



学位論文題目 Title	Learning Complex and Continuous Affective Representations of Facial Expressions(表情における複雑と連続な感情表現の学習に関する研究)
氏名 Author	BAI, WENJUN
専攻分野 Degree	博士（工学）
学位授与の日付 Date of Degree	2019-03-25
公開日 Date of Publication	2020-03-01
資源タイプ Resource Type	Thesis or Dissertation / 学位論文
報告番号 Report Number	甲第7521号
権利 Rights	
JaLCDOI	
URL	http://www.lib.kobe-u.ac.jp/handle_kernel/D1007521

※当コンテンツは神戸大学の学術成果です。無断複製・不正使用等を禁じます。著作権法で認められている範囲内で、適切にご利用ください。

PDF issue: 2020-04-09

Doctoral Dissertation

Learning Complex and Continuous Affective Representations of Facial Expressions

表情における複雑と連続な感情表現の学習に関する研究

BAI WENJUN

January 2019

Graduate School of System Informatics
Kobe University

I would like to dedicate this thesis to my loving parents,
supervisors, tutors, beloved friends, and to ones who guide me
on the journey of scientific discovery.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 35,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 100 figures.

BAI WENJUN

January 2019

Acknowledgements

Prior to the delineation of the main content of this thesis, I would like to acknowledge several people that makes tremendous influences during my pursuit of Doctoral degree. I shall pay my chief gratitude towards my supervisors: Prof. Luo Zhi-Wei and Assoc. Prof. Quan Changqin, without their supports both mentally and academically, I would hardly go through these enduring academic years. I like to thank Prof. Luo Zhi-Wei who offers me the academic freedom in largest extent, and turning an arrogant, egocentric me into a still arrogant, egocentric person, who actually knows a bit about machine learning and affective computing. His rigorous criteria and relentless attention on computational science is my life-long pursuing model. To Assoc. Prof. Quan Changqin, with her patience and warm-hearted guidance, it guides me on the research pathway on affective computing.

Have literally no complains at all, my parents back me with their greatest kindness. Without their financial support, it is impossible for me to finish the doctoral study. I also want to thank my beloved Rachel Chux, without her unconditional support, I cannot fully focus on my studies. I would also like to appreciate the colleagues I work or worked with in this laboratory in creating a pleasant, thought-provoking working environment for me.

Last but not least, I would like to thank all the reviewers, especially Prof. Uehara Kuniaki and Prof. Yokokawa Mitsuo, who devote their time in perusing this thesis to perfect the current version.

Abstract

As a key aspect of nonverbal communication, facial expression plays an important role in social interaction between individuals. To improve the human-computer interaction, it urges a computer to arm with the ability to recognise, understand, and generate human facial expressions. To achieve this, it entails the development of a computational model to learn representations of facial expressions. However, to derive such model is a difficult task as most facial expressions are complex, and hard to be categorised. Previous efforts in tackling this issue focused on learning discrete single-label representations. However, these learned single-label representations of facial expressions are only capable of solving simply recognition tasks. More importantly, as these learned single-label representations are discrete, they contradict with the continuous nature of facial expressions. Hence, a computational model that is able to learn complex continuous representations of facial expressions is in demand.

To fill this research gap, the objective of this thesis is to learn complex and continuous affective representations of facial expressions. It targets on representations that are able to reflect the complexity of facial expressions in helping computers to comprehend and generate human-like facial expressions. To achieve this goal, three different researches, ranging from the usage of label relaxation technique in improving the discriminative performance of a neural network on a expression classification task, the proposal of a transfer learning paradigm to output multi-label predictions, to the proposal of the encapsulated variational auto-encoders (EVAE) to generate continuous expressions, are demonstrated in this thesis.

In our first approach, we aim to improve the performance of a neural network on an expression classification task via training the neural network with relaxed labels. The relied label relaxation technique transforms the original one hot encoded labels to real-numbered ones. As a running example, a joy-like expression, which is originally one-hot encoded as $[0, 0, 1]$, can now be relaxed to a continuous one, e.g., $[0.1, 0.1, 0.8]$. This allows the production of relaxed labels for supervised training of a neural network. Through the empirical result on FER2013 dataset, we show that compare to original one-hot labels, training a neural network with relaxed labels is able to improve its discriminative performance on a facial expression classification task.

However, the foregoing label relaxation technique only ensures the production of single-label representation of facial expression. In real world, a single expression can often be interpreted into multiple affects. E.g., it is not uncommon that a facial expression can be interpreted into joyfulness and surprise at the same time. Hence, it demands a model to output multiple labels to categorise a single facial expression. But to collect high quality multi-label annotations for supervised learning is labor intensive. For this reason, in this research, we propose a transfer learning based model to output multi-label annotations for a facial expression. The relied model is our proposed uncertainty flow framework. The proposed uncertainty flow framework consists of four components: two bayesian neural networks, the weakly informative priors, the transferred posterior weight distribution, and three multi-label prediction indexes. Arming with this framework, a complex facial expression that is only allowed to predict as $[0, 0, 1]$ (the joyfulness expression) before, can now be predicted as $[0, 1, 1]$ (the joyfulness and surprise expressions).

In our final research, the objective is to generate facial expressions along certain continuous axes. To achieve this, we propose a novel form of variational auto-encoder, i.e., encapsulated variational auto-encoders (EVAE). Within our proposed EVAE, we identify two continuous factors that exert the direct influence over the final generated expressions. Training this EVAE on two expression datasets, we show that EVAE is capable of generating facial expressions along these continuous factors.

This thesis – with its woven three proposed approaches – demonstrate the feasibility of learning complex continuous affective representations of facial expressions. It firmly believed that advancing this research path further will benefit computers in better recognising, comprehending and generating human facial expressions to improve the quality of human-computer interactions.

Table of contents

List of figures	xv
List of tables	xvii
Nomenclature	xix
1 General Introduction	1
1.1 Affective Computing	1
1.1.1 Aspects of Human Affection	1
1.1.2 Researches in Affective Computing	2
1.2 Automatic Facial Analysis	3
1.2.1 Early Facial Expression Analysis in Behaviour Psychology	4
1.2.2 Theoretical Framework of Automatic Facial Expression Analysis . .	5
1.3 Affective Representations of Facial Expressions	6
1.3.1 Taxonomy of Affective Representations	6
1.3.2 Issues	9
1.4 Priors for Representation Learning in Automatic Facial Analysis	11
1.5 Structure of This Thesis	12
2 A Principled Review on Selected DGMs in Representation Learning	15
2.1 Neural networks	17
2.1.1 NN: under the Representers' theorem	17
2.1.2 NN: as a DGM	21
2.2 Bayesian neural network	24
2.2.1 The Caveat of Overfitting in NN	24
2.2.2 The Description on BNN	24
2.2.3 Prior Choices	26
2.2.4 Posterior Inference	26
2.3 Variational Auto-Encoder	27

2.3.1	VAE: Coding Theory Interpretation	27
2.3.2	VAE: DGM Interpretation	30
2.3.3	Learning VAE	31
3	Learning Single-Label Relaxed Representations via Label Relaxation	35
3.1	Introduction	35
3.2	Methodology	36
3.2.1	Motivation of Adoption of Label Relaxation	36
3.2.2	From Gumbel Variable to Gumbel-Softmax Trick	39
3.2.3	Label Relaxation Technique	40
3.3	Empirical Experiment	41
3.4	Conclusion and Discussion	44
4	Learning Multi-label Discrete Representations via Transfer Learning	45
4.1	Introduction	45
4.2	Uncertainty Flow Framework	46
4.2.1	Two Bayesian Neural Networks	48
4.2.2	Weakly Informative Priors	48
4.2.3	Transferred Posterior Weight Distribution	49
4.2.4	Prediction Related Uncertainty Indexes	51
4.3	Empirical Experiment	52
4.3.1	Dataset	53
4.3.2	Models	54
4.3.3	Evaluation Metrics	56
4.3.4	Results & Discussion	58
4.4	Conclusion & Discussion	63
5	Generating Continuous Representations via EVAE	65
5.1	Introduction	65
5.2	Encapsulated Variational Auto-Encoders	66
5.2.1	Comparison between VAE and EVAE	66
5.2.2	Parameterisations on Encoders & Decoders	67
5.2.3	Hyper-parameter tuned latent representations	67
5.2.4	Learning objective	68
5.2.5	Learning algorithm	70
5.3	Continuous Expression Generation in EVAE	71
5.4	Empirical Validation	72

5.4.1	Experimental Set-ups	72
5.4.2	The optimised lower bound	74
5.4.3	Continuous generated expressions	74
5.5	Conclusion and Further Works	81
6	Discussion and Conclusion	83
6.1	Summary of This Thesis	83
6.2	Shortages of Our Approaches	84
6.2.1	The Micro View	84
6.2.2	The Macro View	85
6.3	Splendid Tomorrow	86
	Appendix A Notations and Rules in Probability and Inferential Statistics	93
A.1	Overall Workflow	93
A.2	Model Statistics	93
A.3	Useful Rules	94
A.4	Model Learning/Parameter Estimation	95
A.4.1	MLE	95
A.4.2	Bayesian parameter estimation	95
A.5	Model Critics	96
	Appendix B The General Learning Algorithm for Encapsulated Variational Auto-Encoders	97
B.0.1	Latent Variable Transformation and Elliptical Standardisation	97
B.0.2	Gradients w.r.t. Learning Objective	98
B.0.3	Pseudo Code for the algorithm	100

List of figures

1.1	Components of Affective Computing	2
1.2	Working pipeline of an affective computing system	3
1.3	Anatomy of facial muscle in human	4
1.4	Correspondence between AUs and facial emotions	5
1.5	Low- to High-level representations of facial expressions.	8
1.6	Demonstration of the ambiguity issue in FER2013 expression dataset. . . .	11
2.1	A sample DGM with latent variables.	16
2.2	A sample FNN: one hidden layer only example	19
2.3	A sample CNN: LeNet-5 model	21
2.4	NN under the DGM interpretation	22
2.5	Demo of a BNN	25
2.6	Taxonomy of Deep Generative Models	28
2.7	Encoder Theory interpretation of a VAE	29
2.8	Probabilistic interpretation of a VAE	32
2.9	Explanation on reparameterisation trick	33
3.1	Working pipeline of neural network training based on relaxed labels	38
3.2	Training in FER2013 dataset	43
4.1	Uncertainty Flow Framework	47
4.2	The Probabilistic Density Curves of Normal, Uniform, and Cauchy Priors .	50
4.3	Comparison of Three Prediction Related Uncertainty Indexes on a Binary Classifier	53
4.4	Model Wise Comparison on Multi-learning Metrics	60
4.5	Different priors on posterior weights under the uncertainty flow framework	61
4.6	Prior induced discriminability differences in the uncertainty flow framework	62
4.7	Discriminative Performance Across Different Prediction Related Uncertainty Indexes	63

5.1	Graphical models of (i) a conventional variational auto-encoder and (ii) our derived encapsulated variational auto-encoder. Solid arrows denote probabilistic decoders, whereas dash arrows represent variational encoders. . . .	66
5.2	The α bounded learning of EVAE on Frey faces dataset	74
5.3	The α bounded learning of EVAE on FERF-DB dataset	75
5.4	Facial expression generation along with the <i>valence</i> dimension on Frey Faces dataset	76
5.5	Facial expression generation along with the modelled <i>valence</i> dimension on FERF-DB dataset	77
5.6	Facial expression generation along with the modelled <i>arousal</i> dimension on Frey Faces dataset	79
5.7	Facial expression generation along with the modelled <i>arousal</i> dimension on FERF-DB dataset	80
A.1	The standard workflow of learning a probabilistic model	94

List of tables

3.1	Model Configurations.	42
3.2	Model-Wise Discriminative Performance in FER2013 dataset	42
4.1	Descriptive Statistics of training and testing dataset	54
4.2	Model Comparison in Various Multi-label Evaluation Metrics	59
5.1	Model architectures and training details on Frey-Face and FERG-DB datasets.	78

Nomenclature

Roman Symbols

Bernoulli Bernoulli Distribution

F Cost Function

f Mapping Function, Can be Linear or Non-linear Mapping

L Objective Function

\mathcal{N} Normal Distribution

p Density; Distribution of Certain Variable

q Variational Approximating Density

W Weight Matrix

w Weight Vector

X Input Variable; Can be either binary or continuous coded

x Data Instance

Y Output Variable(s); Can be either discrete or continuous

Z Latent Variables; Can be either continuous or discrete

Greek Symbols

α A Tuneable Hyper-parameter

β A Tuneable Hyper-parameter

θ Model Parameters

λ	A Tuneable Hyper-parameter
μ	Mean of Certain Density
ϕ	Variational Parameters
σ^2	Variance of Certain Density
τ	A Hyper-parameter that lives between 0 and 1
ω	Weights in Neural Network

Subscripts

n	Indicate the number of instances
-----	----------------------------------

Other Symbols

$*$	Convolution Operation
\sim	Sampling Process

Acronyms / Abbreviations

BNN	Bayesian Neural Network
CNN	Convolutional Neural Network
DGM	Directed Graphic Model
GAN	Generative Adversarial Net
MLP	Multi Layer Perceptron
NN	Neural Network
PGM	Probabilistic Graphic Model
ReLU	Rectified Linear Units
Tanh	Hyperbolic tangent function
VAE	Variational Auto-Encoder

Chapter 1

General Introduction

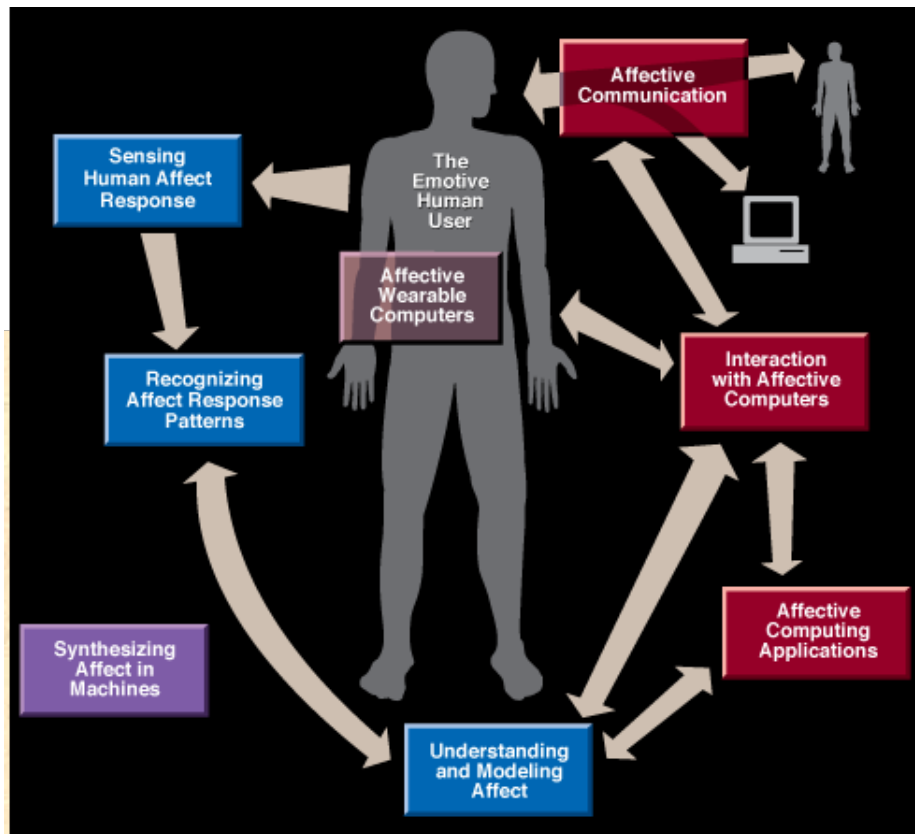
1.1 Affective Computing

The term: **Affective computing** was first coined by Rosalind Picard – the pioneer and foremost scholar in affective computing – refers computing that relates to, arises from, or deliberately influences emotion or other affective phenomena [1].

1.1.1 Aspects of Human Affection

Emotion, affection, sentire, and sentiment are used interchangeably to refer both our internal dynamics of emotional responses. The explanation of human emotion and the rise of emotional state and experience remains unfettered. Despite of the ferocious dispute, most theorists agree to view human emotion into two following aspects: the physical and cognitive ones [2]. The physical aspect of emotion pays overwhelming attention on the overt, explicit expression of a sentiment, such as voice inflection, our focused facial expression and postures. A more internal, implicit aspect of emotion is the cognitive aspect of emotion, which focus on the cognitive mechanism of an emotion.

Undoubtedly, from the computational perspective, the former aspect, i.e., the physical aspect of emotion is much convenient to access, quantify, and simulate, whereas the computation on the later cognitive aspect is a much challenging task [1]. I.e., even if a computer could perceive all the stimuli, how would it reach the interpretation as "love" and "dislike". Furthermore as a successful cognitive emotion model is likely to depend on individual's experience, it is nonetheless unlikely that every cognitive factor that influences emotion will be identified, collected, and recognised for a time-elapsd emotion response.



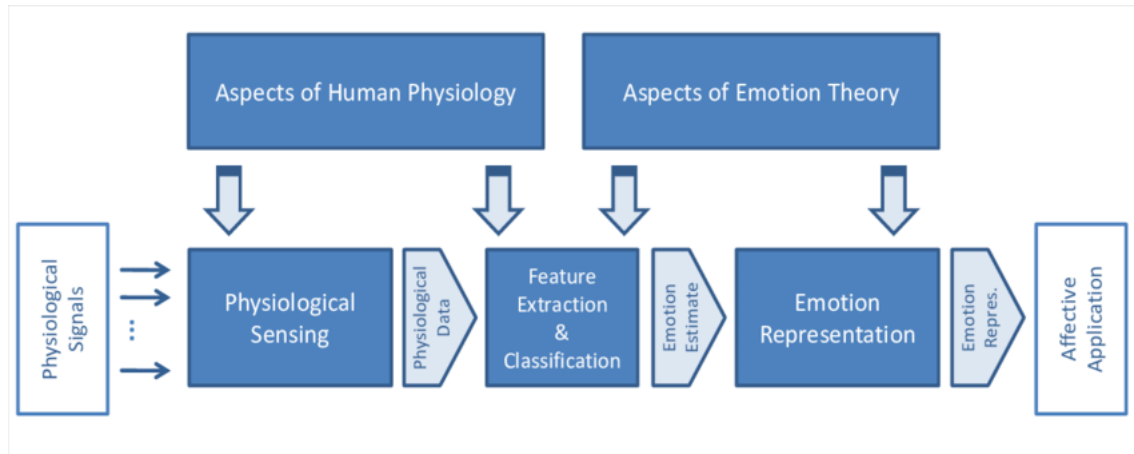
In Fig.1.1, it demonstrates the full spectrum of possible research areas in affective computing. The figure is original produced by MIT media lab, we thank Roseline Picard and her colleagues for their kind permission for redistribution.

Fig. 1.1 Components of Affective Computing

Hence, modern affective computing researches (including this one) are pertaining to the uniformed focus on analysis of the physical aspect of the emotion, leaving the exploitation of cognitive aspect to successors.

1.1.2 Researches in Affective Computing

Focusing on the physical aspect of emotion, as depicted in Fig.1.1, the full spectrum of affective computing researches ranges from the design of affective aware sensor hardware (*the marbled coloured block in Fig.1.1*), the computational analysis of human affects (*the blue coloured blocks in Fig.1.1*), to downstream applications (*the purple and red coloured blocks in Fig.1.1*). Coarsely speaking, the foregoing noted subareas of affective computing can be relating to the three-step process of expressing human sentiments as perception (sensing); modelling (analysis); and expression (application).



In Fig.1.2, it demonstrates three coarsely defined three subcomponents in a full-fledged affective computing system.

Fig. 1.2 Working pipeline of an affective computing system

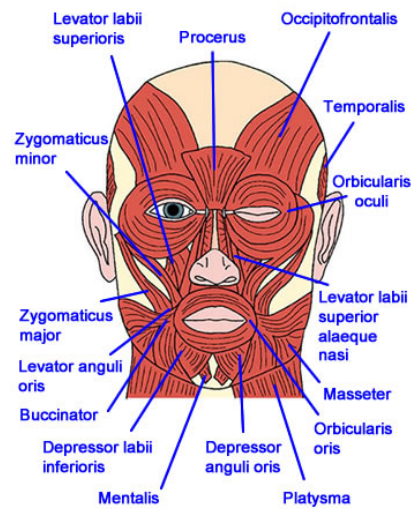
In affective computing, perception of an affect can be posited as a data acquisition or signal detection task in computer science. Emotion signal is carried by a broad range of multimodal cues, ranging from speech, image, gesture, vocal intonation, and et ac. This leads to diversified types of signals, including bio-potential signals, e.g., EEG and fMRI signals, , nonverbal vocalisation, bodily expression, speech and facial expression.

Posterior to the data acquisition stage, the collected signals are fed to the core analysing system to recognise the embedded affective patterns and features, then utilising these features in building effective applications for affective communication, recognition and so forth. This simplified working pipeline, as shown in the following thumbnail Fig.1.2, depicts the a standard procedure of building an affective computing system and three included subcomponents.

1.2 Automatic Facial Analysis

Among the prior noted diversified affective signals, facial expressions¹ has the pivotal status. Facial expressions – as one of foremost forms of nonverbal communication – conveys majority of affective information. Compare to other forms of affective signals, facial expressions are much easier to collect and obtain given their image or video based characteristic. Other forms of affective signals, such as voice intonation, demand intricate apparatus and devices

¹In this thesis, the terms: facial expressions, facial emotions and facial affects are used interchangeably. Moreover, from this section onwards, the terms: affects, emotions, and sentiments are all referred to the sentic modulations of the facial expressions.



In Fig.1.3, it renders the basic anatomical information of facial muscles. We adopt this hand-drawing rendering of the facial muscles from [7], the redistribution of this figure is only educational, non-profit purposes.

Fig. 1.3 Anatomy of facial muscle in human















to receive, and the release of large scale of such datasets, which are much less commonly seen in comparison to the ones of facial expressions.

To this end, we narrow our research focus on affective computing to the analysis on facial expression.

1.2.1 Early Facial Expression Analysis in Behaviour Psychology

The earliest documented analyses on facial expressions can be dated backed to late nineteenth century, Charles Darwin and Guillaume Duchenne demonstrated the uniqueness of human facial expression over other mammals [3], and the physiological relation of facial muscles and their corresponded facial expression through mild electric simulation [4]. Following the heir of the later approach, Ekman and Friesen [5] postulated six prototypic emotions, e.g., anger, disgust, happiness, sadness. and surprise. Their hypothesis was rationalised by the discrete emotion theory claiming that there is a merely small number of core emotions, and these emotions are largely innate. Continuing on their exploitation of basic emotions, and basing on the anatomical information of facial muscles that are laid in Fig.1.3, Ekman and his colleagues drafted a taxonomy system to analyse human facial movements. This system is commonly referred as Facial action unit system (FACS) [6].

The basic unit in FACS is action unit (AU), which is independent of any sorts of interpretations. Upholding the belief from Behaviourism [8] – it stipulates the fact that the percep-

Facial Muscles	Actions	AUs
Occipito Frontalis	Draw scalp forward and raises eyebrows	AU1  AU2 
Procerus	Pull the glabella down	AU4 
Orbicularis Oculi	Spread tears across cornea and close eyelid tightly	AU6  AU7 
Levator Labii Superioris Alaeque Nasi	Raise upper lip and wrinkles the nose	AU9  AU10 
Zygomaticus Major and Minor	Draw angle of mouth upward	AU11  AU12 
Depressor Anguli Oris	Draw angle of mouth downward	AU15 
Depressor Labii Inferioris	Lowers lower lip	AU16 
Mentalis	Draws chin up	AU17 
Orbicularis Oris	Levator/depressor of lip and angle of mouth	AU23  AU24 

In Fig.1.4, sample correspondences between AUs and facial expressions are their induced facial expressions.

Fig. 1.4 Correspondence between AUs and facial emotions

tions are the product of the combination of simple un-reducible components – it permits the simple arithmetic operations on AUs to define some of commonly seen facial expressions, some sample correspondences can be found in Fig.1.4.

1.2.2 Theoretical Framework of Automatic Facial Expression Analysis

The development of FACS also breeds the first stream of attempts in automating the analysis of human facial expressions in computer, accompanying with the release of large scale AU coded facial expression dataset, such as CK and CK+ [9] [10]. However, as FACS is a mere index of facial expressions, not conveying any bio-mechanical information about the degree of muscle activation, and as all AU codes need to be manually coded, which opens to the induced bias in coding, modern trends in automatic analysis of facial expression are gradually moving away from the adoption of AU computation to direct analysis on facial expressions in forms of static images or sequence of image frames (video).

Based on the modification of the framework from [11] (cf. Fig.2 in [11]), we offer our version of conceptual framework of facial expression analysis system: a two-stage process: **representation, and application**. Despite of the appeared superficial similarities, our

proposed framework differs from the previous ones from [11] [12] on the following two issues.

- **Scope of Application**

The primary one lies on the scope of the proposed theoretical framework. Two preceding frameworks restrict their usage of extracted features or representation onto a classification or regression task; whereas in our proposed theoretical framework, the applications are not limited to discriminative but also to the generative tasks, such as expression synthesis and multi-modal fusion. This extension largely widens the scope of our framework to cover the full spectrum of automatic facial analysis.

- **Scope of Representation Learning**

In both prior noted frameworks, the stage of representation learning is treated as a follow-up stage, which is placed right after registration or face acquisition step. This degrades the role of representation learning as a mere feature extraction process. Being agonist to this foregoing view, we propose the role of representation learning is ought to be more generic. I.e., the face registration and feature pooling stages should be seen as two sub-steps within the representation learning stage.

From the proposed theoretical framework of automatic facial expression analysis, it is lucid to reckon the pivotal status of learned representations from facial expressions. A good representation holds the key to ensure the content performance of an automatic facial analysis system on later tasks or downstream applications. **As a result, it brings up the main topic of this thesis: learning representations of human facial expressions.**

1.3 Affective Representations of Facial Expressions

1.3.1 Taxonomy of Affective Representations

Current existing taxonomies on learnable facial expression representations entails the specification of learned affective representations into spatial and spatial-temporal categories. Where the former only consider the static frame, as the later representations also include the temporal (time) signal. In this thesis, we mainly focus on the spatial representations, leaving the exploitation of spatial-temporal ones to upcoming researches.

Within the category of spatial representation, the rubric of finer specification remains contentious. Here, we adopt the genus that suggested in [11] to catalog the representations into three divisions: low-level and mid-to-high level, and hierarchical representations. In

short, low level information is encoded with low level histograms, Gabor representations, and other simple cell like edge and texture feature extractors. Higher level representations are more semantic interpretable, whereas the hierarchical representations cascade a series of low-to-high levels of representations together.

