15) (2015).
- [261] Mayanka Chandrashekar, Rohithkumar Nagulapati, and Yugyung Lee. “Ontology mapping framework with feature extraction and semantic embeddings”. In: *2018 IEEE International Conference on Healthcare Informatics Workshop (ICHI-W)*. IEEE. 2018, pp. 34–42.

VITA

Mayanka Chandrashekaris from a township called Anupuram, Tamil Nadu, India. She graduated from Women's Christian College, Chennai, Tamil Nadu, India (affiliated to University of Madras) with an Integrated Master in Computer Science degree in 2014. In August 2014, she moved to the USA to start her Interdisciplinary Ph.D. in Computer Science from the School of Computing and Engineering, University of Missouri-Kansas City, under the supervision of Dr. Yugyung Lee. Mayanka Chandrashekar's research interests include Semantic Web, Knowledge Discovery, Natural Language Processing, and Image Processing. She is currently leading two projects with her doctoral advisor, namely "Implicit Bias in STEM" and "Understanding Greek Literature using Deep Learning." Mayanka has received multiple awards, such as Google Lime Scholarship and SGS Research Award. She did a research internship in Pacific Northwest National Lab on the area of Natural Language Processing. She has published in multiple top conferences such as IEEE Big Data, ICDM, and PerCom. After completing her degree requirement, Mayanka Chandrashekar will start her Postdoctoral Research Associate in Oak Ridge National Lab in Oakridge, TN. She will be working in the area of Artificial Intelligence and Natural Language Processing for Biomedical Applications.