Low-Level Representations

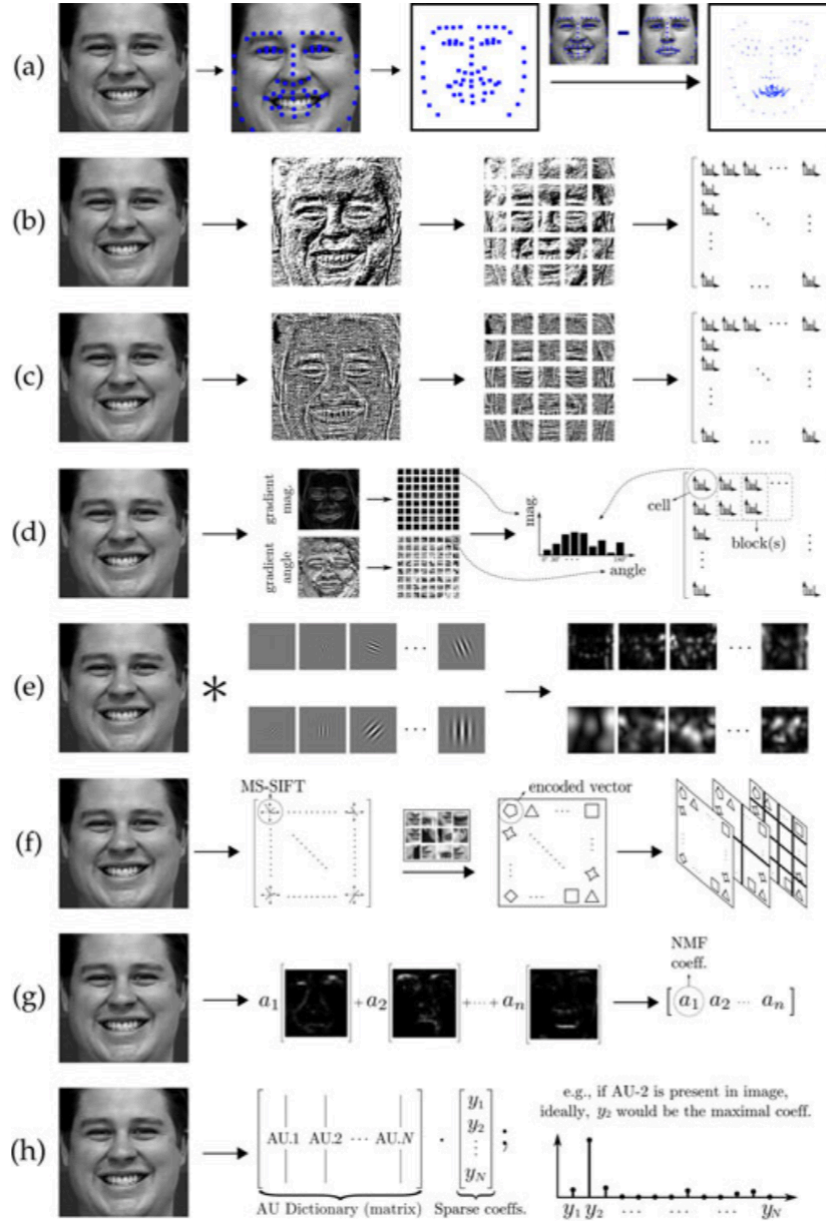
We unfold the description of several commonly used lower level representations in the following segments. These representations include local binary pattern (LBP) and local phase quantisation (LPQ), histogram of gradient (HOG), Gabor feature, scale invariant feature transformation (SIFT) feature representations. A demonstration of HoG representation is shown in Fig.1.5.

Local binary pattern (LBP) descriptor [13] captures the local pixel intensity change, i.e., the computed LBP values for each pixel in one designated image can merely reflect the intensity change in proximity range of the neighbourhood pixels. The convenience of applying LBP descriptor to acquire the LBP value for each pixel can enlarge the pixel differences, which may infer as the important features for discriminative tasks. A close relative to this representation is the local phase quantisation (LPQ) [14], which is used mostly in detecting the blur insensitive texture information. The demonstration of LBP and LPQ representations are shown in Fig.1.5 (b) and Fig.1.5(c).

The HoG (histogram of gradient) method [15] aims at detecting the abrupt intensity change on the surface of input images, which can be viewed as a good feature descriptor for finding edges and corners. To compute HOG feature vectors/maps, we firstly apply L2 normalisation on global images, e.g., either computing the square root or the log of each colour channel. Then we compute first order image gradients to capture the contour and silhouette from the images. A demonstration of HoG representation is shown in Fig.1.5(d).

Similar to 'simple cell' in the primary cortex, the Gabor-like filters [16] are designed to extract texture and edge features, are linear filters. 2-D Gabor filter is a Gaussian Kernel Function modulated by a sinusoidal plane wave. The computational process is identical to the convolution, except the pre-assigned weight in mask, which is designed for sliding over the image. A demonstration of Gabor feature representation is shown in Fig.1.5 (e).

SIFT, the shorthand term for scale invariant feature transform. SIFT features [17] allow for objects in multiple images of the same location, taken from different positions within the environment, to be recognised, and provide a set of features of an object that are not affected by many of the complications experienced in other methods, such as object scaling and rotation. A demonstration of SIFT representation is shown in Fig.1.5 (f).



In Fig.1.5, it demonstrates several widely used representations in automatic facial expressions. (a) facial points (b) LBP histograms (c) LPQ histograms (d) HoG feature; (e) Gabor feature (f) BoW (g)NMF; (h) sparse coding. This is an abridged rendering of Fig.2 in work of [11] with their kind permission to redistribute.

Fig. 1.5 Low- to High-level representations of facial expressions.

Mid-High Level Representations

All prior discussed representations are low-level, their features excel in detecting edges and textures. Summarised in [11], past widely usage of these representations in affective recognition tasks had been proved a series of success. However, there is a common flaw that suffers from almost all types of low-level representations. That is the lack of semantic, affective aware interpretable meaning for these low-level representations. The ideal affective representation should be at least semantically interpretable in certain degree, thence, a type of data-driven mid or high-level representation of facial expressions is demanded.

Conventionally, both none negative matrix factorisation (NMF) [18] and sparse coding [19] methods encode the representations are semantic interpretable. In specific, NMF approaches decompose a matrix into two non-negative matrices that can be assigned with varied semantic interpretations. The rendering of this technique is shown in Fig.1.5(g). Differ to the NMF methods, the usage of sparse coding in learning mid-high-level representation assumes any image based facial expressions can be transformed into a dense dictionary associated with a group of sparse coefficients (with lots of zero coefficients). The transformed dictionary can be corresponded to different AUs, shown in Fig.1.5(h).

Hierarchical Representations

Indeed, it is ideal to encode facial expressions in low- to high-level manners altogether in single computational model. The foremost and widely applied learning paradigm for hierarchical representations is deep neural networks. Armed with layered architecture and the under-complete hidden layer structure, i.e., the dimensionality of hidden vector is much smaller than the input, it ensures the learning of a cascade of representations that ranges from simple cell like low-level representations to semantic interpretable high-level ones [20] [21]. One indisputable merit of using deep neural network paradigm is its assumed freedom from important but labour-intensive feature engineering. Moreover, in comparison to the prior methods, the representations learning process can be fully automatic in the end-to-end form. The technical review of the NN learning paradigm will be detailed in the following Chapter 2.

1.3.2 Issues

Both low- and high-level representations have their innate weaknesses thwarting advancing their usage in automatic facial analysis. For low representations, despite of their robustness to illumination and mild head-pose variations, its induced difficulty in perceiving these rep-

representations in the semantic interpretable manner. High-level representation ameliorate this issue but are general susceptible to generic image processing issues.

Hierarchical representations – merges the merits of both low- and high-level representations – seem to present us the panacea to cure all the issues. However, we argue that, even arming with the learned hierarchical representations, to proclaim that the usage of deep neural networks is the copestone of learning affective representations of facial expressions remains myopic.

The noted short-sightedness of treating deep neural networks as the definitive answer to our quest for ultimate affective representations can be elaborated into two pressing issues.

- Lack of attention towards dimensional representation of emotion

A much severe issue comes with the overwhelmed focus on discrete emotions whereas dimensional representations are largely ignored. Contrast with the basic emotion account, dimensional models of emotion suggest that a common and interconnected system is responsible for all affective states and experience [22]. This envisioned system should allow the mapping of all affective states onto a dimensional plane. The most prominent two-dimensional theoretical model is the circumplex model [23]. The circumplex model upholds the emotions including facial expressions should be distributed in a two-dimensional circular space containing arousal and valence dimensions. In spite of the unsettled argument on whether human facial expressions should be understood in discrete or dimensional manner, from the computational perspective, the adoption of circumplex model in representing facial expression enjoys several merits such as compostability; versatility. Hence, it is imperative to consider the affective representations that based on dimensional, continuous circumplex model of emotions.

- Reliance on ambiguous annotations

The primary issue relates with the nature of labelled training datasets. Undoubtedly, a good affective representation that are encoded from a deep neural network can serve as the useful input to a supervised predictor. The quality of these representations ties closely with the fed training dataset. However, given the nature of facial expressions – the interpretations of them vary across different contexts and individuals – their annotations are less likely to reach the consensus. Different to other recognition datasets such as object or colour discrimination datasets, the occurrence of ambiguous annotations is much common in facial expression datasets. A running example is demonstrated below in Fig.1.6, showing that ubiquitous nature of ambiguity in facial expressions. These ambiguous annotations of facial expressions jeopardise the foun-



In Fig. 1.6, it demonstrates the pressing issue of ambiguity in single-labelled training dataset. This threatens the validity and reliability of majority of expression dataset. The running example is on the most widely used FER2013 expression dataset [24]. The upper annotations are the original ones, whereas the lower ones are the reworks from [25]. It is obvious that most of annotations from two groups of coders are failed to reach the agreement.

Fig. 1.6 Demonstration of the ambiguity issue in FER2013 expression dataset.

dation of the supervised learning paradigm, which is used in majority of prior noted representation learning approaches.

1.4 Priors for Representation Learning in Automatic Facial Analysis

Glancing over the foregoing text, we gradually narrow our research focus from the general affective computing to the learning of affective representations of facial expressions. Through a brief review on past attempts and highlighted pros and cons of these learnable representations, we render out our criteria for a good representation of facial expression in the following point-wise format.

- **Complex**

The concept of complexity refers the richness of the informative features of an encoded representation preserves. Mentioned in the foregoing section, the learning of hierarchical organisation of representations ranging from low- to high-level is a fine realisation of the concept of complexity. Undoubtedly, the most influential learning paradigm for learning hierarchical representations is deep neural networks. For this

account, in our search for complex representations of facial expressions, we frequently resort on deep neural networks.

- **Affective Aware (Semantic Interpretable)**

The second featured characteristic for being a good representation to model facial expressions is the concept of affective awareness. This targeted feature ensures semantic interpretation of the learned representations in certain degree. On facial expressions, the semantic features are mostly related with the anatomical structure of a face. This posits the enquiry of learning affective aware representation as the request to learn anatomical, deformable features relating to facial expressions. Note here, a high-level representation does not necessarily imply our aimed affective awareness feature, thence the employment of a deep neural network may not be sufficient in learning such representations.

- **Dimensional**

The third, perhaps foremost, but most challenging criterium to satisfy is the realisation of a continuous representations². This reflects the desire to use the dimensional circumplex model to represent the facial affects. The challenging aspect of realising this property lies on its difficulty from the algorithmic perspective. Precisely, to learn our targeted continuous or dimensional representations, it demands the training of a computational model to be totally independent of any sorts of annotations, which lead towards the learning of discrete affective representations. Moreover, besides the demanding requirement of the learning manner, a continuous representation should be bifurcated into two continuous factors that are independent with each other. These factors are ought to correspond to the arousal and valence dimensions in the conceptualised model.

1.5 Structure of This Thesis

In this thesis, we present three different approaches in attempting to learn complex continuous affective representations of facial expressions. The remaining thesis is unfolded as it follows. Prior to the delineation of our three researches, we firstly render out a concise technical review on some selected directed graphical models, e.g., neural networks, Bayesian neural networks, and variational auto-encoder (VAE), in *Chapter 2*.

²From this onwards, we use terms continuous and dimensional interchangeably.

To break the ice, in *Chapter 3*, we present our first approach: the usage of label relaxation approach to improve the classification performance of a neural network. In this research, we rely on a simple transformation technique that allows the discrete label distribution to be transformed into a partial continuous one in training a neural network to solve a facial expression classification task.

However, the prior approach only ensures predicting single label in categorisation of a facial expression. But in real world, many facial expressions may be interpreted in multiple ways, e.g., happy and surprise at the same time. Hence, it is critical to output multi-label predictions towards the same facial expression. To achieve this goal, we resort on a novel zero-shot transfer learning paradigm, e.g., uncertainty flow framework, in *Chapter 4*, to generate multi-label predictions on FER2013 dataset.

Unfortunately, these above-mentioned approaches merely ensure the discrete predictions on facial expressions, irrespective of single-label or multi-label predictions. Hence, in *Chapter 5*, we rely on the unsupervised learning paradigm to shun away from any sorts of emotional annotations. The objective of this research is to generate facial expressions along continuous scales. For this reason, we propose a novel form of variational auto-encoder, i.e., encapsulated variational auto-encoders (EVAE), to generate continuous expressions. The crux of this EVAE lies on the finding of two continuous factors that have direct impact on the expression generation process. Based on the identified continuous factors, we are able to train a EVAE to generate facial expressions along certain continuous axes.

On the path to learn complex, continuous and affective aware representation of facial expressions, we present three novel approaches. It is the very first time that a principled framework of learning such representations from diversified learning paradigms, e.g., supervised learning, zero-shot transfer learning, and unsupervised learning, can be rendered out. We wish this thesis – in spite of its nascent standing – will steer away from current research orientation on learning discrete representations to the learning of complex and continuous representations of facial expressions in affective computing.

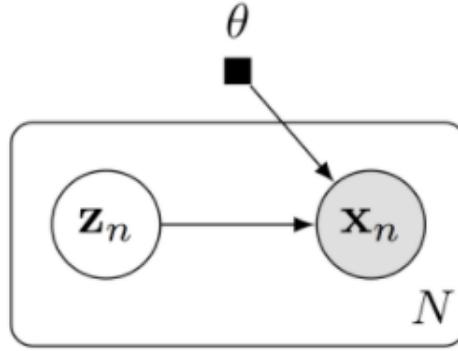
Chapter 2

A Principled Review on Selected DGMs in Representation Learning

From the preceding chapter, the essence and history of affective representation learning are briefly unfolded. To allow further chapters, e.g., our proposed approaches, to be digested more thoroughly, we devote this chapter to prepare the readers with a technical and principled review on representation learning from the probabilistic learning perspective. As the main focus of this review is to serve as a knowledge bed for seeding our later approaches, we are gravitated toward an exclusive review that rules out the models, concepts, learning algorithms that are less relevant to our approaches. Moreover, as a theoretical remark, we are agnostic to the convention dichotomy of supervised and unsupervised learning. The reason for that lies on the observations on human studies that revealed the trivial differences between aforementioned ones. And in real world, we are less prone to explicitly verify whether one learning process is in supervised or unsupervised manner.

For rendering out a principled preparation, we default the probabilistic graphical model as our fundamental umbrella framework. Probabilistic model – merges the classic probabilistic theory with the graph theory – breeds wealthy computational models that fuels the both theoretical and application advancements in machine learning and statistics. For full-range discussion on the progressive development of probabilistic models and its impact on machine learning, it is advisable to consult the following two books: [26] [27]. Furthermore, in order to keep this chapter concise, we attach a brief review in *Appendix A* on some commonly used laws and definitions in statistics and probability.

In large, there are two types of graphical models, e.g., directed or non-directed ones. The directed one reflects the causal relations between each edge and node, whereas the undirected ones only render the correlations among variables not causality. In this principled review,



In Fig.2.1, it renders out a sample DGM that contains multi-fold interpretations.

Fig. 2.1 A sample DGM with latent variables.

we omit the undirected ones, such as conditional random field model, and focus purely on the directed networks, denoting as **directed graphical model** (DGM) ¹.

One of central advantages that can be harvested from employing DGM in modelling is its induced conditional independence (CI). Consider three variables example, where X, Y, Z are three random variables, a DGM naturally assume the statement that X and Y are conditionally independent given Z , i.e., $X \perp Y|Z$, if and only if the joint distribution of $p(X, Y, Z)$ can be factorised into the following form:

$$P(X, Y, Z) = P(Z) \cdot P(X, Y|Z) = P(Z) \cdot P(X|Z) \cdot P(Y|Z). \quad (2.1)$$

Aside from its induced conditional independence, the usage of DGM is also convenient in rendering the model structure. A running example is jotted down below in Figure 2.1.

In Figure 2.1, it shows that data points x_i are conditionally dependent given a random/latent variable z_n , and conditionally independent given parameters θ . More specifically, if we have D - dimensional data $\{x_n\} \in \mathfrak{R}^{(N \times D)}$. The latent variables z_n for each data point x_n (local variables). In line with the PGM tradition, we are usually shading the nodes for observable variables; whereas the unshaded ones reflect as the hidden(non observable variable), and the direction arrows \longrightarrow represents the conditional independency in DGM (or more specifically the Bayesian network structure). The exemplar modelled joint distribution:

¹Commonly, this type of model is also denoted as bayes nets or bayes network (BN), and all three terms are used in this thesis interchangeably.

If we use the mixture and hierarchical model:

$$p(x, z, \theta) = p(\theta) \prod_{n=1}^N p(z_n | \theta) p(x_n | z_n, \theta). \quad (2.2)$$

If generative model is modelled, we use θ to model the latent variable z_n :

$$z_n \sim \text{Normal}(z_n | 0, I) \quad (2.3)$$

$$x_n | z_n \sim \text{Bernoulli}(x_n | p = NN(z_n; \theta)). \quad (2.4)$$

The usage of a DGM can be standardised as a sequence of process that incorporates defining a model, estimating the model parameters and making the inferences based on observation, to finally revise the model via unfitted observations. From above-mentioned components, it is unequivocally to state that the model specification is the most important step to consider as it determines the choice of its suitable learning algorithm and the learnable representations. Again, as the full scope of DGM ranges from latent linear model, kernel methods, Gaussian processes to more advanced state space model, it is unnecessary to go over every single model. Instead, the focuses are mainly on three main DGMs: **Neural networks**, **Bayesian neural networks**, and **Variational Auto-Encoder** that are involved heavily with our approaches unfolded in later chapters.

2.1 Neural networks

Since the coinage of contrastive divergence from Geoffrey Hinton [28] that they proved it is possible to learn deep neural nets via greedy layer-wise unsupervised pre-training, the renewed interests on neural nets (NN) have spawned all over the machine learning areas and applications.

There are several interpretations to understand the neural network and its variants, ranging from the biological to the constrained optimisation perspectives. Despite the original making of a neural network is far from the consideration of a DGM, we are able to fit the content of neural nets under the umbrella of DGM.

2.1.1 NN: under the Representers' theorem

Representers' theorem and logistic modelling

Prior to render out the DGM interpretation of a NN, we foremost begin the discussion with a classical Representers' theorem interpretation of a NN. To solve problems via parametric

modelling in machine learning, the Representers' theorem is the fundamental theorem we rely on. Basically, the Representers' theorem states as the approach to express the model parameters as $\{\theta, w\}$, and the features as x . Representer's Theorem entails the following statement: if there exists a model parameter θ , it can be decomposed into $\theta = \sum_{i=1}^m a_i x_i$. I.e., there always exists some real-value weight a_i that can be tuned in expressing the relation between inputs and outputs. Equipped with this notation, if we stick to $\{\theta\}$ as our model parameters, and $\{x\}$ as our observations, we can transform $\theta^T x$ as following Eq.2.5:

$$\theta^T x = \left(\sum_{i=1}^m a_i \cdot x_i \right)^T x^i \quad (2.5)$$

Armed with this expression, we can use it to express our to-be-tested hypothesis. Commonly, a hypothesis is expressed as $h_\theta(x)$, entails that in our hypothesis, the variables are $\{x\}$ and parameters as $\{\theta\}$. In linear model, as we are assuming our dataset is linear separable, we can parameterise the model in linear fashion (Eq.2.6).

$$h_\theta(x) = \theta^T x. \quad (2.6)$$

Denoting the hypothesis as z , where $h_\theta(x) = \theta^T x_i y = z$, and the loss function as φ , we can transform the foregoing linear model into a logistic non-linear model with the following non-linear mapping, i.e., $\varphi_{\text{logistic}}(z) = \log(1 + \exp^{-z})$. The resultant model is the *logistic model*.

NN: a series of logistic models

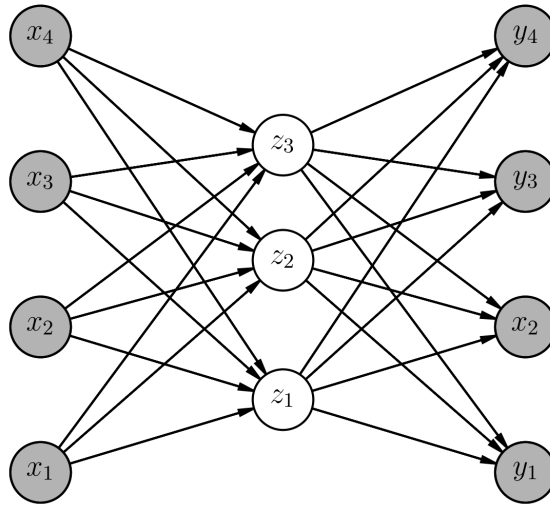
Under prior derived logistic models from the Representers' theorem, a feedforward neural network (FNN), aka, multi-layer perceptron (MLP) is merely a series of logistic regression models stacking up together. To smooth the delineation in the further section, we introduce some commonly used concepts in NN community along with a sample multi-layer FNN below as Fig.2.2.

- node/neuron

The smallest unit in a NN, on the high level, it can be perceived as a single unit to store each instance.

- input layer

The leftmost layer ($x_1 \dots x_4$) indicates the input layer. Regarding different types of input signal, it represent different bit of information in each unit (node). In image processing, each node in the input layer represents the binary data on each pixel of a single image.



In Fig.2.2, it demonstrates a simple two layer FNN or MLP. From the leftmost input layer to rightmost output layer, the middle one is the hidden layer.

Fig. 2.2 A sample FNN: one hidden layer only example

- hidden layer

The middle layer that is rendered in Fig.2.2. Mathematically, it delineates a deterministic function of the input. The nodes in the hidden layer commonly refer as the hidden units. The employment of hidden layer is crucial in a NN as it allows a NN to learn low-dimensional latent representations of observed input, since the mapping from leftmost to rightmost layers are commonly viewed from high-to-low in terms of dimensionality. Hence, it is a tradition to treat these z_n as the learned latent representations.

- output layer

The rightmost layer in Fig.2.2, reflects the output layer. The number of incorporated nodes is determined by the nature of the to-be-solved task. If a NN is deployed to solve a classification task, the number of nodes should be matched exactly with the prediction classes, whereas in a regression task, only a single output node is sufficed.

- Activation/transfer function

Each directed pointing arrow in Fig.2.2, represents the mapping or connection from one layer to the next. A important feature of a NN is that in each directed arrow, a non-linear activation function is applied to the recipient node. Hence, a NN can be

seen as a stack of non-linear functions, thence given enough hidden nodes, a NN is capable of modelling any suitably smooth function to serve as a universal approximator. The choice of non-linear activation function is open to the practitioners, the common options are ReLU, Tanh, and sigmoid functions, denoting as $g()$ function.

- **Weights and biases**

In a nutshell, weights and biases are the learnable parameters of a NN that come with their initialisation values (can be assigned or randomly initialised). The training of a NN is equivalent with recursive tuning its weights and biases. In NN community, it is custom to denote W as the big weight matrix that contains all the learnable weights in each layer. Here, to be more precise, we use different notations to denote input-to-hidden layer weight vector w and hidden-to-output weight vector v respectively. In theoretical derivation, the biases are likely to be omitted for the brevity.

- **shallow or deep architecture**

Depending on the number of employed layers, a NN can be seen as shallow, i.e., only one single hidden layer, or deep, i.e., has multiple (over 2) hidden layers. As the learning performance improves exponentially with the deeper architecture of a NN, it is now a custom to use a deep over shallow NN.

Convolutional Neural Networks

While a FNN or MLP is suitable for 1 dimensional signal, e.g., the raw sine or cosine wavelets of speech or text signal, the analysis on 2 dimensional signal like image works better on a variant of FNN: convolutional neural networks (CNN).

CNN has similar layer-wise architecture with a prior noted NN or MLP but with two below-listed modifications.

- **Convolution layer**

Differ to the defaulted affine operations in a vanilla NN, a CNN assumes the hidden units have local receptive fields (similar to the simple cell in the primary visual cortex in vivo), e.g., a black lined bounding box depicted in Fig.2.3, and the weights (connections) are tied or shared across the image to reduce the number of parameters. The convolution layer allows sliding a low dimensional kernel K on a two dimensional image I in generating the feature maps as following: $S(i, j) = (K * I)(i, j) = \sum_m \sum_n I(i - m, j - n)K(m, n)$, assuming image I and the applied kernel K have defined row and column as i, j and m, n respectively. The produced feature maps preserve

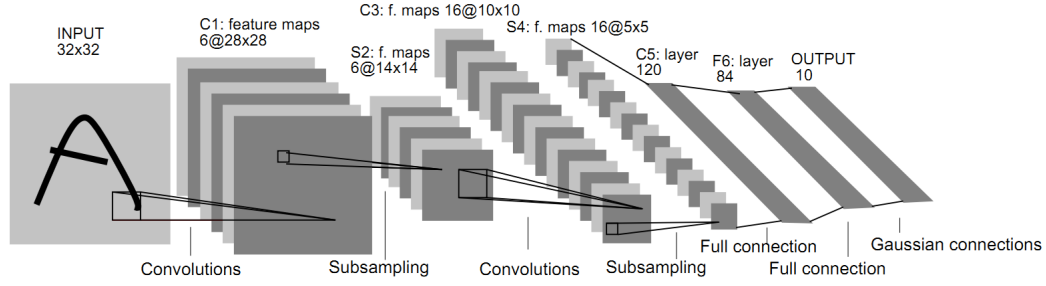


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

In Fig.2.3, it shows the foremost CNN model: LeNet-5 [29]. Reprint with the permission.

Fig. 2.3 A sample CNN: LeNet-5 model

most of informative features in the original image input. In the lower level convolution layer (cf. the leftmost convolution in Fig.2.3), the extracted features are largely similar to low level ones, e.g., the Gabor features and LBP ones. When it proceeds to the higher level convolution (cf. the second convolution in Fig.2.3), the encoded features contains more interpretable semantic features.

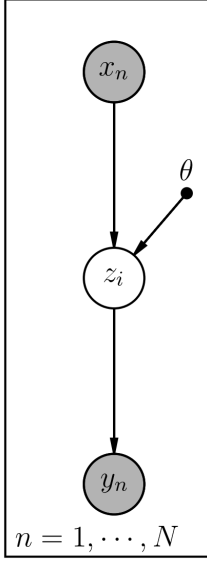
- subsampling (pooling) layer

As shown in Fig.2.3, a pooling layer is commonly inserted in between two consecutive convolutions. The role of pooling layers is two-fold. The primary one is to reduce the number of parameters to speed up the computation, whereas the other role is to introduce the translation invariance on the extracted feature maps. The induced invariance permits the encoded features to robust to small translation of the input, such as mild distortion with its accompanied sacrifice on losing the fine-grained details of the extracted features.

2.1.2 NN: as a DGM

DGM interpretation of a NN

The standard interpretation of a NN and CNN in the foregoing description is none probabilistic. To render a DGM interpretation of a NN, we chiefly transform the above-rendered rolled demonstration of a NN in Fig.2.2 to its compatible graphical model rendering in the following Fig.2.5. Inline with foregoing notations, both X and Y are observable variables, indicating input and output variables, whereas the z represents the latent variables, which



In Fig.2.5, it shows the graphical model rendering of a Feedforward NN with observable input and output variables.

Fig. 2.4 NN under the DGM interpretation

are parameterised as trainable model parameters θ . Following the foregoing notations on the inner weights and activation functions, the DGM description of a NN in solving a regression task can be derived as following Eq.2.7:

$$p(y|x, \theta) = \mathcal{N}(y|w^T z(x), \sigma^2) \quad (2.7)$$

$$z(x) = g(V(x)), \quad (2.8)$$

here we can view the foregoing non-probabilistic view of a NN as the core, the DGM interpretation can be seen as coating a NN with different parameterised output distribution. If a given task is binary or multi-class discriminative task, the preceding DGM derivation of a shallow NN can be changed accordingly into Eq.2.9:

$$p(y|x, \theta) = \text{Bernoulli}(y|w^T z(x), \sigma^2) \quad (2.9)$$

$$p(y|x, \theta) = \text{Categorical}(y|w^T z(x), \sigma^2). \quad (2.10)$$

$$(2.11)$$

DGM interpretation of a CNN

Similar with the DGM interpretation of a NN, the DGM interpretation of a CNN can be viewed as a two-step process of treating the inner convolution and pooling operations as the core, then placing a suitable output distribution to wrap the foregoing distribution. However, to delineate the DGM account of a CNN, it demands some extra terminologies and works.

The needed terminology is the prior probability distribution. This term will be reiterated in later discussion of Bayesian neural network. In short, the prior probability distribution is a probability distribution over the parameters of a model that encodes our beliefs about what models are reasonable in prior to expose to any data instances.

Depending on the how concentrated the probability density (low or high variance), a prior can be either weak or strong. A CNN can be seen as imposing infinitely strong prior on the model parameters and these parameters are valued as zero. The convolution operation can be comprehended as enforcing one hidden node must be identical to the weights of its neighbour but shifted in image, all weights must be zero except for in the small spatially contiguous receptive field assigned to that hidden node.

Learning Algorithm: back-propagation algorithm

For neural network community, there is no need for the extra introduction in stating the historic importance and current status of the back-propagation algorithm in training a NN. At its essence, back-propagation is just a clever application of the chain rule. The goal of back-propagation is to layer-wise compute the partial derivatives of the certain defined cost function J with respect to its associated weights and biases $\omega^{[l]}$, $b^{[l]}$, in l th layer, i.e., $\frac{\partial J}{\partial \omega^{[l]}}$.

Once we have these partial derivatives, we update the weights and biases in this layer via gradient descent algorithm, Eq.2.12:

$$\omega^{[l]} = \omega^{[l]} - \alpha \frac{\partial J}{\partial \omega^{[l]}}, \quad (2.12)$$

the α is the defined step size. The partial derivatives give us the direction of greatest ascent, thence, we take a small step in the opposite direction – the direction of the greatest descent, i.e., the direction which will take us to the local minima of the cost function. Due to the conciseness of this review, for more detailed review on back-propagation algorithm, please refer to the work of [30].

2.2 Bayesian neural network

2.2.1 The Caveat of Overfitting in NN

Besides the obvious computational hurdles, e.g., the issue of local minimum and the heavy burden of automatic differentiation for a large-scale network, an, the issue of over-fitting is also worthwhile to discuss here. In a nutshell, over-fitting describes as a phenomenon that a model learns the training data too well that thwarts its generalisability to unseen data. In most downstream applications, where the training of a model is mostly accomplished in the off-line manner, which the testing data is overwhelmingly outweigh the training data, then over-fitting is likely to jeopardise the performance of a model.

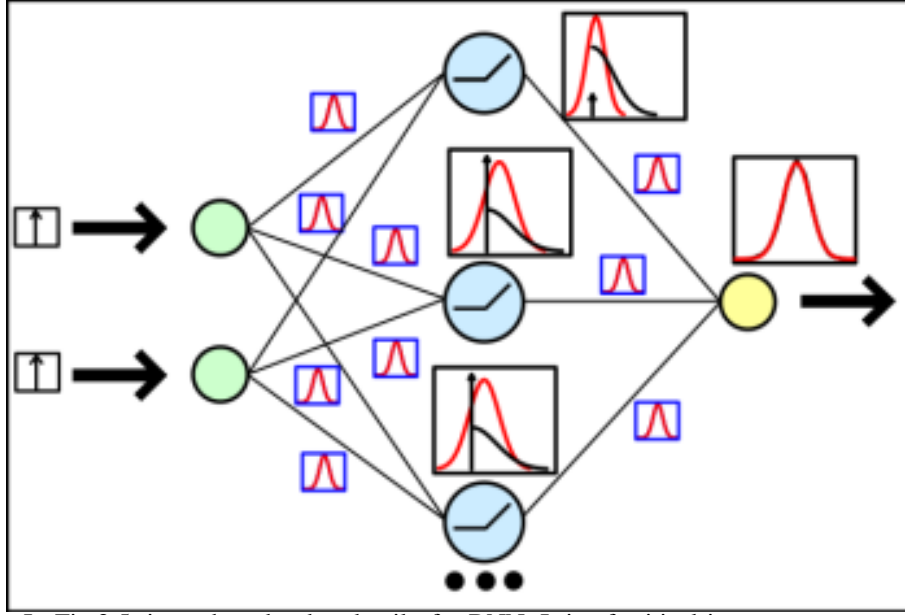
One way of viewing the overfitting problem from neural network community is the trade-off between the bias and variance of a NN from the complexity of a NN. This leads to a choice of a NN that varies with the amount of training data available – the more data, the more complex model can be chosen. This posits a practical issue in applying NN as it is almost impossible to reveal the complexity of upcoming data to tune the corresponding NN with the matched complexity.

From a Bayesian perspective, adjusting the complexity of the model on the amount of training data yield very little ratification. Under the framework of DGM, a Bayesian model defines a model, selects a prior, collects data, computes the posterior, and then makes prediction. There is no need in Bayesian modelling to change the model likelihood or prior depending on the quantity of data. I.e., if the model and prior is acceptable in terms of its performance for a hundred data instances, they should be correct for a thousand observations as well. Hence, under a Bayesian perspective, it is vain to implement the techniques such as cross validation to prevent the over-fitting issue of the parameters.

2.2.2 The Description on BNN

In light of preceding discussion, it is natural to view a NN under a Bayesian model. The resultant model is our aimed Bayesian neural network (BNN). Previously, BNN is merely treated as one type of learning that takes on a NN, i.e., the Bayesian learning of the NN [31] [32]. With the previous efforts from [33] [34] [35], a trend grows on treating this method as an important extension to conventional NN. This coins the terminology of Bayesian neural network (BNN).

In a nutshell, a BNN can be seen as a NN with a prior distribution on its weights [35], shown in Fig.2.5. The superficial similarity between NN and BNN belies their diversified learning process. In NN, the weights and biases are learned from the minimisation a measure



In Fig.2.5, it renders the thumbnail of a BNN. It is of critical important to note that in NN, the layer-wise weights are deterministic, but in BNN, the weights are stochastic probability, having no fixed values.

Fig. 2.5 Demo of a BNN

of 'error' on the training instances. However, in BNN, the learning objective is drastically different from this. Inline with most of DGMs, the learning objective of a BNN is to find the predictive distribution for the target values in the unseen 'testing' case, given the input in this case. This targeted predictive distribution can be rendered as Eq.2.13:

$$p(y^{(n+1)}|x^{(n+1)}, (x^{(1)}, y^{(1)}), \dots, (x^{(n)}, y^{(n)})) = \int p(y^{(n+1)}|x^{(n+1)}, \theta) p(\theta|(x^{(n)}, y^{(n)})) d\theta. \quad (2.13)$$

From this, to obtain an accurate measure of posterior density for the model parameters, i.e., $p(\theta|x, y)$, is the key to yield a good predictive density². Both model likelihood and the prior density are served for the derivation of our targeted posterior for parameters. For the model likelihood, we can use the defined ones directly from Eq.2.9 for either a regression or binary classification problem. Then the residual part is the choice of prior density for the parameters.

²Here, we assume i.i.d. data instance.

2.2.3 Prior Choices

In discussion of the prior choice in BNN, unfortunately, there is no general guidance for picking a prior for this type of DGMs. Despite of the lack of general principle in selecting the prior, we can resort on some heuristic mean basing on the property of different types of prior density. In the following text, we render out the taxonomy of prior distributions as a reference to highlight some of commonly used ones.

Despite of being named as the misnomer, the informative prior denotes as the normal conjugate or non-conjugate prior, which has strong not minimal influence on the posterior, but some objective Bayesian's may instead prefer priors which do not strongly influence the posterior distribution. There are a couple of classical uninformative priors, e.g., super flat priors, uniform prior.

Informative Prior

- Binomial & Bernoulli Prior
- Dirichlet Prior
- Logistic Normal prior
- Gamma conjugate prior
- Conjugate priors for exponential families

Uninformative Prior

- Jeffrey prior
- Reference prior

2.2.4 Posterior Inference

Equipping with the defined model likelihood and the chosen prior density for the parameters, we are positioned to derive the parameter posterior. Notwithstanding, the posterior distribution for the parameters of a BNN is typical very complex and intractable. For years, there are numerous tryouts in dealing with this intractable posterior inference. These attempts can be summarised into three categories: (1) direct inference; (2) approximation or sampling (MCMC) based inference; (3) variational inference [36].

2.3 Variational Auto-Encoder

The foregoing two DGMs, e.g., the neural network and bayesian neural network, shine their usages mostly in the discriminative modelling of the tasks, i.e., in solving either a regression or classification problem. Their performance of defaulted parameter fitting approaches, i.e., back-propagation algorithm in either frequentist view or maximum likelihood estimation in bayesian perspective are largely determined by the quality and quantity of the labelled observations. Their encoded representations is nonetheless dataset dependent and narrow focused, as the true wealth of representation learning is still buried beneath the supervised learning. In other words, without the full exploitation of unsupervised learning, we are distant away from mining the true affective aware representation in affective computing.

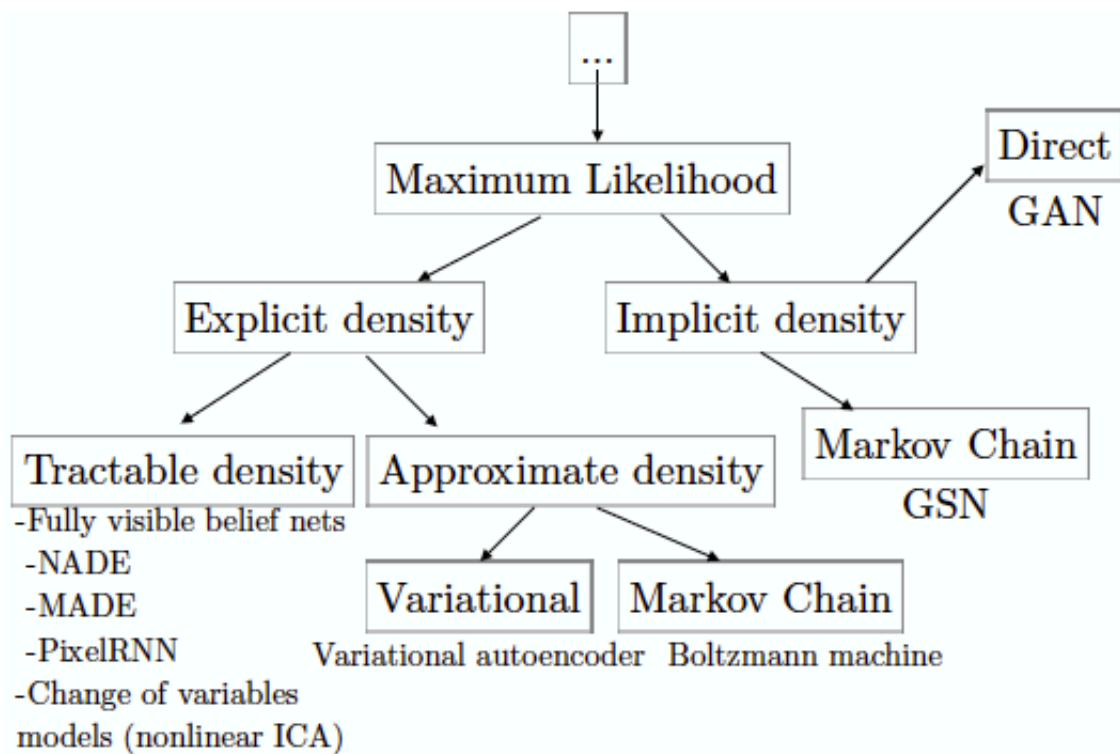
As a good portion of unsupervised learning methods, the generative modelling is a mature probabilistic approach that insists a view of 'analysis-via-synthesis', i.e., the understand of the world is through recreating the world per se. This enforces generative networks to learn good representations of the observations, and harnessing them in the procedure of data regeneration. The scope of generative networks is spanning from early linear and non-linear principle component analysis to now widely used generative adversarial networks and variational auto-encoder. We attach a fine-grained taxonomy of mainstream generative networks below in Fig 2.6.

As one celebrated generative model, variational auto-encoder (VAE) allows explicit and fast approximate marginal inference of the dataset x . Since its birth, the interpretation of VAE has been diversified into several streams ranging from the frequentist view, i.e., the coding theory and neural network perspective to a bayesian perspective, i.e., the discussion over global v.s. local variational parameters. Hence the following sections are unfolded into two segments in delineation of both frequentist and bayesian views on VAE.

2.3.1 VAE: Coding Theory Interpretation

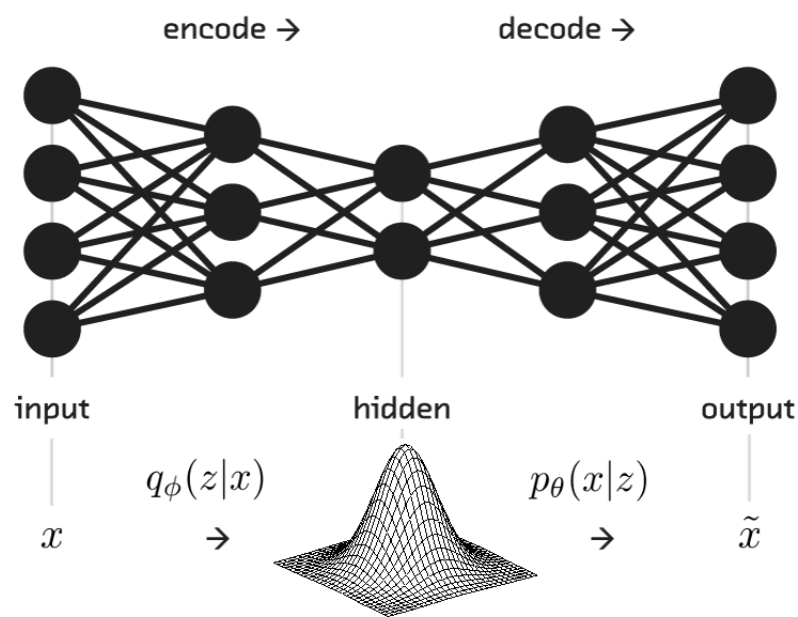
From the perspective of coding theory, a variational auto-encoder (VAE) can be seen as perceived as a sandwich-like auto-encoder structure, where the outer layers "bread" are the encoder and decoder respectively. Rendered in Fig. 2.7, the crucial "sandwich meat" in VAE is the hidden representations that are encoded from the encoder. These hidden (latent) representations are normally low-dimensional and crucial in representing certain salient features of the inputted data instances. The decoder utilises these features to reconstruct the input signal.

Not only reserved with the prior-mentioned neat structure of a conventional auto-encoder, the beauty of a VAE lies its usage of two neural networks to substitute the encoder and



In Fig.2.6, it offers a coarse taxonomy of several mainstream generative models. This figure is original produced in the work of [37]. We appreciate the generosity of the author in permitting us to redistribute this figure here.

Fig. 2.6 Taxonomy of Deep Generative Models



In Fig.??, it outlines a VAE under the coding theory: a pair of encoder and decoder

Fig. 2.7 Encoder Theory interpretation of a VAE

decoder respectively. This degrades the parameter estimation of a VAE to the learning of two easy-to-train neural networks. As a running example, we assume the input data instance is an image with 28x28 dimensions, this serves as the input to learn the hidden representations of this image, e.g., a two-dimensional representation. The corresponding decoder utilises this encoded hidden representation to regenerate a new image that looks similar to the original image.

2.3.2 VAE: DGM Interpretation

Different to the foregoing simplified explanation of a VAE under the coding theory, we are gravitated towards a more principled probabilistic compatible explanation. Importantly, a VAE can be viewed one form of DGMs with specification on the probability model of data and latent variable. As a DGM, a VAE also consisted of the classic trilogy: a prior, a defined model likelihood and a to-be-inferred posterior. Similar to other DGMs, the computation of the to-be-inferred posterior is intractable, i.e., the computational time of the evidence $p(x)$ in deriving the posterior grows exponentially. Hence, to compute this intractable posterior, a plethora of past attempts have been devoted to accomplish this goal, ranging from early exact inference, Monte-Carlo sampling, to later developed variational inference.

Variational inference – as one form of inference networks for parameterising approximating distributions – relies on the variational family to approximate the foregoing intractable posterior. Early researches on variational inference implied the usage of local latent variables, as Eq.2.14:

$$q(z; \lambda) = \prod_{n=1}^N q(z_n; \lambda_n), \quad (2.14)$$

here z_n are the latent variables and λ are the factorised latent variational factor assuming mean-field approximating distribution. However, the size of λ grows with the size of data, this posits a computational issue for fitting large-scale dataset into the memory, and lowers down the entire inference speed. To ameliorate this issue, the coinage of probabilistic encoder allows the employment of a neural network to take x as input and which outputs its local parameters λ , shown in Eq.2.15:

$$q(z|x, \theta) = \prod_{n=1}^N q(z_n; \lambda_n = NN(x_n; \theta)), \quad (2.15)$$

where NN represents a neural network or MLP. This technique gain its fame to be termed as: amortised inference. Literally, through adapting a neural network in defining a set of

global parameters, we are able to fix the number of to-be-estimated parameters (including both model and variational parameters) given a fixed structure of a neural network. This neural network wrapped variational approximation of the intractable posterior is the DGM interpretation of the encoder.

In terms of the probabilistic decoder, under the framework of DGM, it can be seen as a specification on model likelihoods. I.e., it is a distribution $p(x|z)$ over each value (discrete or continuous) x given an encoded latent code (representation) z . The process of x generation is dependent upon the value of z and the mapping between z and x ³. Rather than assuming the simplified linear mapping between x and z , which makes several classic latent linear models, such as principle component analysis, independent component analysis, factor analysis, and et ac. It is common to apply a NN or MLP to parameterise the probabilistic decoder for its allowed compositionality and assumed non-linearity that beats most of linear model in representation learning and feature extraction. As a result, for the real valued and binary data, we can parameterise the decoder in the following two ways respectively in Eq.2.16

$$p(x|z) = \text{Normal}(x|[\mu, \sigma^2] = NN(z; \theta)) \quad (2.16)$$

$$p(x|z) = \text{Bernoulli}(x|p = NN(z; \theta)). \quad (2.17)$$

To end the discussion over the DGM interpretation of a VAE, we lay out the following technical summary corresponds to each component that depicted in Fig.2.8.

- Prior: $p(z) = \text{Normal}(z|0, I)$
- Decoder (model likelihood): $p(x|z) = \text{Normal}(x|[\mu, \sigma^2]) = NN(z; \theta)$
- Encoder (approximating posterior): $q(z|x; \phi) \approx p(z|x; \theta) = \prod_{n=1}^N q(z_n; \lambda_n)$

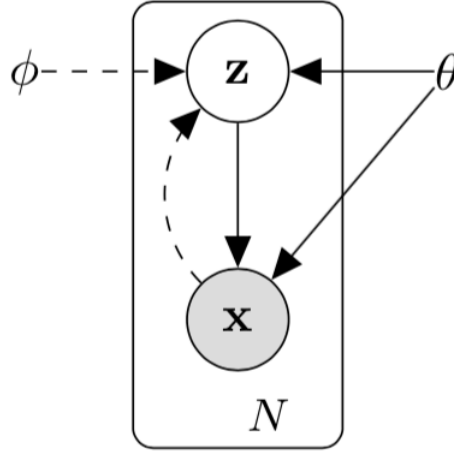
2.3.3 Learning VAE

Objective Function

Armed with the following coding theory and DGM interpretations of VAE, it is natural to discuss in the succeeding text over its implied objective function.

The derivation of the targeted objective function starts with the defined KL divergence (the detail of KL divergence can be consulted in *Appendix A*) between the employed variational encoder (approximation) $q(z|x)$ and the true posterior $p(z|x)$ in Eq. 2.18

³For its generative property, it is sometimes named as generative network.



In Fig.2.8, it demonstrates the DGM interpretation of a VAE. The dotted arrows represented the variational encoder with variational parameter ϕ , whereas the solid arrows represent the decoder with model parameter θ [38].

Fig. 2.8 Probabilistic interpretation of a VAE

$$KL(q(z|x)||p(z|x)) = \mathcal{E}_q[\log q(z|x)] - \mathcal{E}_q[\log p(x, z)] + \log p(x), \quad (2.18)$$

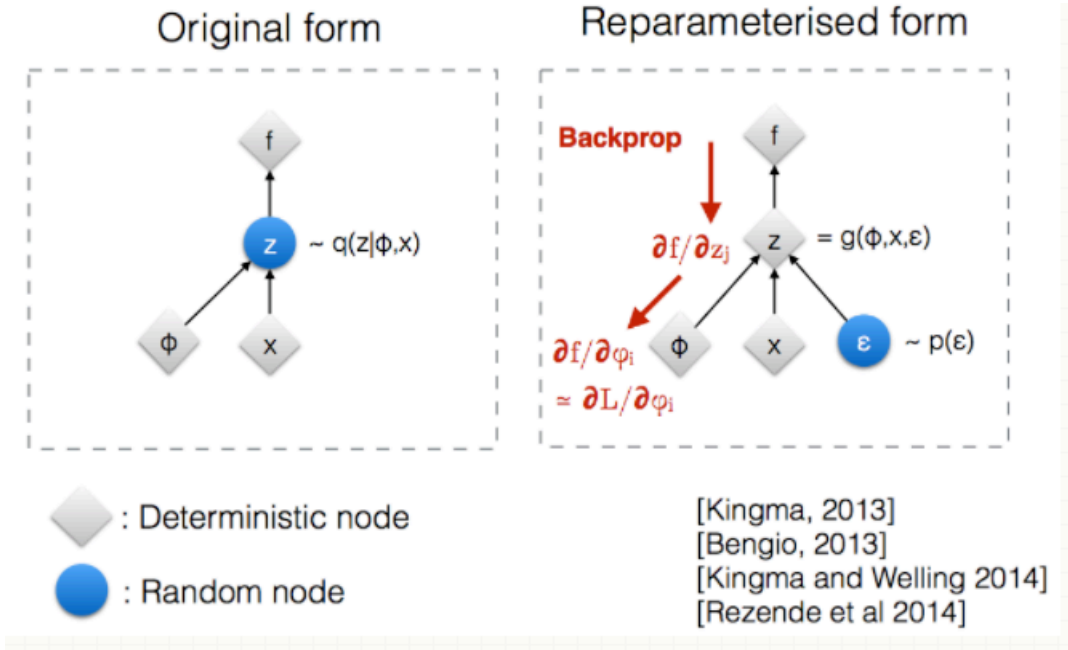
assuming the marginal density $\log p(x)$ is fixed, therefore, we can optimise the residual parts. With the repeated usage of Bayesian' rule, we are able to derive the following form as Eq.2.19:

$$\mathcal{E}_{q(z|x)}[\log p(x|z)] - KL(q(z|x)||p(z)). \quad (2.19)$$

Here this derived analytic form is denoted as evidence lower bound (ELBO) of the original KL associated objective. It is important to note that the optimisation objective for ELBO is maximisation instead of original minimisation objective (in terms of KL divergence).

It is interesting to note that from Eq.2.19, the objective function can be understood as two components. The first component, e.g., $\mathcal{E}_{q(z|x)}[\log p(x|z)]$ is the expected reconstructed loss that measures the differences between the original data instance x and the reconstructed input, where comes from the samples of the decoder $\tilde{x} \sim p(x|z)$. Whereas the second component acts as a regulariser to encourage the approximate the posterior, i.e., the variational encoder, to be close to the assumed prior $p_\theta(z)$.

In terms of parameter estimation, as both global variational parameters and model parameters from the corresponding encoder and decoder are determined via the parameters of parameterised neural network(s), both parameters are optimised in the same time.



In Fig.2.9, it delineates the role of re-parameterisation in letting stochastic node in deep NN to be run the back-propagation rule. Copyright of this nicely plotted demo remains for Dr.Kingma in his tutorial talk at NIPS2015 workshop on Variational Learning.

Fig. 2.9 Explanation on reparameterisation trick

Reparameterisation Trick

Following the foregoing text, the beauty of a VAE can be seen as its attempts to replace local variational parameters with the painstaking global parameters, where the production of these local variational parameters can be wrapped by the neural network. However, as there is technical barrier needs to be jumped off: it is impossible to run gradient based optimisation method, e.g., back propagation algorithm on a model that contains stochastic units rather than the deterministic units. Referring to the left delineation in Fig.??, given the observable variable x and a random (stochastic) variable z , the deterministic mapping (the nonlinear function) of two variables, i.e., $f(x, z)$, the gradient w.r.t the parameter ϕ is $\nabla_{\phi} \mathcal{E}_{z \sim q(z|x)}[f(x, z)]$.

Historically, the computation of this gradient is notoriously hard and tends to be high in variance due to the value of z is dependent upon an unknown density $q(z|x)$. Reparameterisation trick comes to the game to express the unknown density $q(z|x)$ with certain conditions. In specific, for a Gaussian prior, we can express the distribution $q(z|x)$ in two step procedure. Firstly, it starts with the sampling of a noise variable ϵ from a simple distribution $p(\epsilon)$ to be a standard Normal, i.e., $\epsilon \sim p(0, I)$. Secondly, we apply a deterministic

transformation $g_\phi(\epsilon, x)$ that aims to map the random noise ϵ onto our targeted distribution $q(z|x)$, assuming $q(z|x)$ is another Gaussian distribution, this deterministic mapping can be rewritten as $z = g_\phi(\epsilon) = \mu + \epsilon \cdot \sigma$.⁴ Now, we can get much lower variance samples of z in running the back propagation algorithm. That is to say, instead of computing the hopeless $\nabla_\phi \mathcal{E}_{z \sim q(z|x)}[f(x, z)]$, we now pull the expectation term out in computing of $\mathcal{E}_{\epsilon \sim p(\epsilon)}[\nabla_\phi f(x, g(\epsilon, x))]$, where $p(\epsilon) \sim \mathcal{N}(0, I)$.

In terms of the generalisation of this trick, this manner of reparameterisation is applicable to 'local-scale family' of distributions, e.g., Normal distribution, student T distribution, Laplace distribution, and et ac. Furthermore, in the forthcoming chapter in rendering our modification of a VAE, we will discuss a similar but more general training algorithm: automatic differentiation variational inference (ADVI) in *Appendix C*.

⁴The linear transformation of a Gaussian distribution is guaranteed to be another Gaussian distribution.

Chapter 3

Learning Single-Label Relaxed Representations via Label Relaxation

3.1 Introduction

This research focuses on the learning of single relaxed affective representations via label relaxation technique. **We try to use the relaxed labels to improve the performance of a neural network in classifying the pictorial facial expressions into discrete emotional states.**

To improve the classification performance of a neural network under the supervised learning paradigm, common approaches range from the usage of a regulariser [39] to the more convoluted architecture design of a neural network. Little attention is paid towards improving the credibility of assigned one-hot encoded label information as most one-hot encoded annotations are ambiguous [24]. I.e., in comparison to the object classification task, the provided label information is low in creditability, and is susceptible to be controversial under the human eye. Therefore, to improve the classification performance of a neural network on pictorial facial expression dataset, we target on transform the one-hot encoded label to a real-numbered label expression.

To achieve this goal, we rely on a label relaxation trick that is original proposed by [40] [41] to relax the one-hot encoded emotional annotations to the real-numbered format. Compare with their original usage of categorical reparameterisation in training a stochastic network, we use the one-hot encoded labels as input rather than the logarithmic value of class probability to train a non-stochastic network in solving a classification task.

Xie et al [42] developed their label transformation method, i.e., DisturbLabel, which is independent from our work. In their DisturbLabel approach, they add noise to the loss layer

serves an additional model regulariser to intentionally predict the incorrect labels. Different from ours, the entire label perturbation process in [42] is accomplished through random walking over the entire label sets to generate labels randomly. In contrast with theirs, the usage of label relaxation technique does not aim to generate random labels, but rather to relax the original one-hot encoded labels to real-number forms that share the semantic similarity with the original labels.

In detail, we first relax the original one-hot encoded labels into real-number forms. This allows the construction of a relaxed training dataset, i.e., concatenating these relaxed labels with facial expression images. Then, we fed the original and relaxed training datasets to train two neural networks with identical layer-wise configurations. After training, we fixed the learned weights of two neural networks for testing. On the testing phase, we use a novel set of facial expression images to test the performance of two neural networks based on the evaluation of their predicted discrete labels. The empirical experiment shows that the usage of relaxed label in neural network training can improve the performance of a neural network on single-label facial expression classification task.

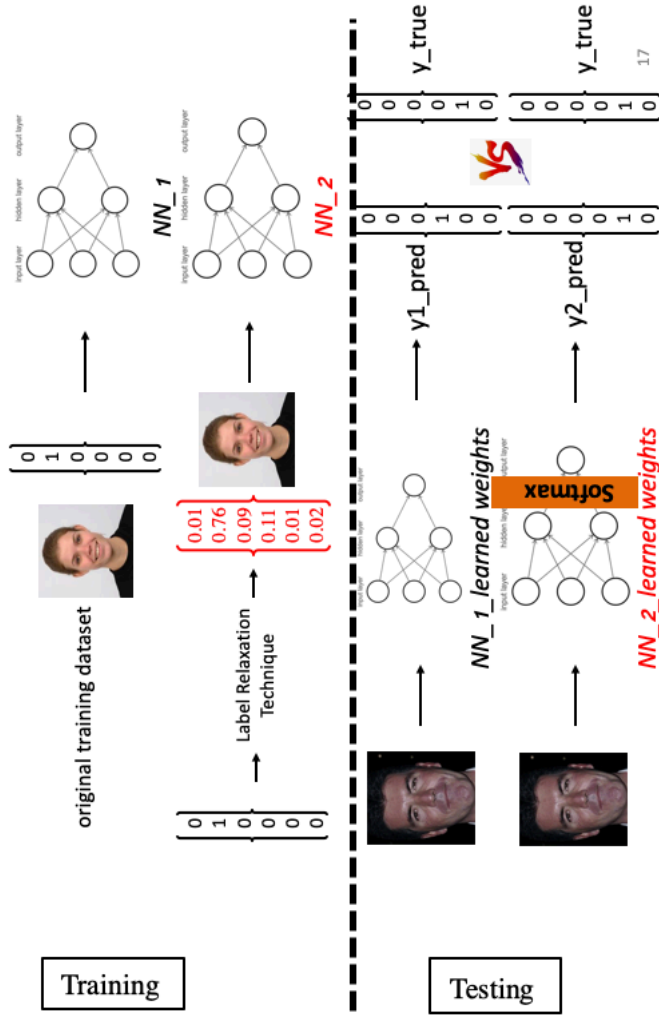
3.2 Methodology

3.2.1 Motivation of Adoption of Label Relaxation

The motivation of this research is to elevate the classification performance of a computational model on facial expressions. Conventionally, when training a model to tackle with a classification problem, the usage of one-hot encoding to transform a categorical label to its numerical form becomes standardised prior to the model training. However, as illustrated in 3.1, in the context of pictorial emotional classification, the relied categorical labels are largely vague and susceptible to be ambiguous. Utilising these labels in training a computational model to solve a facial expression classification task may lead to the lowered prediction accuracy.

Therefore, to improve the prediction accuracy, it is not appropriate to lend total credibility to the 'correct' annotation, e.g., 1 in one-hot encoding, without considering the 'false' annotations in one hot encoding of a categorical label, e.g., 0 in one-hot encoding. That is to say, a method of transforming the original one-hot encoded labels to a new numeric form of labels is needed. The resorted method is the label relaxation technique. This technique is rooted in Gumbel-Softmax trick that is initially proposed for categorical reparameterisation [40] [41]. Their proposed Gumbel-Softmax trick allows the conversion of a discrete variable to a partial continuous one. However, different from their usage of probabilistic variables

in defining the transformation process, here we use the one-hot encoded labels directly to achieve the transformation. Based on the label relaxation technique, the original one-hot encoded 'hard' labels can now be converted to real-numbered form of 'soft' labels. We then concatenate these 'soft' labels with the facial expression images to constitute a novel training dataset. We feed this novel training dataset to train a computational model, e.g., a neural network, to solve a classification task. The working pipeline of the neural network training that is based on relaxed labels can be visualised below in Fig. 3.1.



In Fig.3.1, it demonstrates the working pipeline of training and testing phases for two neural networks with and without the relaxed labels. The upper panel indicates the training case, whereas the lower panel demonstrates the testing phase. In the training phase, two neural networks are fed with original one-hot encoded or relaxed labels for supervised training; whereas in the testing phase, we compare the prediction accuracy of two neural networks, e.g., NN_1 and NN_2 on the testing dataset. It is important to note here, as our testing dataset only involves correct single-labelled annotations for evaluation, we add an extra *Softmax* layer on top of the last hidden layer in NN_2 for producing one-hot labels for evaluation.

Fig. 3.1 Working pipeline of neural network training based on relaxed labels

3.2.2 From Gumbel Variable to Gumbel-Softmax Trick

Gumbel Variable

The parameterisation we choose here is the Gumbel variable for its potential in controlling whether or not the label distribution is discrete or continuous. Gumbel distribution is previously used in modelling the distribution of the extreme value [43]. Formally, we can define a Gumbel variable that belongs to a standard Gumbel distribution as $G \sim \text{Gumbel}(\mu, \beta)$, whereas the distribution location and variance parameters are termed as μ and β , respectively. The reason for picking the Gumbel variable lies on its close connection to uniform distribution. E.g., if U is the uniform distributed, i.e., $U \sim \text{Unif}[0, 1]$, the transition between G and U can be parameterised as Eq.3.1:

$$G(U) = -\log[-\log(U)]. \quad (3.1)$$

Gumbel-Softmax Trick

The parameterisation of the Gumbel variable unfolds its convenience in sampling discrete distribution via the Gumbel max trick [44], which insists the sampling process for any categorical distributions can be delineated as firstly define a Gumbel variable, then adding it to a normalising constant, and output the summation. Formally, if we allow our designated label information y to be a discrete categorical variable, which can take K different values/labels to form one-hot representation in a standard binomial or multinomial discriminative task, e.g., $y \in \{1 \dots K\}$ & $y \in \mathfrak{R}^K$. The class probability that associated with this categorical variable y is defined as π_k .

The entire label sampling process can be parameterised as Eq.3.2:

$$y \sim \text{Cat}(\pi_k) = \text{argmax}_k(\alpha_k + \log(\pi_k)). \quad (3.2)$$

Where the Gumbel variable is defined as $\alpha_k \sim \text{Gumbel}(0, 1) \sim -\log(-\log u_k)$, whereas u_k confirms to the uniform distribution, e.g., $u_k \sim \text{Uniform}(0, 1)$.

The problem of Gumbel max sampling process lies on the *argmax* function, which only permits the 'hard' sampling distribution, i.e., categorical. For tackling with this issue, Jang et al in [40] came up a simple trick, i.e., Gumbel-Softmax trick that uses a simple smoothing function, i.e., *softmax* with a tuning hyper parameter τ to derive continuous approximation to the previous *argmax* function. This allows the generation of sample, e.g., y_k^τ , follows $y_k^\tau = \text{softmax}(\alpha_k + \log(\pi_k)) = \frac{\exp((\log \pi_k + \alpha_k)/\tau)}{\sum_{j=1}^k \exp((\log \pi_j + \alpha_j)/\tau)}$.

3.2.3 Label Relaxation Technique

Similar to Jang et al proposed Gumbel-Softmax trick in [40], we also use a softmax function with a tuning hyper-parameter τ to approximate the argmax function in Eq.3.2. The crucial difference is that in our label relaxation technique, we do not resort on the probability class that associated with the categorical variable, instead, we use the one-hot encoded labels, e.g., y_i directly without taking its log value. The relaxed label y_i^τ follows the generation process in Eq.3.3:

$$y_i^\tau = \text{softmax}(y_i + \alpha_i) = \frac{\exp((y_i + \alpha_i)/\tau)}{\sum_{j=1}^i \exp((y_j + \alpha_j)/\tau)} \quad (3.3)$$

For the clear demonstration of our presented label relaxation technique, we present the Algorithm 1 below to delineate the process of crafting our envisioned 'soft' labels, e.g., y_k^τ .

Algorithm 1 Label relaxation technique

Require: The original one-hot encoded label y_i ; the number of sample N

Require: Set the value of hyper-parameter τ

for $y_i \in Y$ **do**

 sample N uniform variables, i.e., $U_i^N \sim \text{Uniform}[0, 1]$

 generate N gumble variables α_i^N , i.e., $\alpha_i = -\log[-\log(U_i^N)]$

$y_i^{\tau N} \leftarrow \frac{\exp((y_i + \alpha_i)/\tau)}{\sum_{j=1}^i \exp((y_j + \alpha_j)/\tau)}$

$y_i^\tau \leftarrow \frac{y_i^{\tau N}}{N}$

end for

Return y_i^τ

Where $\tau > 0$ acts as the temperature parameter to control the degree of relaxation, e.g., when $\tau \rightarrow 0$, y_i^τ becomes one-hot encoded; when $\tau \rightarrow +\infty$, y_i^τ is no longer one-hot encoded. For example, with the τ fixed at 4, a sample one-hot encoded label in a ten class categorisation task can be expressed as (0; 0; 0; 1; 0; 0; 0; 0; 0; 0), the crafted soft label is the vector of (0.0971; 0.0973; 0.0970; 0.1246; 0.0973; 0.0973; 0.0972; 0.0973; 0.0972; 0.0973). Numerically, the resultant soft labels enforce the model to prevent lending the total creditability towards the 'correct' labels in computing the loss, but consider other 'false' labels as well. This forced hard label relaxation may provide a cure to resolve the issue of vague labels in imbalanced dataset. Armed with the previously defined Gumbel-Concrete relaxed label, we can now implement this technique in training a supervised neural network by substituting the original one hot encoded labels with the relaxed labels. The rest part is trained conventionally with the gradient based supervised training algorithm.

3.3 Empirical Experiment

FER2013 is a well researched public available dataset for pictorial sentiment discrimination. It is comprised of total 35887 facial expression images. Prior to our implementation, all images in FER2013 had gone through the standard preprocessing steps, e.g., fixating the faces at center, standardise the image size to 48 by 48 pixels in resolution, and all faces are properly registered. We then normalised the pixel values of input images, and strictly follow the ratio of 8:1:1 for dividing the dataset into training, validation, and testing subsets for this experiment. All FER2013 images are labelled as one of seven emotion categories, e.g., angry, disgust, fear, happy, sad, surprise, and neutral. The sample size for each emotion category is drastically different, ranging from the smallest (disgust: 547) to largest (happy: 8989).

In implementation of our proposed hard label relaxation, we adopt the VGG-16[45] like stacked convolutional neural network architecture for its promising performance in classification tasks in general. As the small size of training dataset in FER2013 merely permits to train a small to medium size of convolutional neural network, we further tailored the large VGG-16 like model down to a tight eight layer structure. As our major focus is to compare the vanilla supervised trainings with our proposed label relaxation, it is essential to adopt the exact layer wise model architecture for both training methods. Moreover, as rendered in the foregoing section, the degree of τ has direct impact on the 'discreteness' of the label distribution that can further influence the performance of soft label training, therefore, we specify the degree of the temperature, e.g., τ into 1.0, 1.5, 2.0, 2.5, producing four variants of training types for model comparison. The layer wise configuration we employed in this experiment is shown in Table 3.1.

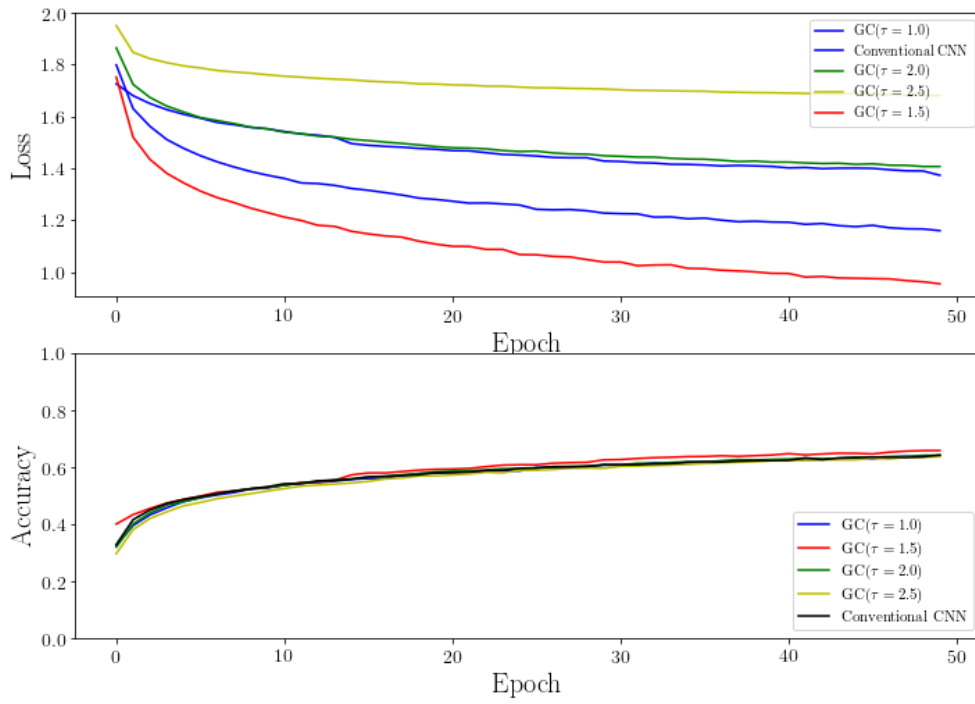
The results that render in Table 3.2 and Fig. 3.2, unequivocally elucidate that our proposed hard label relaxation outperformed the vanilla supervised training in this imbalanced pictorial sentimental classification task with no sacrifice on running time. Moreover, by comparison among four different variants specified by different values of τ , i.e., $\tau = 1.0, 1.5, 2.0, 2.5$, the largest performance gain was spotted at the intermediate level of τ , which refers as the middle ground between the complete discrete and complete continuous distribution of the label information. The detailed analysis on finding the optimal τ is reserved for future researches.

Table 3.1 Model Configurations.

Layer Index	Computation	Activation Types	Number of Trainable Parameters
1	Convolution with Filter Size (7x7)	PreLU	37664
2	Average Pooling with Filter Size (5x5)	N/A	0
3	Convolution with Filter Size (5x5)	PreLU	31264
4	Average Pooling with Filter Size (3x3)	N/A	0
5	Convolution with Filter Size (5x5)	PreLU	13854
6	Average Pooling with Filter Size (3x3)	N/A	0
7	Fully Connected with 1028 neuron	Linear	1185284
8	Fully Connected with 7 neuron	Linear	7203

Table 3.2 Model-Wise Discriminative Performance in FER2013 dataset

Model	Test Accuracy (%)	Run Time Per Epoch(s)
Vanilla CNN	62.28	291
G-C($\tau = 1.0$)	63.33	293
G-C($\tau = 1.5$)	64.16	294
G-C($\tau = 2.0$)	62.59	294
G-C($\tau = 2.5$)	62.55	298



In Fig. 3.3, we render out the loss and accuracy plots of our proposed approach. The plain neural network serves as the benchmark for the comparative purpose. As the batch size fixed at 100 samples, each epoch in this experiment is roughly equal with 280 iterations.

Fig. 3.2 Training in FER2013 dataset

3.4 Conclusion and Discussion

To lower the dependency on the vague and discrete labels that are commonly used in pictorial sentiment classification tasks, we propose a Gumbel-Concrete tuned label relaxation method that aims to learn partial and pseudo continuous affective representations in the supervised learning context. Our approach is parameter parsimonious, i.e., via tuning a single hyper-parameter τ , it allows the total control over the degree of discreteness in tuning the label distribution. Moreover, our training technique is fully compatible with all existing supervised learning architectures, and requires minimal to none architecture revisions. In FER2013 dataset, which is composed of vague labels, we observe clear performance improvement that brought by the implementation of our label relaxation technique with no sacrifices on the training speed.

However, this proposed label relaxation technique suffers from the following weakness: the heuristical determined τ , how to theoretical derive an optimal τ for each implementation remain as a challenge to be solved in future studies.

Chapter 4

Learning Multi-label Discrete Representations via Transfer Learning

4.1 Introduction

In the foregoing attempt, we suggest the usage of label relaxation technique to relax the original label distribution from one-hot expressions to real-numbered forms. However, in the proposed approach, i.e., the outputted labels are still one-hot encoded. In this research, we extend the single-label classification to the multi-label one. **Our objective here is to use transfer learning to solve a multi-label facial expression classification task.**

The researches on multi-label learning have been divided into two streams: problem transformation and algorithm adaptation, respectively [46]. The former approach allows a multi-label learning problem to degrade to a single-label one. Two widely applied problem transformation algorithms are binary relevance [47] and hierarchical of multi-label classifier, AKA., ML-ARAM [48]. The latter approach directly tackles the multi-label learning via the reconstructed loss function. Within this scope, the representative models are ranging from k-nearest neighbour related ML-KNN [49], to label relevance based multi-label neural networks [50].

Despite of the bulk of researches on multi-label learning in general, their applications on affective computing are rarely documented. To fulfill this research gap, Mower et al [51] [52] proposed a feature-agglomerate extraction method to encompass all appeared distinctive emotions in single prediction. Their approach coincides with the foregoing ML-ARAM model in ensuring the structured multi-label predictions. However, their claimed confidence rating – the computed Euclidean distance between input space and feature hyperplane – is mere an metric to index the importance of a feature. Another study that aimed in apply-

ing multi-label learning in affective classification relied on a novel regularisation to further penalise the max margin loss [53]. In spite of their claimed effectiveness in extracting multi-label affective features, the success of their proposed Group LASSO regulariser depended heavily on their manual and recursive feature extraction process.

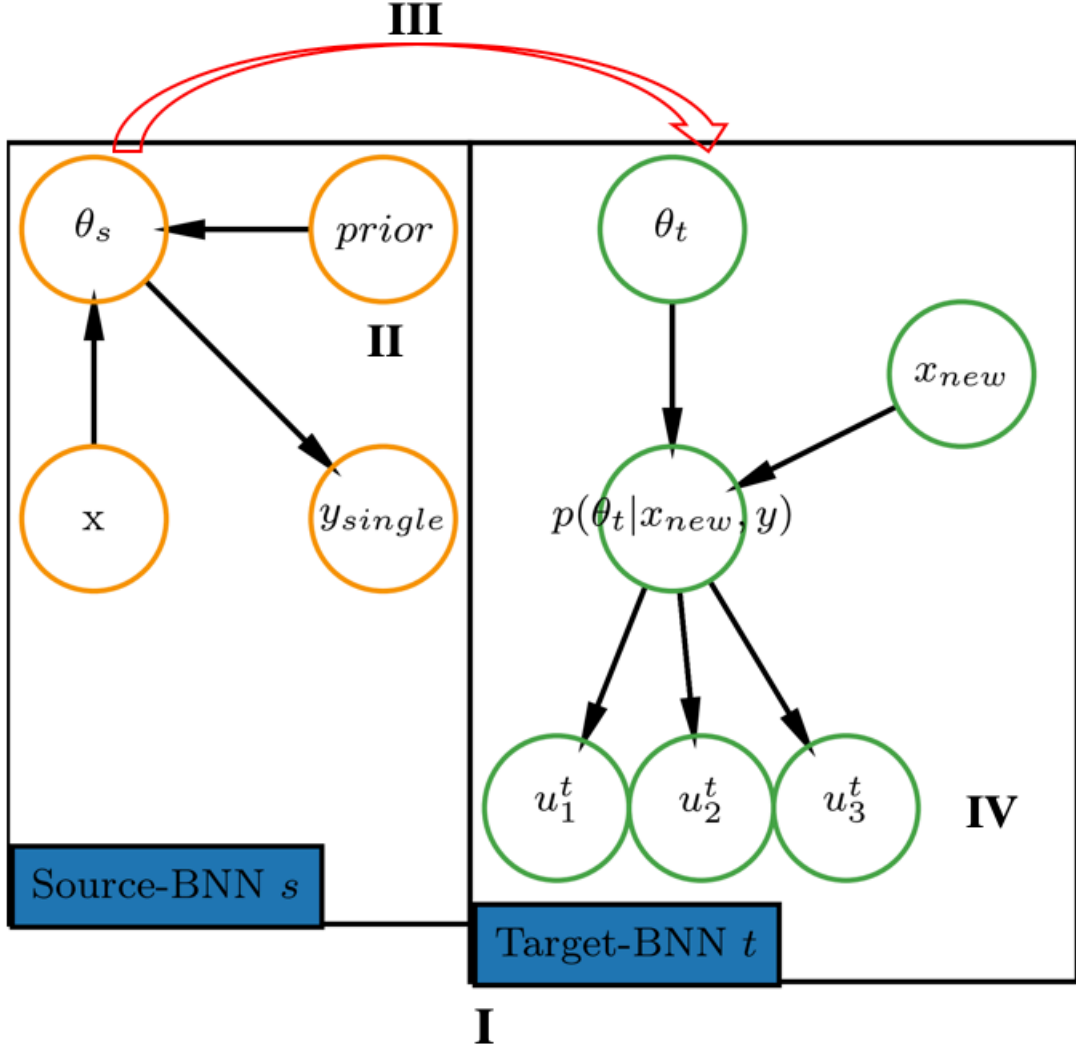
The difficulty that is posited in front of almost every multi-label learning algorithm is the scarcity of high quality multi-label annotated training dataset, i.e., it is laborious and expensive to collect the multi-label training dataset[54]. Hence, in combating with this noted issue, unlike conventional multi-label learning approaches, in this research we resort on one extreme form of transfer learning [55]: zero-shot transfer learning to allow the knowledge that is distilled from a source model, i.e., a single-labelled affective recognition task, to a target model on a more complex task such as a multi-label affective discriminative task without any forms of multi-label training.

To achieve our goal, we propose a learning framework, e.g., **Uncertainty Flow** framework with four components: two Bayesian neural networks, the usage of weakly informative priors, the transferred posterior weight distribution, and three final prediction indexes to output multi-label prediction on certain facial expression.

The remaining chapter is organised as following: we chiefly introduce the proposed **Uncertainty Flow** framework in sketch along with the detailed description of four core components. To demonstrate the effectiveness of our proposed **Uncertainty Flow** framework, we then present the empirical results from a large-scale comparative experiment. This large-scale experiment contains three levels of comparisons, i.e., the comparison among models, the comparison among different priors, and the comparison among three uncertainty indexes. The observed pronounced discriminability in solving a multi-label expression classification task empirically validates the effectiveness of the proposed **Uncertainty Flow** framework.

4.2 Uncertainty Flow Framework

Sketched in Fig.4.1, the proposed **Uncertainty Flow** framework is consisted of four components, i.e., two Bayesian neural networks, the weakly informative priors, the transferred posterior weight distribution, and three prediction indexes. The work pipeline of **Uncertainty Flow** initiates at standard supervised training of a source Bayesian neural network(BNN) with a weakly informative prior, e.g., uniform or Cauchy prior, follows the computation of the weight posterior in preparation of model uncertainty in the source BNN, then this distilled model uncertainty is transferred to a target BNN, which is specialised in outputting multi-label predictions. Finally, three prediction indexes are introduced to output multi-label prediction on certain facial expression.



The graphic model explanation of our proposed **Uncertainty Flow** framework shows four essential elements. I.e., the dual Bayesian neural networks in **I**, e.g., a source and a target BNN (separate by different colours in this figure); the weakly informative prior in **II**; the transferred posterior weight distribution in **III**; and three proposed uncertainty indexes in perfecting the final multi-label categorisation in **IV**.

Fig. 4.1 Uncertainty Flow Framework

4.2.1 Two Bayesian Neural Networks

The core part of proposed **Uncertainty Flow** is the usage of two Bayesian neural networks (aka., BNNs) to serve a source and target task respectively. A Bayesian neural network can be seen as a neural network with a pre-defined prior distribution on its weights [35]. Under the classification set-up, where we have sets of input and output variables, e.g., $\{x, y\}$, the two BNNs for source and target task respectively should be specified into following:

- The source BNN

$$\theta_s \sim \text{Normal}(0, 1) \quad (4.1)$$

$$NN(x; \theta_s) = \text{sigmoid} \circ \left(\tanh \circ \sum_{i=1}^n (x \cdot \theta_s) \right) \quad (4.2)$$

$$y_{single} \sim \text{Categorical}(y_{single} | NN(x; \theta_s), p). \quad (4.3)$$

- The target BNN

$$NN(x; \theta_t) = \text{sigmoid} \circ \left(\tanh \circ \sum_{i=1}^n (x \cdot \theta_t) \right) \quad (4.4)$$

$$y_{multi} \sim \text{Normal}(y_{multi} | NN(x; \theta_t), \sigma^2). \quad (4.5)$$

Where θ_s and θ_t indicate the weights in the source and target BNNs respectively, and y_{single} and y_{multi} indicate different output predictions (single-label or multi-label predictions) from the source and target BNNs. For simplicity, only *tanh* non-linearity is considered as the activation function used in both source and target BNNs, and p in the source BNN is set to number of class in training dataset. We assume σ^2 is a known variance.

Notice here, we tailor the target BNN in concord with multi-label prediction task demand. I.e., to allow a Bayesian neural network to produce multiple outputs, the prediction distribution is no longer categorical, but rather in the source BNN is replaced with a real-value function, e.g., sigmoid function, in the target BNN. Also, as no form of training taken place in the target BNN, there is no defined prior distribution associated with θ_t .

4.2.2 Weakly Informative Priors

The prior, which determines the first and second order statistics of model parameters, is de facto the driving force in bayesian learning. Hence, the proper specification of a model ties

closely with the choice of an applicable prior for a given task. Unfortunately, the majority works on Bayesian learning pay overwhelmed attention towards the prior that are informative and conjugate for their analytical convenience, the family of uninformative and weakly informative priors had been largely ignored.

Argued in [56] [57], differ to the conventional used informative prior, the usage of weakly informative prior, i.e., a semi-flat prior, yielded superior predicative performance in single-label discrimination. Therefore, it is rational to extend this finding in multi-label learning. The formal definitions of informative and weakly informative priors are rendered below, and their corresponding probabilistic density curves are plotted in Fig.4.2.

- Informative Prior

- Normal Prior

$$\theta_N \sim N(\mu, \sigma^2)$$

$$p(\theta_N) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(\theta - \mu)^2\right)$$

- Weakly Informative Prior

- Uniform Prior

$$\theta_U \sim U(\alpha, \beta)$$

$$p(\theta_U) = \frac{1}{\beta - \alpha}$$

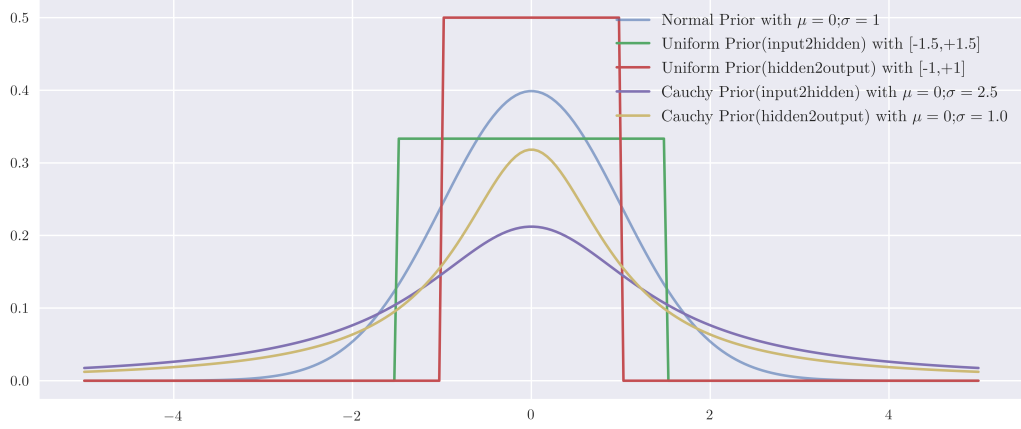
- Cauchy Prior

$$\theta_C \sim Cauchy(\alpha, \beta)$$

$$p(\theta_C) = \frac{1}{\pi\beta(1 + (\frac{x-\alpha}{\beta})^2)}$$

4.2.3 Transferred Posterior Weight Distribution

Armed with the previous defined source BNN and different kinds of prior, e.g., informative or weakly informative, the next step is to infer the posterior distribution for θ_s , i.e., $p(\theta_s | x, y_{single})$. Here, we rely on the commonly used variational inference to infer this statis-



This graph demonstrates the probabilistic density curve of each prior used in this research, i.e., Normal, Uniform and Cauchy priors. Here, the further specification of input-to-hidden, hidden-to-hidden priors are defined as the hierarchical shrinkage in variance of the corresponding priors.

Fig. 4.2 The Probabilistic Density Curves of Normal, Uniform, and Cauchy Priors

tic via minimising the KL-divergence [58][59]¹ between approximated variational distribution, i.e., $q(\theta_s)$ and true posterior, i.e., $p(\theta_s|x, y_{single})$.

To construct the variational distribution to approximate the intractable true posterior, we resort on mean-field Normal distribution, which is parameterised by the mean and standard deviation, e.g., $q_\phi(\theta_s) = \text{Normal}(\mu, \sigma^2)$.

As we cannot compute the KL-divergence directly, the common approach is to resort on optimising an alternative objective, i.e., maximising the ELBO (evidence lower bound) \mathcal{L} as

$$\mathcal{L} = \mathcal{E}_{q_\phi}[\log(p(y_{single}|x, \theta_s)p(\theta_s)) - \log q_\phi(\theta_s)]. \quad (4.6)$$

We then optimise this ELBO (\mathcal{L}) by ADVI (automatic differentiation variational inference) algorithm in [60] to yield the optimal $q^*(\theta_s)$ to approximate the posterior $p(\theta_s|x, y_{single})$, i.e., $q^*(\theta_s) \approx p(\theta_s|x, y_{single})$.

This approximated posterior weight distribution is then served as the knowledge to transfer from the source BNN to the target BNN, i.e., $p(\theta_t|x, y_{multi}) \leftarrow q^*(\theta_s)$. This step of

¹

$$KL(q||p) = \sum_{\theta} q(\theta_s) \log \frac{q(\theta_s)}{p(\theta_s|x, y_{single})}$$

knowledge transfer allows the target BNN to obtain posterior weight distribution without going through the standard supervised training procedure.

4.2.4 Prediction Related Uncertainty Indexes

Armed with the transferred posterior weight distribution, i.e., $p(\theta_t|x, y_{multi})$ and the defined target BNN from Eq. 4.4 to Eq. 4.5, we are now positioned to render out the posterior predictive distribution, e.g., $p(x_{new}|x, \theta_t)$ for any incoming input (in our case, the new facial expression image, denoting as x_{new}). The making of posterior predictive distribution can be approximated by the usage of Monte Carlo estimation as:

$$p(x_{new}|x) \approx \frac{1}{M} \sum_{i=1}^M p(y|x, \theta_i) \quad (4.7)$$

$$\theta_t \sim p(\theta_t|x, y_{multi}). \quad (4.8)$$

In actual experiment, we set the number of Monte Carlo estimation to $M = 500$ for quick sampling.

Obtained this posterior predictive distribution, we render out the following three multi-label prediction indexes. The first index is the mean of the previous yielded predictive distribution, e.g., denoting as *soft-max* index (u_1^t):

$$u_1^t = \mathbb{E}_{p(x_{new}|x)} \approx \frac{1}{M} \sum_{i=1}^M p(y|x, \theta_i). \quad (4.9)$$

Importantly, as our final goal is to output multiple discrete labels (2 labels) for certain facial expressions, posterior to the computation on *soft-max* u_1^t index of each new input on number of class entry. I.e., if there are 7 classes associated with the new input, we compute the *soft-max* index on each of 7 classes. This type of probabilistic index tells us the most probable outputs given the input, which is similar to the softmax probability that used producing single-label prediction in non-Bayesian neural networks [61] [62]. For the multi-label prediction, **we then refer two classes with the highest index and the second highest index to 1 and other classes as 0.**

From Bayesian learning perspective, the above-mentioned *soft-max* index reflects the belief of applying predictive mean in indexing the prediction uncertainty. Numerically, this type of uncertainty index captures the mere classification type in a multi-label classification task, is equivalent with the class type probability in non-Bayesian neural networks. However,

as the predictive mean does not capture the full picture of parameter posterior distribution, we draw our attention towards the predictive variance instead.

We argue that the yielded predictive variance reflects the degree of uncertainty that is associated with each prediction. As a result, based on the approximated weight posterior, i.e., $p(\theta|x)$, a better prediction index is expressed below in Eq.4.10:

$$u_2^t = Var[p(x_{new}|x)] \approx Var\left[\frac{1}{M} \sum_{i=1}^M p(y|x, \theta_i)\right] \quad (4.10)$$

We denote this measure of prediction uncertainty index as *pure* (u_2^t) index for multi-label prediction. Different from the previous *soft-max* index, in Pure index, each prediction score reflects the variance of the prediction, which should be negatively correlate with our certainty in prediction. I.e., the larger the variance, less certain the prediction. **Hence, we refer two classes with the least index and the second least index to 1 for multi-label prediction and other classes as 0.**

One step further, the two foregoing indexes can be combined altogether, which allows the prediction index to reflect both class type probabilistic prediction (*soft-max* prediction index) and the certainty associated index (*pure* prediction index). We denote this prediction index as *uncertainty plus* (u_3^t). This prediction index is computed as follows:

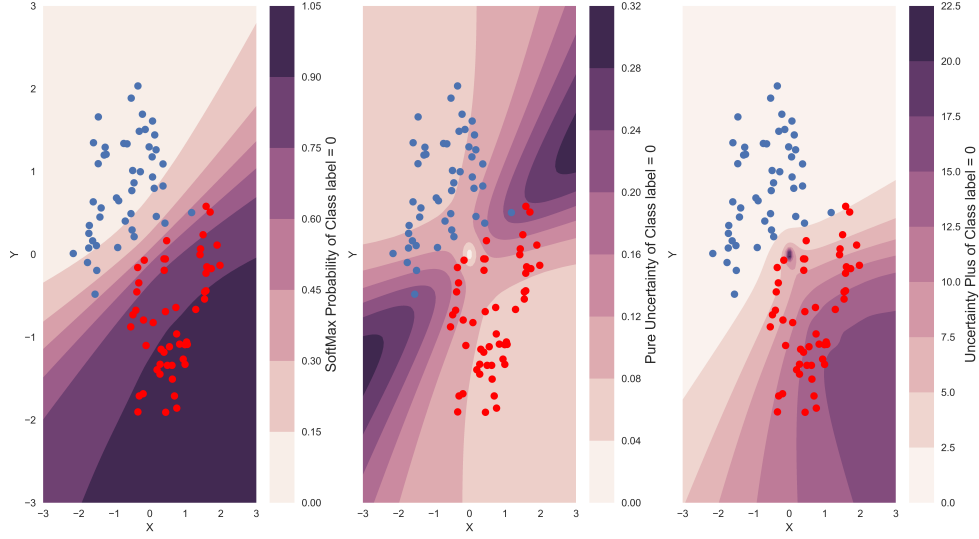
$$u_3^t = \frac{\frac{1}{M} \sum_{i=1}^M p(y|x, \theta_i)}{Var\left[\frac{1}{M} \sum_{i=1}^M p(y|x, \theta_i)\right]} \quad (4.11)$$

In production of this index, each class type probabilistic prediction is proportionately adjusted to its associated certainty. I.e., the less certain class type predictions should be down weighted in comparison to the more certain prediction. **For the final multi-label prediction, we refer the highest and second highest classes in *uncertainty plus* index as the correct labels, e.g., as 1, whereas the rest classes are all converted to 0.**

For illustrative purposes, how each of three above mentioned uncertainty indexes, e.g., **soft-max uncertainty**, **pure uncertainty**, and **uncertainty plus**, influences on a simple binary classifier, is demonstrated in Fig.4.3.

4.3 Empirical Experiment

Relying on the transferred model uncertainty, the proposed **Uncertainty Flow** framework allows a learner to output multi-label predictions under single-label training curriculum. Empirical validation of our proposed framework contains two enquiries that need to be ad-



In this figure, three different means of crafting uncertainty boundary for obtaining classification prediction on a simple binary classification problem, i.e., two classes are separated by blue and red, is delineated.

Fig. 4.3 Comparison of Three Prediction Related Uncertainty Indexes on a Binary Classifier

addressed. I.e., one is to investigate that whether or not our proposed **Uncertainty Flow** is superior to conventional multi-label learning algorithms in facilitating the zero-shot multi-label learning; whereas the other is to see which uncertainty index leads to the most significant performance elevation. Also, in order to investigate the role of suggested weakly informative priors, we additionally specify our **Uncertainty Flow** into three types of priors. Hence, in total, there are three-level comparisons in our experiment, i.e., *model comparison*; *prior comparison*; and *uncertainty comparison*.

4.3.1 Dataset

Training Dataset

We selected the first 1500 images from FER2013 [63] as our training dataset. The reason for intentionally lowering the size of the training dataset is to enforce the similar model complexity between a source BNN and a target BNN. FER2013 is a well-researched public dataset, which is comprised of facial expression images for pictorial sentiment discrimination. Prior to our implementation, all images in this truncated version of FER2013 had gone through the standard preprocessing process, e.g., fixate the faces at centre, standardise the image size

Name	# of Instances	# of Labels	Cardinality[46]	Source
Training	1500	7	1.0	[63]
Testing	200	7	1.89	This Research

Table 4.1 Descriptive Statistics of training and testing dataset

to 48 by 48 pixels in resolution, and all faces are properly registered. We then normalised the pixel values of input images. The original FER2013 images are labelled as one of seven emotion categories, e.g., angry, disgust, fear, happy, sad, surprise, and neutral.

Testing Dataset

To allow the evaluation of outputted multi-label predictions, it is imperative to rely on some existing benchmark annotations. Unfortunately, there is no current reliable multi-label annotations for FER2013 facial expressions. For this, we conducted a small-scale, i.e., 200 images, experiment on manual annotating the multi-label version of FER2013. We invite five annotators to annotate two discrete labels associated with each of 200 facial expression images.

After annotation process, we take the majority voting on the two mostly annotated labels for each expression. In terms of inter-rater reliability, the computed medium Fleiss-Kappa coefficient index [64], e.g., 0.25, ² suggests reliable annotations among different annotators. The descriptive statistics of this annotated multi-label testing dataset is summarised in Table 4.1.

4.3.2 Models

To conduct an experiment that contains above-mentioned three-level comparisons, i.e., *model comparison*, *prior comparison*, *uncertainty comparison*, it demands explicit specification of all models in current experiment. In *model comparison*, four widely used multi-label learning algorithms, ranging from adaption algorithms, e.g., Multi-Label K-means Nearest Neighbour (MLkNN), Multi-label Neurofuzzy Classifier (ML-ARAM), to problem transformation algorithms, e.g., Binary Relevance (BR) and Label Powerset (LP), are included. In addition, two multi-label compatible neural networks. i.e., a multi-label feedforward Neural network (ML-FNN) and a multi-label convolutional neural network (ML-CNN), are also included in *model comparison* comparison.

In *prior comparison* and *uncertainty comparison*, depending on the prior type, i.e., informative or weakly informative, and different prediction related uncertainty indexes, e.g.,

²between -1 to 1 , higher is more reliable

soft-max uncertainty, **pure uncertainty**, **uncertainty plus**, the **Uncertainty Flow** generates 9 variants, denoting as BNN-normal-soft; BNN-normal-pure; BNN-normal-plus; BNN-uniform-soft; BNN-uniform-pure; BNN-uniform-plus; BNN-cauchy-soft; BNN-cauchy-pure; BNN-cauchy-plus. To further elevate the discriminative performance in multi-label prediction, we additional frame a convolutional neural network under Bayesian learning, producing Bayesian convolutional neural network within the proposed **Uncertainty Flow** framework, with its associated 9 variants, i.e., BCNN-normal-soft; BCNN-normal-pure; BCNN-normal-plus; BCNN-uniform-soft; BCNN-uniform-pure; BCNN-uniform-plus; BCNN-cauchy-soft; BCNN-cauchy-pure; BCNN-cauchy-plus. The configurations of above-mentioned models are summarised as the following itemised format.

1. ML-kNN

The number of k mixture components was set up to 4, and the default smoothing parameter was tuned at 0.

2. ML-ARAM

The vigilance was set to 0.9 to reflect the high dataset dependence, the threshold was set to 0.02 in line with the original algorithm implementation[48].

3. Binary Relevance

Base classifier: SVC(support vector classifier)

4. Label Powerset

Base classifier: Naive Gaussian classifier

5. ML-FNN Layer-wise Architecture:

Dense(128) -> Dropout(p = 0.2) -> Dense(128) -> Dropout (p = 0.2) -> Dense(Output)
3

Epoch:50(1500 iterations)

6. ML-CNN Layer-Wise Architecture:

Convolution(3*3) -> Convolution(3*3)- > Max Pooling(2*2) -> Dropout(p= 0.2) -> Dense(128) -> Dense(Output)

Epochs: 50(1500 iterations)

³This notation indicates the information pathway from a dense connected layer with 128 units, to the final dense connected layer via intermediate dense connected and dropout layers

7. BNN Layer-Wise Architecture: Same as *ML-FNN*

Priors: Normal/Uniform/Cauchy

Inference Method: Variational Mean Field

Number of Posterior Sampling: 500

8. BCNN Layer-Wise Architecture: Same as *ML-CNN*

Priors: Normal/Uniform/Cauchy

Inference Method: Variational Mean Field

Number of Posterior Sampling: 500

4.3.3 Evaluation Metrics

Different than the uniformed metric that used in single-label classification, i.e., classification accuracy, diversified evaluation metrics have been proposed. In line with the rough classification from [46], we adhere the dichotomy classification of the evaluation metrics as bipartition and ranking based. For illustrative purposes, assuming a multi-label evaluation dataset is consist of input, i.e., x_i , and the set of true labels, i.e., Y_i , where $i = 1...m$ and $Y_i \subseteq L$, L is the set of all correct labels. Under this notation, the set of predicated labels are denoted as Z_i , where $i = 1...m$, while the rank predicted by learning method for a label λ is denoted as $r_i(\lambda)$. The most relevant label, receives the highest rank (1), while the least relevant one, receives the lowest rank (q).

Bipartition based

Delegated from the single-label metric, bipartition based metrics are proposed to capture the differences between actual and predicted sets of labels over all evaluation dataset. These differences can be computed in various means via either averaged over all samples or all label sets.

1. Hamming loss

$$Hamming - Loss = \frac{1}{m} \sum_{i=1}^{m_i} \frac{|Y_i \triangle Z_i|}{M}$$

Where \triangle represents the symmetric difference of two sets, i.e., predicted and true label sets. Contrast with other over-strict measures of multi-label classification accuracy, i.e., low tolerance on partial label misclassification, e.g., $\frac{1}{m} \sum_{i=1}^m I[Y_i = Z_i]$, the hamming

loss, which sums up to 1, offers a mild criteria for wider range of measurement application.

2. Micro-Averaged F-Score & Average Precision

Inherited from classic binary evaluation in information retrieval tasks, F-score and average precision, which both reflect their corresponded combinations of averaging over precision and recall, are two readily applicable metrics in multi-label learning. Among various averaging operations, e.g., macro, weighted, and micro, the preferred operation is micro-average as it offers each sample-class pair an equal contribution to the overall metric. Consider a binary evaluation measure t_p, t_n, f_p, f_n that is computed via the number of true positives t_p , true negatives t_n , false positives f_p , false negatives f_n , the n th threshold for precision and recall are P_n and R_n , the interested micro-averaged F-score and average precision score(AP) are derived as following:

$$P_n = \frac{t_p}{t_p + f_p}$$

$$R_n = \frac{t_p}{t_p + f_n}$$

$$MicroAveraged(F_\beta) = (1 + \beta^2 \frac{P_n \times R_n}{\beta^2 P_n + R_n})$$

$$AveragePrecision(AP) = \sum_n (R_n - R_{n-1}) P_n$$

Ranking Based

1. Converge

To measure the needed distance to cover all true label sets, i.e., Y_i in the predicted label sets, we resort on the converge error metric. It can be defined as following:

$$Coverage - Error = \frac{1}{m} \sum_{i=1}^m \max_{\lambda \in Y_i} r_i(\lambda) - 1$$

2. Ranking Loss

The ranking loss targets at the incorrect ordering of the predicted label sets. Presume \bar{Y}_i is expressed as the complementary set of Y_i , its computation can be defined as fol-

lowing:

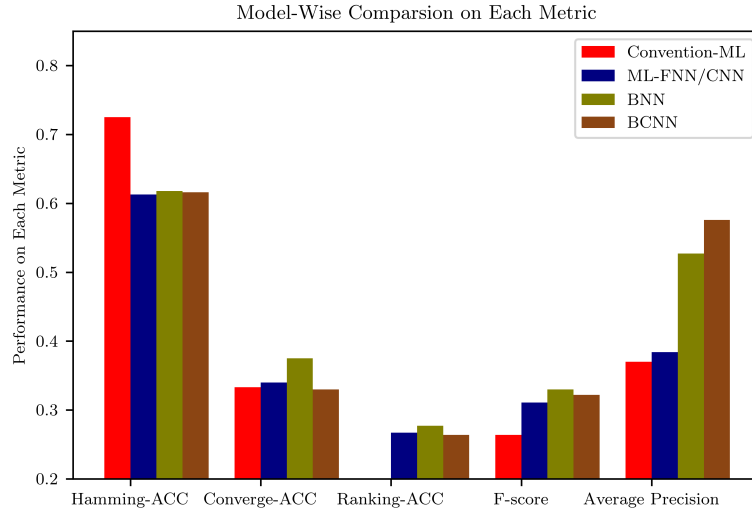
$$R - Loss = \frac{1}{m} \sum_{i=1}^m \frac{1}{|Y_i| |\bar{Y}_i|} \{(\lambda_a, \lambda_b) : r_i(\lambda_a) > r_i(\lambda_b), (\lambda_a, \lambda_b) \in Y_i \times \bar{Y}_i\}$$

4.3.4 Results & Discussion

The overall result of our conducted large-scale comparative experiment is chiefly presented in Table 4.2. For illustrative purposes, we grouped the results to highlight the comparison among different models. As we used various of evaluation metrics to assess the performance of corresponding models, it is difficulty to obtain a clear judgement that is based on single metric. I.e., when we pitted our approach, i.e., **Uncertainty Flow** against the MLkNN approach in conventional multi-label models, our approach, including all nine variations, is inferior to the MLkNN approach on the metric of *Hamming-Loss*. However, when we accessed the model according to its performance on *Average Precision*, our approach largely outperformed the MLkNN approach. Moreover, as we incorporated nine variations in **Uncertainty Flow**, the in-depth analyses of the prior types and uncertainty indexes are demanded. We then divided our discussion of the overall result into three parts, e.g., the results on model comparison, the results on prior comparison, and the results on uncertainty comparison.

Table 4.2 Model Comparison in Various Multi-label Evaluation Metrics

Candidate Models	Hamming-Loss	Converge-Loss	Ranking-Loss	F-Score	Average Precision	Source
Conventional Multi-Label Models						
MLkNN	0.286	7.000	0.950	0.090	0.346	[49]
ML-ARAM	0.374	6.670	0.801	0.264	0.369	[48]
Binary Relevance	0.275	7.00	1.000	0.263	0.340	[65]
Label Powerset	0.328	6.940	0.837	0.215	0.370	[66]
Multi-Label Compatible Neural Networks						
ML-FNN	0.402	6.700	0.761	0.282	0.376	[67]
ML-CNN	0.387	6.600	0.733	0.3108	0.384	[68]
Uncertainty Flow - Bayesian Neural Networks						
BNN-normal-soft	0.404	6.500	0.75	0.279	0.360	This research
BNN-normal-pure	0.382	6.500	0.723	0.318	0.389	This research
BNN-normal-plus	0.402	6.673	0.759	0.282	0.525	This research
BNN-uniform-soft	0.414	6.750	0.7816	0.302	0.353	This research
BNN-uniform-pure	0.385	6.450	0.723	0.330	0.389	This research
BNN-uniform-plus	0.404	6.450	0.765	0.310	0.530	This research
BNN-cauchy-soft	0.400	6.7	0.759	0.285	0.378	This research
BNN-cauchy-pure	0.382	6.525	0.727	0.312	0.402	This research
BNN-cauchy-plus	0.401	6.250	0.741	0.290	0.527	This research
Uncertainty Flow - Bayesian Convolutional Neural Networks						
BCNN-normal-soft	0.421	6.750	0.791	0.250	0.385	This research
BCNN-normal-pure	0.384	6.700	0.737	0.319	0.449	This research
BCNN-normal-plus	0.400	6.675	0.751	0.288	0.561	This research
BCNN-uniform-soft	0.403	6.675	0.762	0.282	0.421	This research
BCNN-uniform-pure	0.404	6.800	0.769	0.279	0.311	This research
BCNN-uniform-plus	0.387	6.725	0.736	0.310	0.527	This research
BCNN-cauchy-soft	0.401	6.675	0.760	0.285	0.416	This research
BCNN-cauchy-pure	0.396	6.800	0.753	0.322	0.390	This research
BCNN-cauchy-plus	0.401	6.75	0.758	0.285	0.576	This research



Note here, for illustrative purposes, we used Hamming-ACC, Converge-ACC and Ranking-ACC instead of original loss based metrics. Rather than the averaging over the models in each category, e.g., Conventional-ML, we picked the most representative model in each category for different metric.

Fig. 4.4 Model Wise Comparison on Multi-learning Metrics

Model Comparison

Ruling out the factors of prior types and prediction related uncertainty indexes, the empirical comparison between **Uncertainty Flow** framework and the alternatives demonstrated mixed results, illustrated in Fig.4.4.

Previous findings from [46] and [54] stated that it is practical difficult to observe a single model or algorithm, which is competitive enough to beat others in every multi-label evaluation metric. Hence, it is imperative to investigate each loss metric independently. Focusing on *Hamming-Loss*⁴, interestingly, the conventional multi-label learning models are particular good in minimising this type of loss. However, indicated in *Converge-Loss* and *Ranking-Loss*, both **Uncertainty Flow** and multi-label compatible neural networks, e.g., ML-FNN and ML-CNN are superior than the conventional multi-label learning alternatives.

Under two precision related metrics, e.g., *F-score* and *average precision*, with the help from a weakly informative prior, e.g., uniform or Cauchy prior, and an advanced prediction related uncertainty indexes, e.g., **pure uncertainty**, or **uncertainty plus**, both BNN and BCNN, exhibited clear performance advantage over their alternatives, e.g., ML-FNN, ML-CNN and conventional multi-label models. Especially on the metric of *average Precision*,

⁴Hamming-ACC = 1- Hamming loss

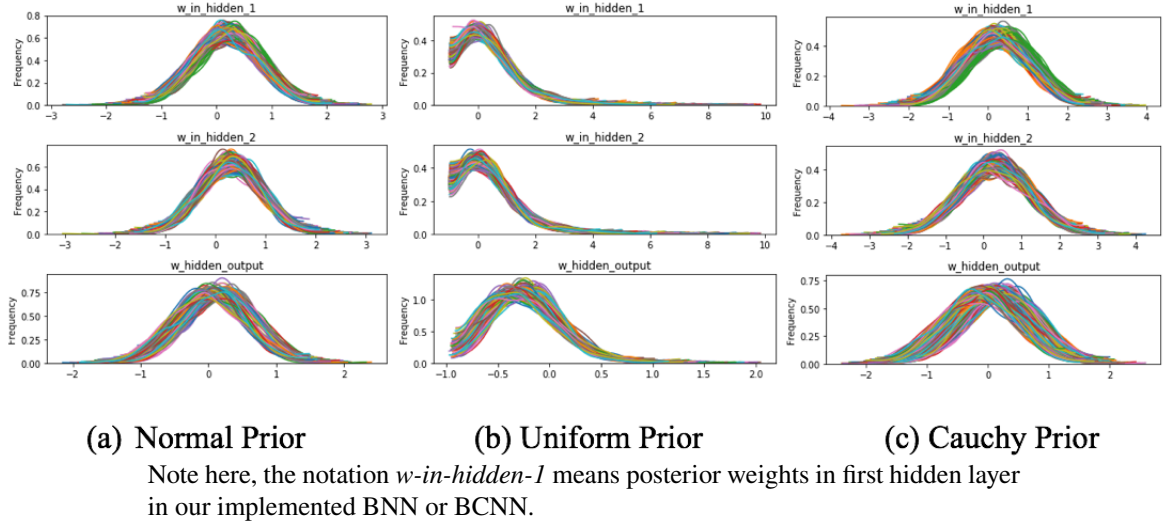


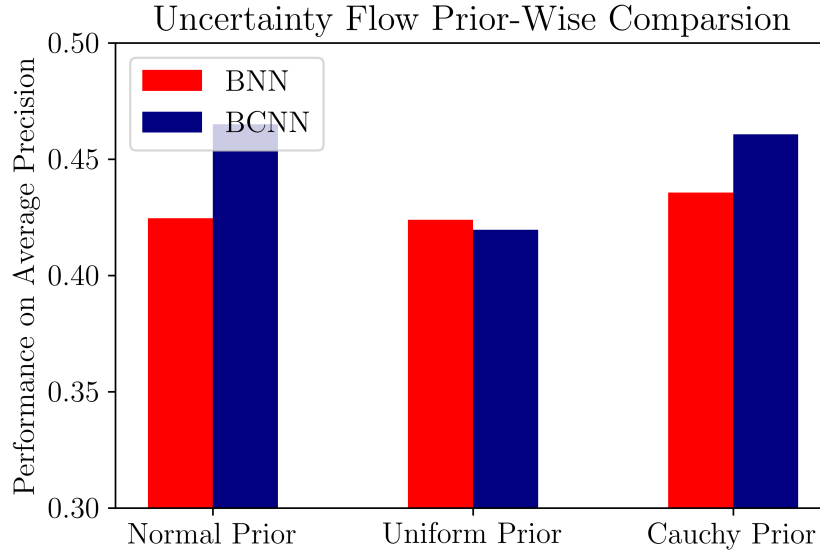
Fig. 4.5 Different priors on posterior weights under the uncertainty flow framework

the nontrivial performance enhancement, i.e., over 20% accuracy increase, demonstrated the superior discriminability that tags to our proposed **Uncertainty Flow**. Moreover, comparing the performance between BNN and BCNN, the convolutional architecture, e.g., BCNN, should be credited for overall performance improvement.

Prior Comparison

To verify the most applicable prior in our proposed **Uncertainty Flow**, the prior comparison among three candidate priors is worthy to be fully investigated. Shown in Fig.4.5 (b), despite some similarities in shapes, it is clear that each prior has its unique effect in shaping the corresponded posterior distribution of the weights. In specific, the effect of uniform prior on posterior weight is seen as the restriction on the approximated posterior weights, i.e., the posterior weights have to be higher than a fixed value, e.g., 1 in our implementation. This restriction effect may lower the discriminability of the uniform prior imposed model, shown in Fig.4.6. Interestingly, the posterior distribution of weights from imposed normal and Cauchy priors respectively rendered nearly identical distribution shape, shown in Fig.4.5 (a) and (c). The minute difference between these two is the enlarged variance for Cauchy prior induced posterior distribution of weights. Despite seemingly trivial, this difference in variance lead to the discrepancy in discriminative performance, shown in Fig.4.6. Overall, based on final induced discriminability, a Cauchy prior is considered as the most applicable prior in our proposed **Uncertainty Flow**.

The performance enhancement that can be reflected by above-mentioned 'clustered' effect in weight posterior was observed in examination of the discriminability of three imple-



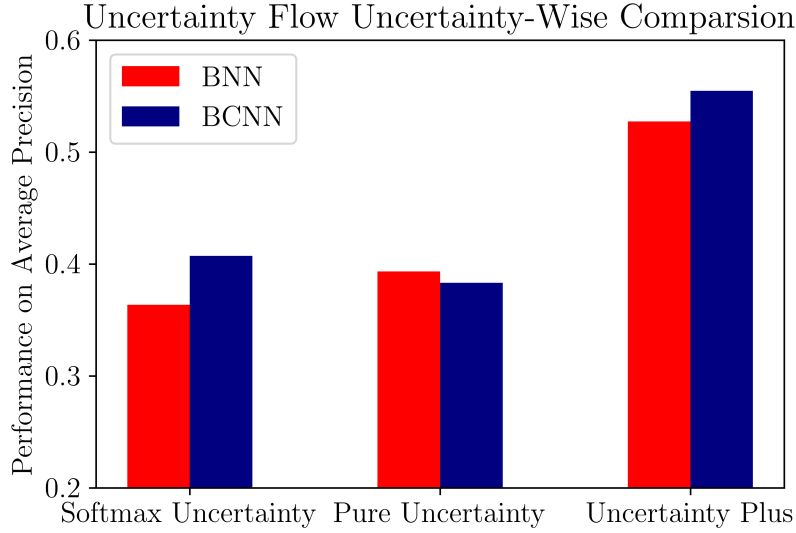
Note here, we rule out the impacts of uncertainty indexes via averaging the performance of models with same implemented prior. Average precision is chosen as other metrics failed to render clear discriminability comparison .

Fig. 4.6 Prior induced discriminability differences in the uncertainty flow framework

mented prior. Plotted in Fig.4.6, focusing on the *average precision* evaluation metric, regardless of the variations in Bayesian neural networks, i.e., BNN or BCNN, the employment of Cauchy prior - as one kind of weakly informative prior - leaded competitive multi-label affective classification. However, as another implemented weakly informative prior, uniform prior was inferior to the used informative prior, e.g., normal prior. This observed attenuation in discriminability from uniform prior may due to the its above-mentioned spike-and-slab effect on weight posterior that requires extra training epochs to stabilise the pre-to-posterior inference.

Uncertainty Comparison

Undoubtedly, the most pronounced performance improvement is pertaining to the inclusion of advanced prediction related uncertainty indexes, e.g., **pure uncertainty** and **uncertainty plus**. To recall the foregoing definition of prediction related uncertainty indexes, the **soft-max uncertainty** is a mere indication of multi-class prediction type, which is equivalent with the predictions in non-Bayesian alternatives. The pure uncertainty, i.e., on the contrary - depends heavily on weight posterior - can be produced exclusively in our proposed **Uncertainty Flow** framework. Reflected in Fig.4.7, when the feedforward architecture was adopted, **soft-max uncertainty** is inferior to **pure uncertainty** in producing multi-label pre-



Note here, the effects of priors were marginalised via prior-wise averaging.

Fig. 4.7 Discriminative Performance Across Different Prediction Related Uncertainty Indexes

diction. Interestingly, when the convolutional architecture was chosen, it uncovered a different story, i.e., the discriminability from **pure uncertainty** became inferior to **soft-max uncertainty**.

Not surprisingly, the combination of **soft-max uncertainty** and **pure uncertainty**, i.e., the craft of **uncertainty plus**, allows a set of multi-label predictions to be tuned based on its uncertainty value. Shown in Fig.4.7, it is clear that the crafted predictions that are benefited from **Uncertainty plus** are superior to other two uncertainty indexes. I.e., its introduced improvement in *average precision* is over 20% compare to other two indexes. Combining the most applicable weakly informative prior and the advanced uncertainty index together, the two most efficient variants in feedforward and convolutional architectures are **BNN-cauchy-plus** and **BCNN-cauchy-plus**, respectively. We leave sensitivity analysis of our proposed advanced uncertainty indexes to future research.

4.4 Conclusion & Discussion

To move forward on our pursuit of learning continuous affective representation, in this research, we aim to lower the reliance on overwhelming single-label annotations in pictorial affective analysis. We wish a target system (computational model) can output multiple labels on single evoked source. However, the scarcity of large-scale high-quality pictorial posits a

challenge for direct deploying some of conventional multi-label learning algorithm here. As a results, we propose a novel zero-shot inductive transfer learning framework: **Uncertainty Flow** to tackle the prior noted issue.

Under this pioneer framework, we argue that the model uncertainty can be distilled from a source single-label recognition task. The distilled knowledge is then fed to a to-be-learned multi-label affective recognition task. For predictions, three types of uncertainty indexes, i.e., **soft-max uncertainty**, **pure uncertainty**, and **uncertainty plus**, are further proposed. For empirical validation, the authors conducted a large-scale comparative experiment on the manual annotated multi-label FER2013 dataset across three levels of comparisons, i.e., *model comparison*, *prior comparison*, and *uncertainty comparison*. The observed performance superiority in **Uncertainty Flow** unequivocally renders the feasibility of applying this framework in zero-shot multi label affective learning.

However, even under the permitted computational resources, to run a full Bayesian posterior remains as a daunting task. How to speed up the posterior inference remains as an open research question. Asides from the issue of prolonged training speed, the proposed **Uncertainty Flow** framework suffers the same problem as our initial attempt, i.e., **Label Relaxation**. Irrespective of their to-be-solved different learning problems, the learned representations from two prior attempts are still relied heavily on the quality of included discrete labels. I.e., this sets up a limit for these encoded pictorial affective representations, which their expressiveness is largely restricted with the label. Therefore, these learned affective representations can not be treated as authentic continuous representations that allows dimensional representation of human facial expression. Moving forward on our journal of complex continuous affective representation learning, we introduce our final tryout: the proposal of **encapsulated variational auto-encoder** in learning continuous, affective aware representation of facial expressions.

Chapter 5

Generating Continuous Representations via EVAE

5.1 Introduction

Previous label relaxation technique (cf. Chapter 3) and uncertainty flow framework (cf. Chapter 4) allow us to learn relaxed single label and multi-label representations of facial expressions. However, these learned representations quickly reached their bottlenecks to reflect the richness of sentiments [69]. I.e., emotions and sentiments in wild are not restricted in certain fixed taxonomy, e.g., the basic six emotions[70]. In contrast, human facial expressions should be represented as granularity at continuous scales. A recent neurological finding suggests human emotion may represent in distributed dimensional manner [71]. Hence, in this research, **our objective is to generate facial expressions along continuous scales.**

To achieve our goal, we rely on the generative model to generate expressions continuously. Among the widely used generative models, we choose the variational auto-encoder (VAE) [38]. Unfortunately, the mere usage of a VAE in generating continuous expressions is suboptimal due to the following issue: The generated expressions do not exhibit clear expression transition patterns that can be corresponded to certain psychological conceptualised continuous axes, such as arousal and valence axes in [23].

To this end, we propose a novel form of variational auto-encoder: **encapsulated variational auto-encoders (EVAE)** that is tailored specifically for generating continuous facial expressions. To validate our approach, two empirical experiments on Frey faces and FERF-DB datasets were introduced to access the generative performance.

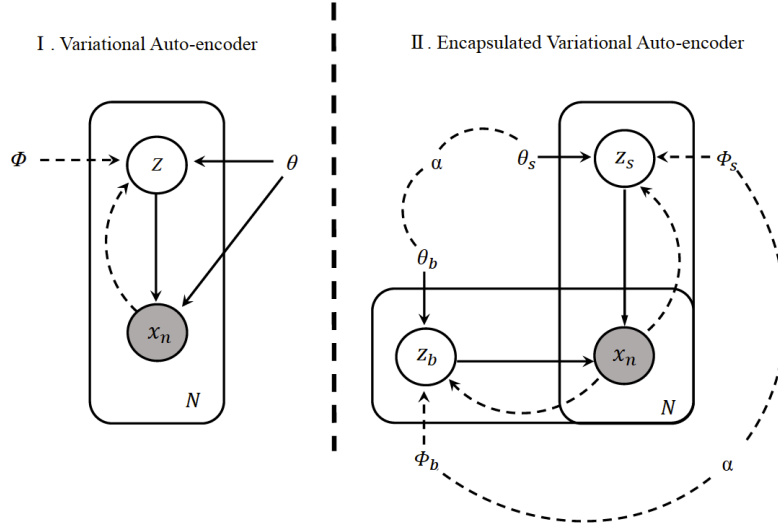


Fig. 5.1 Graphical models of (i) a conventional variational auto-encoder and (ii) our derived encapsulated variational auto-encoder. Solid arrows denote probabilistic decoders, whereas dash arrows represent variational encoders.

5.2 Encapsulated Variational Auto-Encoders

5.2.1 Comparison between VAE and EVAE

Different to a conventional VAE (cf. Fig.5.1(i)), in our proposed EVAE (cf. Fig.5.1(ii)), the employment of two latent variables – under the coding theory – implies a structured novel encoder-decoder architecture. I.e., two separate probabilistic encoders, e.g., $q_{\phi_b}(z_b|x^{(i)})$ and $q_{\phi_s}(z_s|x^{(i)})$ and two decoders, e.g., $p_{\theta_b}(\tilde{x}|z_b)$ and $p_{\theta_s}(\tilde{x}|z_s)$ compiles up to the coinage of two VAEs, i.e., denoting as VAE_b and VAE_s . **It is crucial to note here that in our proposed EVAE, we encourage differentiated network architectures for two component VAEs. This encouragement permits the learning of two different sets of parameters, e.g., $\{\theta_b; \phi_b\}$ and $\{\theta_s; \phi_s\}$ for VAE_b and VAE_s , respectively.**

The representation encoding and data generative process can be described as following: two latent representations (values of z_b and z_s) are firstly encoded from the probabilistic encoders, then these latent representations are fed to the probabilistic decoders to generate two new data instances.

Importantly, as grouping two latent variables together, e.g., (z_b, z_s) , it allows us to denote the joint decoder and encoder in our proposed EVAE, i.e., $q_{\phi_b, \phi_s}((z_b, z_s)|x^{(i)}); p_{\theta_b, \theta_s}(\tilde{x}|(z_b, z_s))$. However, it is essential to note here that **we do not assume the conditional independency of two latent variables**. I.e., the simplified assumption of factorised joint encoder, e.g., $q_{\phi_b, \phi_s}((z_b, z_s)|x^{(i)}) = q_{\phi_b}(z_b|x^{(i)}) \cdot q_{\phi_s}(z_s|x^{(i)})$ or factorised joint decoders does not hold

in this research. Instead, we introduce a hyper-parameter α to induce the relation of two latent variables in deriving its corresponded specific analytic expressions of the joint encoder and decoder.

5.2.2 Parameterisations on Encoders & Decoders

Prior to derive the analytic form of our targeted joint encoder and decoder in EVAE, we chiefly introduce the parameterisations that are used in the component encoders, e.g., $q_{\phi_b}(z_b|x^{(i)})$ and $q_{\phi_s}(z_s|x^{(i)})$ and decoders, e.g., $p_{\theta_b}(\tilde{x}|z_b)$ and $p_{\theta_s}(\tilde{x}|z_s)$.

In this case, we parameterise the base encoder, e.g., $q_{\phi_b}(z_b|x^{(i)})$ under a simplified multivariate Gaussian distribution as: $\log q_{\phi_b}(z_b|x^{(i)}) = \log \mathcal{N}(z_b; \mu_b, \sigma^2)$, where the optimised variational parameters are ϕ_b , which can be used to produce the mean (μ_b) and s.d. (σ^2) of the approximating distribution.

Differ to the simplified parameterisation that is used in base encoder, we let a more complex full rank Gaussian distribution to parameterise the scaffolding encoder as follows. I.e., $\log q_{\phi_s}(z_s|x^i) = \log \mathcal{N}(z_s; \mu_s, L)$.

Where the parameterised variational parameter ϕ_s is the concatenation of the mean vector and the decomposed covariance matrix, i.e., $\{\mu_s, L\}$. Here, we use the numerical stable Cholesky decomposition to decompose the correlation matrix, e.g., Σ , into two lower triangular matrices, i.e., $\Sigma = LL^T$, to speed up the variational inference.

Armed with encoded latent representations (z_b and z_s), we let two decoders, i.e., $p_{\theta_b}(\tilde{x}|z_b)$ and $p_{\theta_s}(\tilde{x}|z_s)$, to take the similar forms of multivariate Gaussian that are used in forming the preceding encoders. I.e., $\log p_{\theta_b}(\tilde{x}|z_b) = \log \mathcal{N}(x; \mu_b, \sigma^2)$ and $\log p_{\theta_s}(\tilde{x}|z_s) = \log \mathcal{N}(x; \mu_s, L)$.

5.2.3 Hyper-parameter tuned latent representations

Armed with previous described parameterisations over two component encoders and decoders, it is positioned to derive the analytic expressions of our joint encoder and decoder. An optimal analytic expression of joint encoder or decoder should satisfy two following requirements. (1) The derived expression should be flexible enough to imply full factorised, full equality and the mixed relations between two component encoder or decoder. (2) The derived expression needs to be full differentiable to enable fast and accurate approximate marginal inference of the variable x .

To this end, we consider the analytic expression of our joint encoder that allows smooth interpolation of two component encoders in the following form in Eq.5.1:

$$q((z_b, z_s)|x^{(i)}; \phi_b, \phi_s, \alpha) \propto q_{\phi_b}(z_b|x^i) \cdot q_{\phi_s}(z_s|x^{(i)}) \alpha \cdot \exp\{\mathcal{R}_e(z_b^l, z_s^l)\}. \quad (5.1)$$

Note here, we insert a hyper-parameter α to exert direct control over the output of the discrepancy function, e.g., $\mathcal{R}_e(z_b^l, z_s^l)$. The added exp transformation of the discrepancy function is to ease the later derivation of the learning objective. The introduced discrepancy function, e.g., $\mathcal{R}_e(z_b, z_s)$ can be further expressed as following Eq. 5.2:

$$\mathcal{R}_e(z_b, z_s) = \frac{1}{L} \sum_{l=1}^L \exp\{-||z_b^l - z_s^l||^2\}. \quad (5.2)$$

As the samples are more easy to work with, we further Monte-Carlo sample these encoded representations from two encoders, i.e., $z_b^l \sim q(z_b|x)$ and $z_s^l \sim q(z_s|x)$, where l stands for the number of samples. In practise, a single sample suffices.

Similar to the prior noted factorisation in the encoder case, the analytic expression of the joint decoder, e.g., $p_{\theta_b, \theta_s}(\tilde{x}|(z_b, z_s))$ obtain a similar form in the following Eq 5.3,

$$p_{\theta_b, \theta_s}(x|(z_b, z_s); \alpha) \propto p_{\theta_b}(\tilde{x}_b|z_b) \cdot p_{\theta_s}(\tilde{x}_s|z_s) \alpha \cdot \exp\{\mathcal{R}_d(\tilde{x}_b, \tilde{x}_s)\}. \quad (5.3)$$

Notice here, we also incorporate the same hyper-parameter α to scale the differences of two reconstructed data instances, where the discrepancy function for the joint decoder, e.g., $\mathcal{R}_d(\tilde{x}_b, \tilde{x}_s)$ can be defined as 5.4:

$$\mathcal{R}_d(\tilde{x}_b, \tilde{x}_s) = \frac{1}{L} \sum_{l=1}^L \exp\{-||\tilde{x}_b - \tilde{x}_s||^2\}. \quad (5.4)$$

Here, \tilde{x}_b, \tilde{x}_s represent the reconstructed data instances from VAE_b and VAE_s , respectively, i.e., $\tilde{x}_b \sim p_{\theta_b}(\tilde{x}|z_b)$ and $\tilde{x}_s \sim p_{\theta_s}(\tilde{x}|z_s)$.

5.2.4 Learning objective

The model is complete by defining a simple factorised joint prior on two latent variables in the following form Eq. 5.5:

$$p_{\theta_b, \theta_s}(z_b, z_s) = p_{\theta_b}(z_b) \cdot p_{\theta_s}(z_s). \quad (5.5)$$

Armed with defined analytic expressions of joint encoder (cf. Eq.5.1) and joint decoder (cf. Eq.5.3), and the joint prior (cf. Eq.5.5), we can finally derive the objective function of our proposed EVAE.

Recall that the analytic formation of the learning objective of original VAE in [38] as Eq.5.6),

$$\mathcal{L}_{VAE}(\theta, \phi, x^{(i)}) = -\mathbb{D}_{KL}\{q_\phi(z|x)||p_\theta(z)\} + \frac{1}{L} \sum_{l=1}^L p_\theta(x^{(i)}|z). \quad (5.6)$$

Where the first term denotes the regularisation penalty from the variational encoder, and the second term expresses the decoder induced reconstruction loss between the generation and the original input. Following the similar derivation, the learning objective for the EVAE can then be derived in Eq.5.7),

$$\mathcal{L}_{EVAE} = \mathbb{E}_{q((z_b, z_s)|x^{(i)})} \left[-\mathbb{D}_{KL}\{q((z_b, z_s)|x^{(i)})||p(z_b, z_s)\} + \log p_{\theta_b, \theta_s}(\tilde{x}|(z_b, z_s)) \right]. \quad (5.7)$$

From the previous analytical derivation, the learning objective of EVAE can be seen as comprised of two components: the first one corresponds to the regularisation penalty of the derived joint encoder of our EVAE, whereas the residual term relates to the overall reconstruction errors from our derived joint decoder.

Substituting the defined joint encoder, joint decoder and the joint prior in Eq. 5.1 Eq. 5.3 and Eq. 5.5 into Eq. 5.7, we can further derive the learning objective into the following form:

$$\begin{aligned} & \mathcal{L}_{EVAE}(\theta_b, \theta_s, \phi_b, \phi_s, x^{(i)}) \\ &= \mathbb{E}_{q((z_b, z_s)|x^{(i)})} \left[-\{ \log(q_{\phi_b}(z_b|x^{(i)})) + \log(q_{\phi_s}(z_s|x^{(i)})) + \mathcal{R}_e(z_b^l, z_s^l) - \log p_{\theta_b}(z_b) - \log p_{\theta_s}(z_s) \} \right. \\ & \quad \left. + \log p_{\theta_b}(\tilde{x}|z_b) + \log p_{\theta_s}(\tilde{x}|z_s) + \mathcal{R}_d(\tilde{x}_b, \tilde{x}_s) \right] \\ &= \mathbb{E}_{q((z_b, z_s)|x^{(i)})} \left[-\{ \log(q_{\phi_b}(z_b|x^{(i)})) - \log(p_{\theta_b}(z_b)) \} + \log p_{\theta_b}(\tilde{x}|z_b) - \{ \log(q_{\phi_s}(z_s|x^{(i)})) \right. \\ & \quad \left. - \log(p_{\theta_s}(z_s)) \} + \log p_{\theta_s}(\tilde{x}|z_s) + [\alpha] \left\{ \mathcal{R}_e(z_b^l, z_s^l) - \mathcal{R}_d(\tilde{x}_b, \tilde{x}_s) \right\} \right] \\ &= \mathbb{E}_{q((z_b, z_s)|x^{(i)})} \left[-\mathbb{D}_{KL}\{q_{\phi_b}(z_b|x^{(i)})||p(z_b)\} + \log p_{\theta_b}(\tilde{x}|z_b) - \mathbb{D}_{KL}\{q_{\phi_s}(z_s|x^{(i)})||p(z_s)\} + \log p_{\theta_s}(\tilde{x}|z_s) \right. \\ & \quad \left. + [\alpha] \left\{ \mathcal{R}_d(\tilde{x}_b, \tilde{x}_s) - \mathcal{R}_e(z_b^l, z_s^l) \right\} \right] \\ &= \mathcal{L}_{VAE_b}(\theta_b, \phi_b, x^{(i)}) + \mathcal{L}_{VAE_s}(\theta_s, \phi_s, x^{(i)}) + [\alpha] \left\{ \mathcal{R}_d(\tilde{x}_b, \tilde{x}_s) - \mathcal{R}_e(z_b^l, z_s^l) \right\}. \quad (5.8) \end{aligned}$$

To rewrite the learning objective this way is to group two discrepancy functions together. Interestingly, the remaining terms, e.g., $\mathcal{L}_{vae_b}(\theta_b, \phi_b, x^{(i)})$ and $\mathcal{L}_{vae_s}(\theta_s, \phi_s, x^{(i)})$, match with the learning objectives of two conventional VAEs. Note here, to stress the importance of α in

the derived learning objective, we extract this α out from the original discrepancy functions of joint encoder in Eq. 5.1 and decoder in Eq. 5.3.

5.2.5 Learning algorithm

To learn our proposed EVAE, we resort on the reparameterisation trick in [38] to develop the optimisation algorithm. This approach directly employs the Monte Carlo sampling to attain the samples of latent variables. A more general learning algorithm that is developed for learning EVAE without the specific defined decoder (model likelihood), i.e., only probabilistic model can stick with its unspecified form, is documented in the Appendix B for interested readers.

As two sets of latent variables in EVAE are assumed to be continuous and differentiable in the real coordinate. And two sets of variational parameters are approximated in mean-field and full rank Gaussians, i.e., $z_b \sim z_b|x$ and $z_s \sim z_s|x$. Two valid re-parameterisations that apply to z_b and z_s can be specified as $z_b = \mu_b + \sigma \cdot \epsilon_b$ and $z_s = \mu_s + L \cdot \epsilon_s$, where ϵ_b and ϵ_s are two independent noise variables that both follow the normal distribution. The reconstruction errors in $\mathcal{L}_{VAE}(\theta_b, \phi_b, x^{(i)})$ and $\mathcal{L}_{VAE}(\theta_s, \phi_s, x^{(i)})$ can be further expressed as $\frac{1}{L} \sum_{l=1}^L p_\theta(\tilde{x}|\mu_b + \sigma \cdot \epsilon_b)$ and $\frac{1}{L} \sum_{l=1}^L p_\theta(\tilde{x}|\mu_s + L \cdot \epsilon_s)$.

In terms of KL divergence in $\mathcal{L}_{VAE}(\theta_b, \phi_b, x^{(i)})$ and $\mathcal{L}_{VAE}(\theta_s, \phi_s, x^{(i)})$, under Gaussian approximation, their KL terms can be integrated analytically as $\frac{1}{2} \sum_{k=1}^K (1 + \log(\sigma_k^{(i)})^2 - (\mu_{b_k}^{(i)})^2 - (\sigma_k^{(i)})^2)$ and $\frac{1}{2} \sum_{k=1}^K (1 + \log(L_k^{(i)})^2 - (\mu_{s_k}^{(i)})^2 - (L_k^{(i)})^2)$ [38]. The overall maximisation objective, \mathcal{J} , is therefore derived in (5.9),

$$\begin{aligned} \mathcal{J}_{(\theta_b, \theta_s, \phi_b, \phi_s)} = & \frac{1}{2} \sum_{k=1}^K (1 + \log(\sigma_k^{(i)})^2 - (\mu_{b_k}^{(i)})^2 - (\sigma_k^{(i)})^2) + \frac{1}{2} \sum_{k=1}^K (1 + \log(L_k^{(i)})^2 - (\mu_{s_k}^{(i)})^2 - (L_k^{(i)})^2) \\ & + \frac{1}{L} \sum_{l=1}^L p_\theta(\tilde{x}|\mu_b + \sigma \cdot \epsilon_b) + \frac{1}{L} \sum_{l=1}^L p_\theta(\tilde{x}|\mu_s + L \cdot \epsilon_s) + \alpha \{ \mathcal{R}_d(\tilde{x}|z_b, \tilde{x}|z_s) - \mathcal{R}_e(z_s|x^{(i)}, z_b|x^{(i)}) \}, \end{aligned} \quad (5.9)$$

where $\epsilon_b \sim \mathbb{N}(0, 1)$ and $\epsilon_s \sim \mathbb{N}(0, 1)$. As two sets of variational parameters stay in one learning objective, we rely on the alternating optimisation to derive the set of (θ_b, ϕ_b) and (θ_s, ϕ_s) iteratively. The overall learning algorithm is summarised below as Algorithm 2.

Algorithm 2 The learning algorithm of EVAE

Require: Dataset $X : x^{(i)}$; Encoder: $q(z_b, z_s | x^{(i)}; \phi_s, \phi_b)$; Decoder: $p(\tilde{x} | z_b, z_s; \theta_b, \theta_s)$
 Initialise the parameters of $\theta_b; \theta_s; \phi_b; \phi_s$
 Set the value of hyper-parameter $[\alpha]$
repeat
 while fix θ_s, ϕ_s **do**
 random sampling from ϵ_b , where $\epsilon_b \sim \mathbb{N}(0, 1)$
 Approximate $\nabla_{\theta_b, \phi_b}(\mathcal{J})$ via the differentiating the learning objective in E.q.5.9
 w.r.t θ_b, ϕ_b
 Update the θ_b, ϕ_b via the off shelf gradient ascent algorithm, e.g., Adam, using $\nabla_{\theta_b, \phi_b}(\mathcal{J})$
 end while
 while fix θ_b, ϕ_b **do**
 random sampling from ϵ_s , where $\epsilon_s \sim \mathbb{N}(0, 1)$
 Approximate $\nabla_{\theta_s, \phi_s}(\mathcal{J})$ via the differentiating the learning objective in E.q.5.9
 w.r.t θ_s, ϕ_s
 Update the θ_s, ϕ_s via the off shelf gradient ascent algorithm, e.g., Adam, using $\nabla_{\theta_s, \phi_s}(\mathcal{J})$
 end while
until the elbo(objective) is converged to certain level
Return the optimal $\theta_b; \theta_s; \phi_b; \phi_s$

5.3 Continuous Expression Generation in EVAE

Relying on a trained EVAE, we are now positioned to render out the generation process. To generate facial expressions, we need to firstly sample the input to encode low dimensional latent representations. Then we feed these latent representations to the joint decoder of a trained EVAE to generate expression images. To generate expressions along certain continuous axes, it demands the investigation of two continuous factors that are able to impact on the generation process.

The first factor is the input we use in encoding the latent representations. Assuming the prior distribution of latent variable as Normal distribution, we can feed any continuous values to the inverse CDF of a Normal distribution to encode the latent representations. We denote this factor as X_{sample} . The linear interpolation on this factor allows encoding of different latent representations on a continuous scale. **Here, we hypothesise that a continuous scale on this X_{sample} factor should work similar to either *arousal* or *valence* axis under the psychological conceptualised Arousal - Valence plane.**

The second continuous factor is the added hyper-parameter α . Differentiated α leads to diversified learning objectives in 5.8, which further correspond to different sets of learned op-

timal parameters in joint decoder, e.g., $\{\theta_b, \theta_s\}$. **Hence, we hypothesise that a continuous scale on this continuous factor α should allow the generation of continuous expressions along either *arousal* or *valence* axis under the psychological conceptualised Arousal - Valence plane in [23].**

5.4 Empirical Validation

We ran empirical experiments on two public available facial expression datasets, e.g., Frey faces and FERF-DB datasets. **The objective is to empirically validate whether the proposed EVAE with a tuneable hyper-parameter $[\alpha]$ is capable of generating continuous facial expressions along two hypothesised continuous scales on factors of X_{sample} and α that can be roughly corresponded to the *arousal* and *valence* axes in conceptualised Arousal-Valence plane in [23].**

Differ to easy assessment of discrete affective representations in classification tasks, to provide an objective metric to assess the performance of an unsupervised-trained is a challenging task. To this end, both quantitative and qualitative evaluations have to be taken into the consideration. The quantitative evaluation is served to measure how well our proposed EVAE is trained, whereas the qualitative one is reserved for assessing the performance (visual quality) of generated expressions. In terms of specific metrics, for the quantitative measurement, we default the commonly used optimised lower bound (elbo), i.e., \mathcal{L} [72], as our target index. For the comparative purpose, we include the original VAE as the benchmark in reporting the relevant statistics in the quantitative evaluation. For the qualitative validation, the perceptual quality via visual heuristics is our prior criteria in assessing the quality of generated expressions. All programs in these empirical experiments are written in python with libraries of tensorflow [73], pymc3 [74], and keras [75].

5.4.1 Experimental Set-ups

The Frey faces dataset contains a series of grey scaled 1956 images of Brendan Frey's face taken from sequential frames of a video with dimension of 20 x 28. Each image went through basic preprocessing step includes the input normalisation. This dataset is ideal to implement our derived EVAE in learning the data-driven A-V dimensional affect due to its assumed homogeneity in facial features, i.e., all emotional expression were rendered out from the single person. In terms of training and testing datasets, we employed the last 100 images for testing, whereas the remaining (1856) ones were used in the training session.

Facial Expression Research Group Database (FERG-DB) [76] is an annotated facial expression dataset, which utilised MAYA 3D modelling software to create the expressions in animated characters. Differ to previous dataset, e.g., Frey face dataset, this dataset is created specifically for automatic facial analysis. The entire dataset contains 55767 annotated facial expression images of six created characters. The modelled facial expressions range from low valence expressions, e.g., anger, disgust, sadness and fear, to high *valence* ones such as surprise, joy. To fit with our objective, we intentionally discard the original discrete annotations that associate with expressions. Each expression image in FERG-DB dataset were grey-scaled and normalised.

The EVAE from section 5.2 is used in this experiment. For two encoders, the two different feedforward neural networks with isotropic Gaussian priors, e.g., $p_{\theta_b}(x|z_b) = \mathcal{N}(z_b; 0, I)$ and $p_{\theta_s}(x|z_s) = \mathcal{N}(z_s; 0, I)$, were employed to warp the variational parameters of base and scaffolding encoders in EVAE in forming two inference networks. For the decoders, two complementary feedforward neural networks were used to serve as recognition networks.

The network configurations on the derived inference networks are summarised in Table 5.1. Note that, reckoning on other feasible configurations for the encoders and decoders, e.g., the number of hidden units and the type of layer wise activation, the relative performance of the model were observed as insensitive to these choices. For the cost function, we strictly follow the derived learning objective in Eq.5.8. To account for the reconstruction loss in the cost function, we were opted to favour the mean square error as the primary choice as it induced stability in the practical implementation.

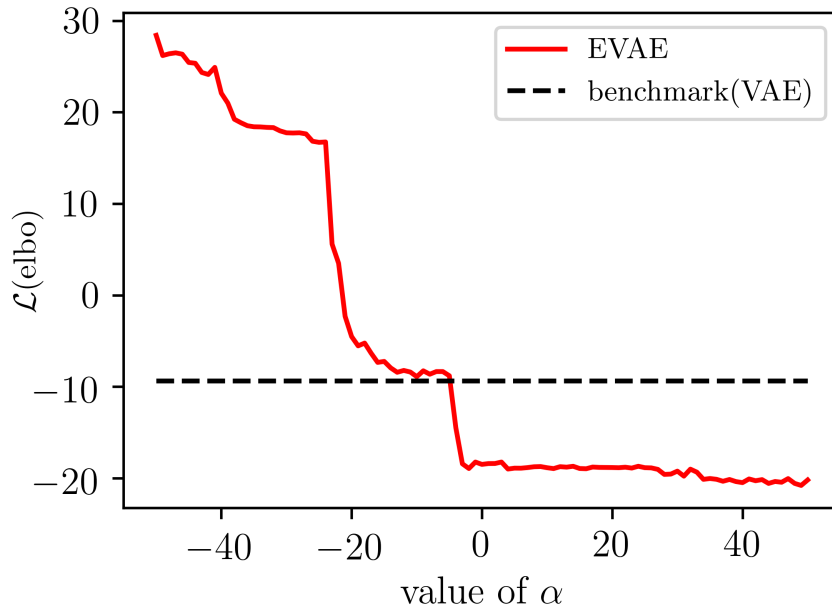
Compare to the previous implementation on Frey faces dataset, to learn A-V dimensional representations on FERG-DB facial expressions is a difficult task as some non-essential facial features for expression perception are also included, e.g., the female or muscular of the faces, the different hair styles. Hence, as for the employed inference and recognition networks, a different set of configurations was applied here, shown in the lower panel of Table 1. In terms of the loss function and the choice of gradient ascent algorithm, we continue to use the mean square loss and Adam, respectively. Different to previous fixed learning epochs, we adopt the early stopping technique here to prevent the overfitting via setting the patience to -2 ¹.

Both model and variational parameters are updated via the derived Algorithm 2. As the choice of gradient ascent algorithm has minor influence on the final performance, we defaulted the usage of adam[77] here. The results can be unfolded into following segments.

¹This corresponds to the termination of the training after two non-improving epochs of training. Therefore, the number of batches included in training was largely varied across training sessions here.

5.4.2 The optimised lower bound

The quantitative evaluation of a generative model commonly resort on either the estimated marginal likelihood of given training points [72] or the optimised lower bound(elbo) for the learned latent space. We adopt the later metric to report the relevant statistics. However, as depicted in previous section 5.2, different to the fixed elbo in conventional VAE, the learning of EVAE is bounded via a continuous scale on $[\alpha]$. This scale dependent lower bound on two datasets are visualised in Fig.5.2 and Fig 5.3, respectively. As a result, in terms of the optimised lower bound, the direct comparison of our generative model, i.e., EVAE, to the conventional VAE yields limited ratifications.

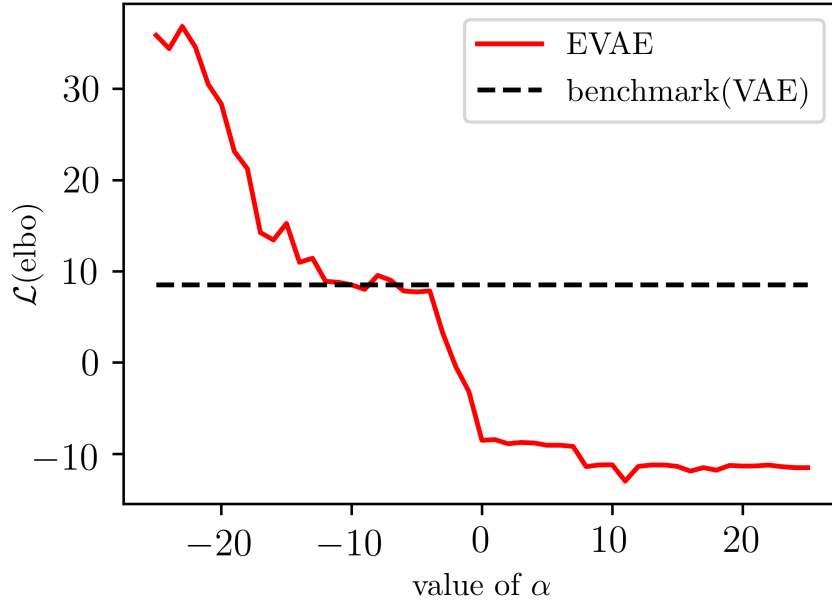


In this figure, it is clear that value of α , e.g., from negative (-50) to positive (50), has direct impact on the optimised learning elbo. As the value of α move towards the positive end, the optimised learning elbo is attenuated accordingly.

Fig. 5.2 The α bounded learning of EVAE on Frey faces dataset

5.4.3 Continuous generated expressions

Compare to the previous quantitative assessment, we are gravitated towards the qualitative evaluation to access the quality of generated expressions. In specific, we aim to evaluate the validity of our hypothesised continuous scales on X_{sample} and α in generating facial



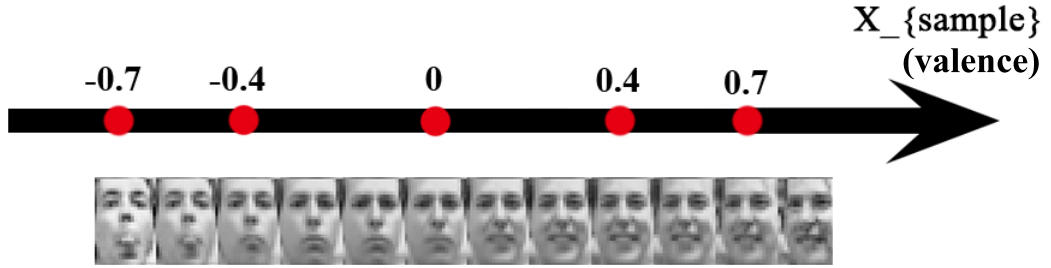
In this figure, it is clear that on FERG-DB dataset, the hyper-parameter α is able to exert the direct control over the optimised learning elbo of EVAE.

Fig. 5.3 The α bounded learning of EVAE on FERG-DB dataset

expressions with continuous transition patterns that behave similar to the *arousal* and *valance* axes in conceptualised Arousal-Valence plane.

The empirical results on both datasets in Fig. 5.4 and Fig. 5.5 show that via tuning the value on X_{sample} , we observe a clear generation pattern that ties closely with the *valence* axis in Arousal-Valence plane that was defined in [23]. I.e., from both generated facial expressions from Frey faces and animated FERG-DB datasets, there is a clear polarity shift from the generation of less negative facial expressions, e.g., sad and frustration looking expressions, to positive ones, e.g., happy and excitement looking expressions.

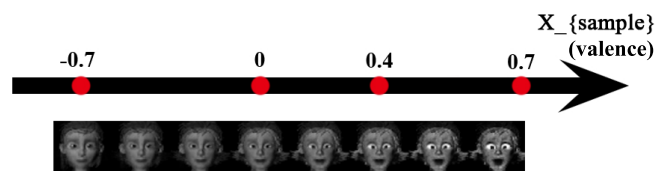
The empirical results on generated facial expressions also demonstrate that through tuning the continuous factor on α , we also observe a clear generation pattern with elevation on ranges of expressions. I.e., tuning the value of α from one end to the other, the generated expressions are diversified along this direction. This degree of diversification matches with the definition of *arousal* dimension in original Arousal-Valence plane. In detail, in Fig. 5.6 and Fig. 5.7, we choose 5 most representative values on this continuous scale respectively for two datasets, e.g., 50, 25, 0, -25, -50 for Frey faces dataset and 25, 10, 0, -10, -25 for FERG-DB dataset. With each α value, its associated generated expressions was visualised. We also adhere the plot of elbo loss for each α (above or below). It is clear that the variation



In this figure, based on the learned *valence* axis, the linear interpolation of it allows the generation of facial expressions that smoothly transformed from negative (low in *valence* dimension) to positive affect (high in *valence* dimension). Note here, the learned valence representations were rescaled to $[-1, 1]$ scale from the original $[0, 1]$ scale for later convenience in plotting.

Fig. 5.4 Facial expression generation along with the *valence* dimension on Frey Faces dataset

on this ratio linearly correlates with the varied affect in terms of their differentiated magnitude on the *arousal* axis. I.e., the negative ratio leads to more dramatic facial expressions, and wider range of regenerated affect (high arousal) in comparison to its positive alternative. This variation is also reflected on the differentiated loss plots. Notice here, the axes that rendered in Fig. 5.6 and Fig. 5.7 are ranged from positive to negative, this is done in purpose to coincide with the theoretical Arousal -Valence plane. Interestingly, as we fixed the α to 0, the reproduced affect exhibited the neutral face like expressions.



In this figure, through tuning the value on the modelled valence dimension, e.g., the sampled input. EVAE is capable of generating expressions from one polarity to the other.

Fig. 5.5 Facial expression generation along with the modelled *valence* dimension on FERG-DB dataset

Dataset	Model Architecture & Training Configurations
Frey-Face	<p>Input: 28 x 20 x 1</p> <p>Base Encoder: Conv 32x3x3 (stride 2)</p> <p>Pooling(stride 2)</p> <p>Conv 64x3x3 (stride 2) FC 200</p> <p>ReLU activation</p> <p>Scaffolding Encoder: Conv 32x3x3 (stride 2)</p> <p>Pooling(stride 2).</p> <p>Conv 32x3x3 (stride 2) FC 200. Gaussian</p> <p>ReLU activation</p> <p>Decoders: Deconv 32x3x3 (stride 2)</p> <p>Deconv 64x3x3 (stride 2)</p> <p>ReLU activation</p> <p>batch Size: 100</p> <p>num of batch: 150</p> <p>early stopping: No</p> <p>optimiser: Adam [77]</p>
FERG-DB	<p>Input: 48 x 48 x 1</p> <p>Base Encoder: Conv 32x3x3 (stride 2)</p> <p>Pooling(stride 2)</p> <p>Conv 64x2x2 (stride 2) FC 128</p> <p>ReLU activation</p> <p>Scaffolding Encoder: Conv 32x2x2 (stride 2)</p> <p>Pooling(stride 2).</p> <p>Conv 32x2x2 (stride 2) FC 140. Gaussian</p> <p>Tahn activation</p> <p>Decoders: Deconv 32x2x2 (stride 2)</p> <p>Deconv 64x2x2 (stride 2)</p> <p>Tahn activation</p> <p>Batch Size: 128</p> <p>num of batch: varied</p> <p>early stopping: Yes</p> <p>optimiser: Adam [77]</p>

Table 5.1 Model architectures and training details on Frey-Face and FERG-DB datasets.

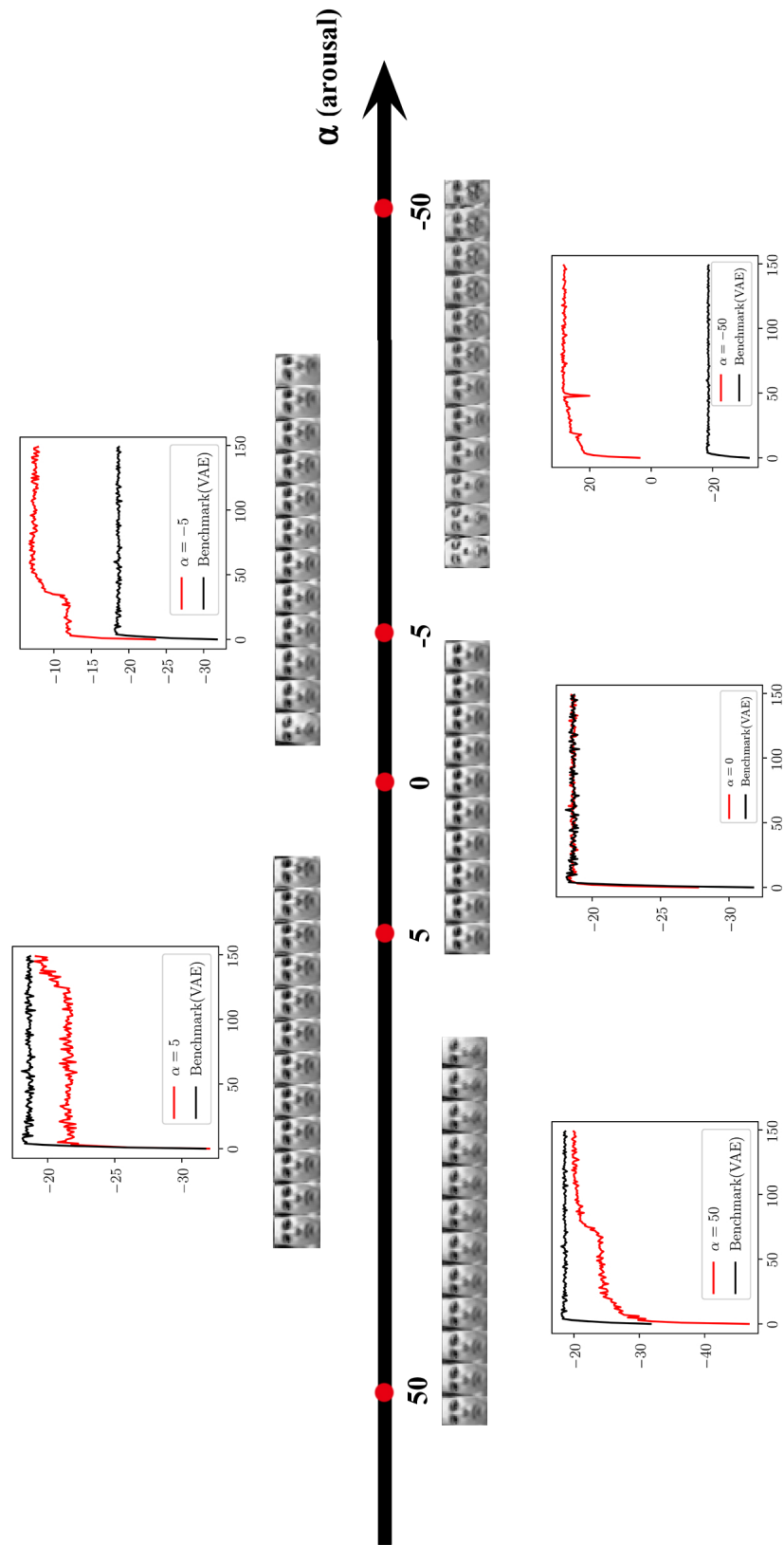


Fig. 5.6 Facial expression generation along with the modelled *arousal* dimension on Frey Faces dataset

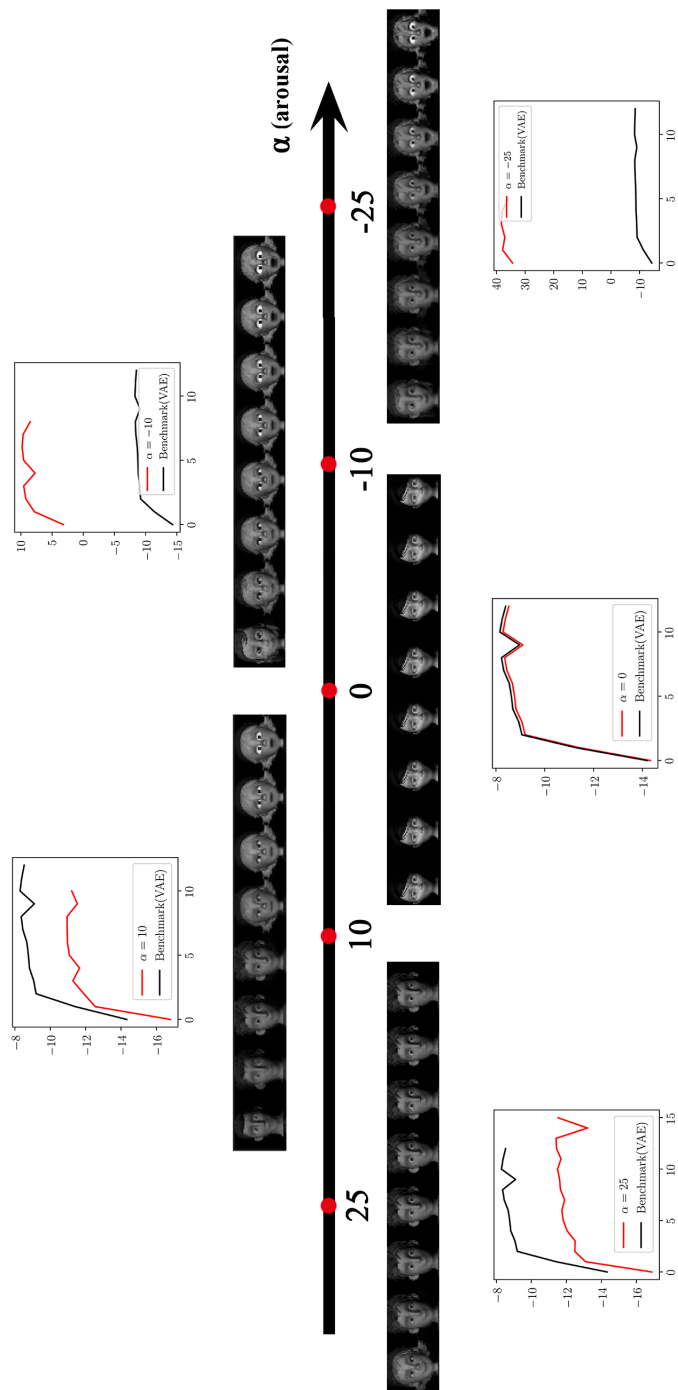


Fig. 5.7 Facial expression generation along with the modelled *arousal* dimension on FERF-DB dataset

5.5 Conclusion and Further Works

In this research, we propose a novel form of variational auto-encoder: encapsulated variational auto-encoders (EVAE) to generate expressions along two continuous axes. As two continuous factors, e.g., X_{sample} and α , exert the direct control on final expression generation process. We hypothesised two continuous axes on these factors can behave similar to the conceptualised *arousal* and *valence* axes in psychological defined Arousal and Valence plane. Through empirical experiments on two expression datasets, these hypothesised continuous axes do lead to patterns of expression generation that convey similar semantic meaning with the conceptualised *arousal* and *valence* axes.

Unfortunately, there are a few limitations in this research. The primary one is the poor perceptual quality of reproduced facial expressions. The secondary one is on the ad hoc judgement of the similarity between our hypothesised continuous scales and the theoretical defined arousal and valence axes. The tertiary one deals with the lack of an objective evaluation metric to access the performance of our generated expressions.

Chapter 6

Discussion and Conclusion

6.1 Summary of This Thesis

Representation learning – the cornerstone behind all downstream Artificial Intelligence applications – has gained enormous attention in both research and industry areas. Affective representation learning targets on learning representations in representing, generating, and comprehending of human affects. Unfortunately, with decades of researches, the consensus on the optimal affective representation has not been reached.

One school of researches that is inline with the influential basic emotion view [70] suggest the possibility of learning discrete representations in solving affect recognition tasks. These features are easy to attain, favourable in solving discriminative tasks in affective computing, and convenient to implement even under the large-scale industry demand. However, learning these representations merely permit outputting discrete annotations to categorise affective signal.

Contrast with the prior noted school in promoting the usage of simplified discrete representations, a different school of researches aims to learn complex continuous representation of human affect to reflect the richness of human affect. To support this school, we devote this thesis to render out three approaches in learning complex, dimensional affective representations of human facial expressions.

In Chapter 3, we lay out our first approach: the usage of label relaxation technique to improve the performance of a neural network on an expression classification task. Relied on the label relaxation technique, we transform the original one-hot encoded labels to real-numbered ones, which the degree of relaxation is tuned via a single hyper-parameter. Training a neural network with these relaxed labels can elevate its classification performance on a expression classification task.

As previous approach only permits single label on a facial expression, we then propose a zero-shot transfer learning framework: the uncertainty flow framework in Chapter 4 to allow a single label trained model to output multi-label predictions on certain expression. Benchmarking on a multi-label annotated expression dataset, we demonstrate the empirical superiority of our proposed uncertainty flow framework in outputting multi-label predictions.

Moving forward, in our final attempt, we rely on the generative model to generate continuous facial expressions in Chapter 5. Defaulting the explored generative model as the variational auto-encoder (VAE), we offer our modification over a conventional VAE: the encapsulated variational auto-encoders (EVAE). In our proposed EVAE, we are able to discover two continuous factors that exert direct impact on the overall generation process. Running our proposed EVAE on two facial expression dataset, we are able to generate continuous facial expressions in accordance with two prior-defined continuous dimensions.

These three approaches shed light on the feasibility of learning complex, continuous, and affective aware representations of human facial expressions. Despite of its nascent status, our attempts provide a glimpse of future in continuous, dimensional affective computing, which allows the intelligent agent to comprehend and unleash affects that once is unique to human being.

6.2 Shortages of Our Approaches

”All models are wrong, but some are useful [78].” The demonstrated success of our proposed three attempts in learning complex, dimensional affective representations does not belie their inherited weaknesses. In the following segments, we outline the weaknesses of our approaches from the micro and macro perspectives respectively.

6.2.1 The Micro View

Starting with our primary approach: the relied relaxation technique to transform the discrete label distribution to a partial continuous one allowing the production of pseudo continuous labels that is compatible with the supervised learning paradigm. In spite of its induced improvements on facial recognition tasks, their encoded representations are pseudo continuous and heavily relied on the discrete, single-labelled annotations. More importantly, under the current theoretical buildups, there is no guarantee that the relaxed label distribution is semantic valid. E.g., it might not be appropriate to relax annotation that is pertaining to the definitive facial expression annotation, such as neutral emotion to a bipolar shape continuous

distribution. Hence, a security check needs to design to ensure the relax label distributions to be semantic reliable.

In terms of our second approach, i.e., our proposed uncertainty flow framework in learning zero-shot transferred multi-label representations, a long-lasting concern in discussing the validity of multi-label learning paradigm pertains to its diversified evaluation metrics. Likewise in our second proposed approach, i.e., uncertainty flow framework of zero-shot transfer learning, we employed five different performance metrics in the empirical experiment to access the performance of our model. However, inline with the works from [54], it is hopeless hard for a multi-label model to outweigh other alternatives on every single metric. This posits an evaluation issue, i.e., on which circumstance can we claim the excellency of a certain multi-label recognition model.

Even under our proposed encapsulated variational auto-encoders (EVAE) in *Chapter 5*, the generated continuous facial representations are nonetheless suffered from two following issues: the lowered visual quality of these generated expressions, and the lack of theoretical analysis on effectiveness of our proposed EVAE. More importantly, the current semantic interpretation of yielded continuous factors are nonetheless in an ad hoc manner, a more theoretical linkage between these factors and dimensions in psychological conceptualised Arousal -Valence plane needs to be probed.

6.2.2 The Macro View

Winding back to *Chapter 1*, we stated our intended ignorance on temporal signal in learning the representation of facial expressions. This allows our thesis to pay the uniformed attention towards the spatial representations per se. Notwithstanding, in reality, an overt expression of a facial expression is usually accompanied with verbal information, such as vocal inflection or bodily postures. Hence, a more comprehensive account of affective representation of facial expressions should take the coherence of spatial and temporal signal into the consideration. More precisely, instead of the analyses on the static facial expressions, it is more advisable to extend our proposed approaches on the sequence of images or on the video directly. The extension of current theoretical model on the analysis on video demands multi-modal fusion of different types of signals other than the mere static image signal.

Additionally, as an essential component of cognitive aspect of emotion, the attention, especially visual attention is intertwined with the perception of a facial expression. Interestingly, numerous attention models that have been coined in solving the natural language problem. For this account, to incorporate the visual attention, i.e., the attention model into our current theoretical buildups may further improve the quality of learned affective representations.

6.3 Splendid Tomorrow

For decades of researches on affective computing, affective computing evolves from a peripheral realm to a central part of artificial intelligence researches. Numerous downstream applications have been developed [79] and applied to real world problems. Nowadays, researches on affective computing attract the inter-disciplined attention ranging from the domain of computer science, behaviour science to neuroscience.

More importantly, in line with our attempts in learning complex, continuous, and affective aware representations, there is a growing research interest in continuous affective computing driven by various advances and demands, e.g., real-time representation and analysis of naturalistic and continuous human affective behaviour. This leads to the pooling of inter-disciplined talents and research skills dedicating into this pioneer domain of affective computing. Despite of its infancy, international cooperations and fiscal efforts that aim to nurture this state-of-the-art subdomain are commonly noted, e.g., the project of European Union FP-7, the SEMAINE project, and et ac.

The featured continuous dimensional representations are able to improve current affective recognition applications in a large extent. I.e., being able to represent affects in human-like dimensional plane, it allows the more genuine affect generation that can smooth the interaction between human and computers. These dimensional affect representations can also benefit the longitudinal application that demands the storage of 'affect' in an efficient and consistent manner.

For the concluding remark of this thesis, we firmly believe the splendid future on learning complex and continuous affective representations of facial expressions. The challenge of learning such representation is formidable, and with tremendous risk, but it stands to move affective computing in a radically different direction: toward embracing part of the spark that makes us human.

Bibliography

- [1] Rosalind Wright Picard et al. “Affective computing”. In: (1995).
- [2] CL Lisetti. *Affective computing*. 1998.
- [3] Darwin Charles, Ekman Paul, and Prodger Phillip. “The expression of the emotions in man and animals”. In: *Electronic Text Center, University of Virginia Library* (1872).
- [4] G.-B.(Guillaume-Benjamin) Duchenne. *The expression of the emotions in man and animals*. John Murray, 1872.
- [5] Paul Ekman. “Universals and cultural differences in facial expressions of emotion.” In: *Nebraska symposium on motivation*. University of Nebraska Press. 1971.
- [6] Paul Ekman and Erika L Rosenberg. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, USA, 1997.
- [7] Kidport Reference Library. *Head and Face Muscles*. [Online; accessed Oct. 28th 2018]. 2012. URL: <http://www.kidport.com/RefLib/Science/HumanBody/MuscularSystem/HeadFaceMuscles.htm>.
- [8] Burrhus Frederic Skinner. *About behaviorism*. Vintage, 2011.
- [9] Takeo Kanade, Yingli Tian, and Jeffrey F Cohn. “Comprehensive database for facial expression analysis”. In: *fg*. IEEE. 2000, p. 46.
- [10] Patrick Lucey et al. “The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression”. In: *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*. IEEE. 2010, pp. 94–101.
- [11] Evangelos Sariyanidi, Hatice Gunes, and Andrea Cavallaro. “Automatic analysis of facial affect: A survey of registration, representation, and recognition”. In: *IEEE transactions on pattern analysis and machine intelligence* 37.6 (2015), pp. 1113–1133.
- [12] Soujanya Poria et al. “A review of affective computing: From unimodal analysis to multimodal fusion”. In: *Information Fusion* 37 (2017), pp. 98–125.
- [13] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. “Face description with local binary patterns: Application to face recognition”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 12 (2006), pp. 2037–2041.
- [14] Ville Ojansivu and Janne Heikkilä. “Blur insensitive texture classification using local phase quantization”. In: *International conference on image and signal processing*. Springer. 2008, pp. 236–243.

- [15] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE. 2005, pp. 886–893.
- [16] Martin Lades et al. "Distortion invariant object recognition in the dynamic link architecture". In: *IEEE Transactions on computers* 3 (1993), pp. 300–311.
- [17] Karan Sikka et al. "Exploring bag of words architectures in the facial expression domain". In: *European Conference on Computer Vision*. Springer. 2012, pp. 250–259.
- [18] Ruicong Zhi et al. "Graph-preserving sparse nonnegative matrix factorization with application to facial expression recognition". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 41.1 (2011), pp. 38–52.
- [19] Shane F Cotter. "Sparse representation for accurate classification of corrupted and occluded facial expressions". In: *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE. 2010, pp. 838–841.
- [20] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), p. 436.
- [21] 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.
- [22] Paul Ekman. "An argument for basic emotions". In: *Cognition & emotion* 6.3-4 (1992), pp. 169–200.
- [23] James A Russell. "A circumplex model of affect." In: *Journal of personality and social psychology* 39.6 (1980), p. 1161.
- [24] Ian J Goodfellow et al. "Challenges in representation learning: A report on three machine learning contests". In: *International Conference on Neural Information Processing*. Springer. 2013, pp. 117–124.
- [25] Emad Barsoum et al. "Training deep networks for facial expression recognition with crowd-sourced label distribution". In: *Proceedings of the 18th ACM International Conference on Multimodal Interaction*. ACM. 2016, pp. 279–283.
- [26] Michael Irwin Jordan, Terrence Joseph Sejnowski, and Tomaso A Poggio. *Graphical models: Foundations of neural computation*. MIT press, 2001.
- [27] Kevin P Murphy. "Machine learning: a probabilistic perspective". In: (2012).
- [28] Geoffrey E Hinton. "Training products of experts by minimizing contrastive divergence". In: *Neural computation* 14.8 (2002), pp. 1771–1800.
- [29] Yann LeCun et al. "Comparison of learning algorithms for handwritten digit recognition". In: *International conference on artificial neural networks*. Vol. 60. Perth, Australia. 1995, pp. 53–60.
- [30] Anil K Jain, Jianchang Mao, and K Moidin Mohiuddin. "Artificial neural networks: A tutorial". In: *Computer* 29.3 (1996), pp. 31–44.
- [31] Radford M Neal. "Bayesian learning via stochastic dynamics". In: *Advances in neural information processing systems*. 1993, pp. 475–482.
- [32] David JC MacKay. "Bayesian interpolation". In: *Neural computation* 4.3 (1992), pp. 415–447.

- [33] Christopher M Bishop. “Bayesian neural networks”. In: *Journal of the Brazilian Computer Society* 4.1 (1997).
- [34] Igor Kononenko. “Bayesian neural networks”. In: *Biological Cybernetics* 61.5 (1989), pp. 361–370.
- [35] Radford M Neal. *Bayesian learning for neural networks*. Vol. 118. Springer Science & Business Media, 2012.
- [36] Michael Irwin Jordan. *Learning in graphical models*. Vol. 89. Springer Science & Business Media, 1998.
- [37] Ian Goodfellow. “NIPS 2016 tutorial: Generative adversarial networks”. In: *arXiv preprint arXiv:1701.00160* (2016).
- [38] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).
- [39] Seán Mc Loone and George Irwin. “Improving neural network training solutions using regularisation”. In: *Neurocomputing* 37.1-4 (2001), pp. 71–90.
- [40] Eric Jang, Shixiang Gu, and Ben Poole. “Categorical reparameterization with gumbel-softmax”. In: *arXiv preprint arXiv:1611.01144* (2016).
- [41] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. “The concrete distribution: A continuous relaxation of discrete random variables”. In: *arXiv preprint arXiv:1611.00712* (2016).
- [42] Lingxi Xie et al. “Disturblabel: Regularizing cnn on the loss layer”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 4753–4762.
- [43] Emil J Gumbel. “Bivariate exponential distributions”. In: *Journal of the American Statistical Association* 55.292 (1960), pp. 698–707.
- [44] Emil Julius Gumbel. *Statistics of extremes*. Courier Corporation, 2012.
- [45] K. Simonyan and A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *CoRR* abs/1409.1556 (2014).
- [46] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. “Mining multi-label data”. In: *Data mining and knowledge discovery handbook*. Springer, 2009, pp. 667–685. URL: https://link.springer.com/chapter/10.1007/978-0-387-09823-4_34.
- [47] Oscar Luaces et al. “Binary relevance efficacy for multilabel classification”. In: *Progress in Artificial Intelligence* 1.4 (2012), pp. 303–313.
- [48] Fernando Benites and Elena Sapozhnikova. “HARAM: a Hierarchical ARAM neural network for large-scale text classification”. In: *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on*. IEEE. 2015, pp. 847–854.
- [49] Min-Ling Zhang and Zhi-Hua Zhou. “ML-KNN: A lazy learning approach to multi-label learning”. In: *Pattern recognition* 40.7 (2007), pp. 2038–2048.
- [50] Min-Ling Zhang and Zhi-Hua Zhou. “Multilabel neural networks with applications to functional genomics and text categorization”. In: *IEEE transactions on Knowledge and Data Engineering* 18.10 (2006), pp. 1338–1351.

- [51] Emily Mower, Maja J Mataric, and Shrikanth Narayanan. “A Framework for Automatic Human Emotion Classification Using Emotion Profiles”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.5 (July 2011), pp. 1057–1070. ISSN: 1558-7916, 1558-7924. DOI: 10.1109/TASL.2010.2076804. URL: <http://ieeexplore.ieee.org/document/5585726/>.
- [52] Emily Mower et al. “Interpreting ambiguous emotional expressions”. In: *Affective Computing and Intelligent Interaction and Workshops, 2009. ACII 2009. 3rd International Conference on*. IEEE, 2009, pp. 1–8. URL: <http://ieeexplore.ieee.org/abstract/document/5349500/>.
- [53] Kaili Zhao et al. “Multi-label learning with prior knowledge for facial expression analysis”. In: *Neurocomputing* 157 (), pp. 280–289. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2015.01.005. URL: <http://www.sciencedirect.com/science/article/pii/S0925231215000260>.
- [54] Gjorgji Madjarov et al. “An extensive experimental comparison of methods for multi-label learning”. In: *Pattern Recognition* 45.9 (Sept. 2012), pp. 3084–3104. ISSN: 00313203. DOI: 10.1016/j.patcog.2012.03.004. URL: <http://linkinghub.elsevier.com/retrieve/pii/S0031320312001203>.
- [55] Sinno Jialin Pan and Qiang Yang. “A survey on transfer learning”. In: *IEEE Transactions on knowledge and data engineering* 22.10 (2010), pp. 1345–1359.
- [56] W Bai and C Quan. “Harness the Model Uncertainty via Hierarchical Weakly Informative Priors in Bayesian Neural Network”. In: *Int Rob Auto J* 3.3 (2017 in press).
- [57] Andrew Gelman et al. “A weakly informative default prior distribution for logistic and other regression models”. In: *The Annals of Applied Statistics* 2.4 (Dec. 2008), pp. 1360–1383. ISSN: 1932-6157. DOI: 10.1214/08-AOAS191. arXiv: 0901.4011. URL: <http://arxiv.org/abs/0901.4011>.
- [58] Michael I. Jordan et al. “An introduction to variational methods for graphical models”. In: *Machine learning* 37.2 (1999), pp. 183–233. URL: <http://link.springer.com/article/10.1023/A:1007665907178>.
- [59] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. “Variational Inference: A Review for Statisticians”. In: *Journal of the American Statistical Association* 112.518 (Apr. 3, 2017), pp. 859–877. ISSN: 0162-1459, 1537-274X. DOI: 10.1080/01621459.2017.1285773. arXiv: 1601.00670. URL: <http://arxiv.org/abs/1601.00670>.
- [60] Alp Kucukelbir et al. “Automatic differentiation variational inference”. In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 430–474.
- [61] Wei-Tsih Lee and Manoel Fernando Tenorio. *On Optimal Adaptive Classifier Design Criterion: How Many Hidden Units are Necessary for an Optimal Neural Network Classifier?* Purdue University, School of Electrical Engineering, 1991.
- [62] John S. Denker and Yann Lecun. “Transforming neural-net output levels to probability distributions”. In: *Advances in neural information processing systems*. 1991, pp. 853–859. URL: <http://papers.nips.cc/paper/419-transforming-neural-net-output-levels-to-probability-distributions.pdf>.
- [63] Pierre-Luc Carrier et al. “FER-2013 face database”. In: *Technical report* (2013).

- [64] Joseph L Fleiss and Jacob Cohen. “The equivalence of weighted kappa and the intra-class correlation coefficient as measures of reliability”. In: *Educational and psychological measurement* 33.3 (1973), pp. 613–619.
- [65] Jesse Read et al. “Classifier chains for multi-label classification”. In: *Machine Learning and Knowledge Discovery in Databases* (2009), pp. 254–269.
- [66] Grigorios Tsoumakas and Ioannis Vlahavas. “Random k-labelsets: An ensemble method for multilabel classification”. In: *Machine learning: ECML 2007* (2007), pp. 406–417.
- [67] Min-Ling Zhang. “ML-RBF: RBF neural networks for multi-label learning”. In: *Neural Processing Letters* 29.2 (2009), pp. 61–74.
- [68] Yunchao Wei et al. “HCP: A flexible CNN framework for multi-label image classification”. In: *IEEE transactions on pattern analysis and machine intelligence* 38.9 (2016), pp. 1901–1907.
- [69] Martin Wöllmer et al. “Abandoning emotion classes-towards continuous emotion recognition with modelling of long-range dependencies”. In: *Proc. 9th Interspeech 2008 incorp. 12th Australasian Int. Conf. on Speech Science and Technology SST 2008, Brisbane, Australia*. 2008, pp. 597–600.
- [70] Paul Ekman. “Are there basic emotions?” In: (1992).
- [71] Keith A. Bush et al. “Distributed Neural Processing Predictors of Multi-dimensional Properties of Affect”. In: *Frontiers in Human Neuroscience* 11 (Sept. 14, 2017). ISSN: 1662-5161. DOI: 10.3389/fnhum.2017.00459. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5603694/>.
- [72] Tom White. “Sampling Generative Networks: Notes on a Few Effective Techniques”. In: *arXiv:1609.04468 [cs, stat]* (Sept. 14, 2016). arXiv: 1609.04468. URL: <http://arxiv.org/abs/1609.04468>.
- [73] Martín Abadi et al. “Tensorflow: Large-scale machine learning on heterogeneous distributed systems”. In: *arXiv preprint arXiv:1603.04467* (2016).
- [74] John Salvatier, Thomas V Wiecki, and Christopher Fonnesbeck. “Probabilistic programming in Python using PyMC3”. In: *PeerJ Computer Science* 2 (2016), e55.
- [75] François Chollet et al. “Keras: Deep learning library for theano and tensorflow”. In: URL: <https://keras.io/k> 7.8 (2015).
- [76] Deepali Aneja et al. “Modeling Stylized Character Expressions via Deep Learning”. In: *Asian Conference on Computer Vision*. Springer. 2016, pp. 136–153.
- [77] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).
- [78] George EP Box, William Gordon Hunter, J Stuart Hunter, et al. “Statistics for experimenters”. In: (1978).
- [79] Rosalind W Picard. “Affective computing: challenges”. In: *International Journal of Human-Computer Studies* 59.1-2 (2003), pp. 55–64.
- [80] David J Olive. *Statistical theory and inference*. Springer, 2014.
- [81] Erhan Çinlar. *Probability and stochastics*. Vol. 261. Springer Science & Business Media, 2011.

- [82] Tijmen Tieleman and Geoffrey Hinton. “Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude”. In: *COURSERA: Neural networks for machine learning* 4.2 (2012), pp. 26–31.

Appendix A

Notations and Rules in Probability and Inferential Statistics

A.1 Overall Workflow

Following Box's philosophy of statistical and machine learning, the work-flow of statistical learning can be viewed in Fig A.1 [78]. The iterative process of applying data to make inferences based on the model, then to criticise the model to make better inferences.

A.2 Model Statistics

- Mean & Expectation

For one random variable z and its sample distribution is Q , its mean is equivalent to $\mathbb{E}_{z \sim Q} = \mu(z)$. It is commonly referred as first-order statistics.

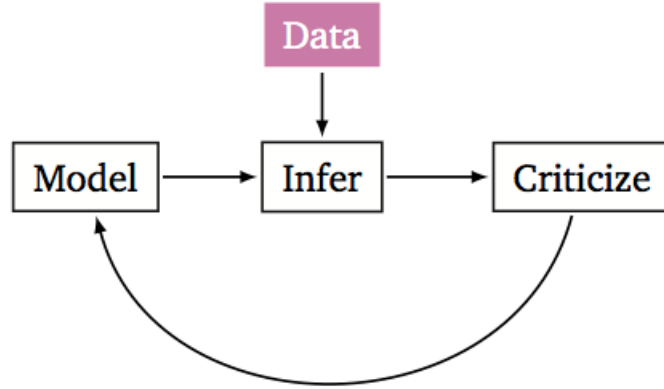
- Variance

The measure of the spread of ones' distribution, i.e., $var[X] = \mathbb{E}[(X - \mu)^2]$. It is commonly termed as second-order statistics.

- Covariance

In multi-variant distribution, it is used to measure the relation among variables, i.e., the magnitude of one variable change on the other. E.g., $cov[X, Y] = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$.

- Prior Distribution



In Fig.A.1, it renders Box's workflow of learning a probabilistic model [78].

Fig. A.1 The standard workflow of learning a probabilistic model

The pre-defined belief about the distribution of certain variables before seeing the observation/applying the datapoint to fit the model, e.g., $p(z)$ for $\{z\}$.

- **Conditional Likelihood**

A probability distribution that describes the how one variable depends on the other one. E.g., in latent variable model, $p(x|z)$ means how any data x depend on the latent variables z (another interpretation is the probabilistic decoder).

- **Joint Distribution**

A probability distribution of modelling two variables together. E.g., $p(x, z)$ for modeling the visible variable x and hidden variable z .

- **Posterior Distribution:**

Applying Bayesian' Rule to fitting the pre-defined prior distribution to the conditional likelihood. E.g., $p(y|x) \propto p(x|y)p(y)$.

A.3 Useful Rules

Chain Rule

$$p(X_{1:D}) = p(X_1)p(X_2|X_1)p(X_3|X_2, X_1)...p(X_D|X_{(1:D-1)}) \quad (\text{A.1})$$

Bayesian' Rule

$$p(X = x|Y = y) = \frac{p(X = x, Y = y)}{p(Y = y)} = \frac{p(X = x)p(Y = y|X = x)}{\int_x p(X = x')p(Y = y|X = x')} \quad (\text{A.2})$$

Linear Transformations of The Distribution of Random Variables

1. Suppose $f()$ is a linear function, i.e., $y = f(x) = Ax + b$ and $x \sim p()$, then $\mathbb{E}[y] = \mathbb{A}\mathbb{u} + \mathbb{b}$; $\text{cov}[y] = \text{cov}[Ax + b] = A\Sigma A^T$, where $\Sigma = \text{cov}[x]$. 2. Suppose $x \sim p(x)$ and $s \sim p(s)$, $x = As$, where A is the mixing matrix, and $A \in \mathfrak{R}^n$, we let $W = A^{-1}$ to be the un-mixing matrix. $p(x) = \prod_{i=1}^n p_s(w_i^T x) \cdot |w|$.

A.4 Model Learning/Parameter Estimation

A.4.1 MLE

The first weapon we have in our parameter estimation arsenal is the maximum likelihood principle. Intuitively, it can be understood as the action of choosing the best fitted parameters, e.g., $\{\theta\}$, to empower the expressiveness and forecastability of the model. Formally, this principle is defined as:

$$\hat{\theta}_{ML} = \arg \max_{\theta} P(D|\theta) \propto \arg \max_{\theta} \log P(D|\theta) \quad (\text{A.3})$$

Where D represents the variable sets, i.e., in supervised setting, it can be $\{x, y\}$, and in generative model ($\{x\}$). Most of time, we work on the $\log P()$, i.e., log probability, for its easy computation over 0-1 domain for numerical stability.

A more advanced interpretation of the MLE is to view MLE as a process of finding the distribution that is 'closest' to the empirical distribution \hat{P}_D from a family of estimated distributions. Thereby, the aforementioned formula can be translated as following:

$$\hat{\theta}_{ML} = M(H_{\hat{P}_D}(X) - KL(\hat{P}_D(X)||P(X|\theta))) \quad (\text{A.4})$$

Here, M represents as M-projection, and the KL is the Kuller-Levenge divergence for measuring the dissimilarity between two distributions.

A.4.2 Bayesian parameter estimation

The major drawback of MLE is its ignorance of prior information, albeit its effectiveness in real life application. To recall that the prior is a pre-defined/assumed brief over our designed

distribution, the inclusion of prior(s) is justified in yielding more accurate density estimation. Then the foregoing MLE is transformed into maximise a posterior, AKA., MAP estimation.

$$\begin{aligned}
 \hat{\theta}_{MAP} &= \arg \max_{\theta} P(\theta|D) \\
 &= \arg \max_{\theta} P(D|\theta)P(\theta) \\
 &= \arg \max_{\theta} \log P(D|\theta) + \arg \max_{\theta} \log P(\theta) \quad (\text{A.5})
 \end{aligned}$$

Hence, via Bayesian's rule, we can compute the MAP to estimate the model parameters, which depended upon the correctness of MLE estimation and in what degree our obtained prior is valid.

A.5 Model Critics

Model criticism typically analyses the posterior predictive distribution,

$$p(x_{new}|x_{obs}) = \int p(x_{new}|z)p(z|x_{obs})dz \quad (\text{A.6})$$

Then we apply the posterior predictive checks (PPC) to see the deviation between true distribution and generated distribution.

Appendix B

The General Learning Algorithm for Encapsulated Variational Auto-Encoders

In *Chapter 5*, we render out a specific tailored learning algorithm for training the encapsulated variational auto-encoder (EVAE) as Algorithm 2.

Differ to the preceding rendered algorithm, here, we present a general learning algorithm that also achieve the training process but with no assumption on the parameterisation of the decoder (model likelihood). For this reason, the yielded algorithm is much general to the previous one, and applicable to models beyond variational auto-encoder, such as variational Gaussian mixture model.

This general learning algorithm is based on the recent proposed inference framework: automatic differential variational inference [60](ADVI). Under this framework, there is no need to specify the model likelihood into the decoder form. For this reason, only the joint encoder from EVAE is considered here, i.e., $q_{\phi_b, \phi_s}((z_s, z_b)|x, \alpha)$, here we use the transformed hyper-parameter *alpha* instead of the raw one λ .

Given the encoder, i.e., $q_{\phi_b, \phi_s}(z_s, z_b|x, \alpha)$, and a unspecified probabilistic model, i.e., $p(x, z_b, z_s)$, the overall learning objective, i.e., elbo(evidence lower bound) is defined as Eq.B.1:

$$\mathcal{L} = \mathbb{E}_{q(z_b, z_s|x)} [\log p(x, z_b, z_s) - \log q(z_s, z_b|x)]. \quad (\text{B.1})$$

B.0.1 Latent Variable Transformation and Elliptical Standardisation

The primary step in deriving a EVAE compatible ADVI learning algorithm is to transform the latent variables, e.g., z_b, z_s from the constrained space to the unconstrained space, i.e., $\mathfrak{R}_+ \rightarrow \mathfrak{R}$, we apply two simple differentiable functions, e.g., $T_1 : \zeta \leftarrow \log(z_b)$ and $T_2 : \beta \leftarrow$

$\log(z_s)$, to automate this transformation. The learning objective is transformed¹ accordingly to Eq.B.2:

$$\mathbb{E}_{q(\zeta, \beta | x; \phi_b, \phi_s)} \left[p(x, T_1^{-1}(\zeta), T_2^{-1}(\beta)) |\det \mathcal{J}_{T_1^{-1}}(\zeta)| |\det \mathcal{J}_{T_2^{-1}}(\beta)| - \log q(\zeta | x) - \log q(\beta | x) + \alpha \cdot \mathcal{R}_e \right]. \quad (\text{B.2})$$

Posterior to the latent variable transformation, the second step is to standardise the unknown Gaussian density of previous transformed parameters, e.g., ζ, β to standard Normal distribution, where $\mu = 0; \sigma^2 = 1$. For this account, we apply the elliptical standardisation inline with original usage of ADVI in [60]. Elliptical standardisation entails that the usage of transformation $S_{\phi_b, S_{\phi_s}}$ to absorb the variational parameters, e.g., ϕ_b and ϕ_s to convert the unknown Gaussian approximation to a standard Gaussian, e.g., η_b and η_s . I.e., $\eta_b = S_{\phi_b}(\zeta) = \text{diag}(\exp(\omega))^{-1}(\zeta - \mu_b)$ and $\eta_s = S_{\phi_s}(\beta) = L^{-1}(\beta - \mu_s)$. Note here, this standardisation is equivalent to the reparameterisation trick we employed before.

B.0.2 Gradients w.r.t. Learning Objective

After the transformation of latent variables from the constrained to unconstrained real space and the elliptical standardisation on parameters, the learning elbo is then derived as Eq.B.3:

$$\begin{aligned} \mathcal{L} = & \mathbb{E}_{\mathcal{N}_{\eta_b(0, I)}; \mathcal{N}_{\eta_s(0, I)}} \left\{ \log p(x, T_1^{-1}[S_1^{-1}(\eta_b)], T_2^{-1}[S_2^{-1}(\eta_s)] |\det \mathcal{J}_{T_1^{-1}}[S_1^{-1}(\eta_s)]| |\det \mathcal{J}_{T_2^{-1}}[S_2^{-1}(\eta_s)]| \right. \\ & \left. - \log q([S_2^{-1}(\eta_b | x)]) - \log q([S_2^{-1}(\eta_s | x)]) + \alpha \cdot ||q([S_2^{-1}(\eta_b | x)]) - q([S_2^{-1}(\eta_s | x)])||^2 \right\}, \end{aligned} \quad (\text{B.3})$$

where to-be-optimised parameters are $\eta_b : \{\mu_b, \omega\}$ and $\eta_s : \{\mu_s, L\}$.

As two sets of optimisation parameters exist in one differential equation, we resort on an alternating process to derive the gradients of each, which is different to the original implementation of ADVI. Respecting to the first parameter μ_b , its gradient ∇, μ_b can be derived as Eq.B.4:

¹According to the theory of variable change, the density of transformed latent variable is the density of original variable adjusted by the determinant of the Jacobian of the inverse transformation [80].

$$\begin{aligned} \nabla \mu_b(\mathcal{L}) = & \mathbb{E}_{\mathcal{N}(\eta_b; 0, I)} \left\{ \nabla z_b \log p[x, z_b, T_2^{-1}(S_2^{-1}(\eta_s))] \cdot \nabla \zeta T_1^{-1}(\zeta) + \nabla \zeta \log |\det \mathcal{J}_{T^{-1}}(\zeta)| - \nabla \zeta \log q(\zeta|x) \right. \\ & \left. + \nabla \zeta 2 \cdot \alpha[q([S_1^{-1}(\eta_b|x)]) - q([S_2^{-1}(\eta_s|x)])] \right\}. \quad (\text{B.4}) \end{aligned}$$

The detailed derivation can be delineated in the following Eq.B.5:

$$\begin{aligned} \nabla \mu_b(\mathcal{L}) &= \nabla \mu_b \mathbb{E}_{\mathcal{N}(\eta_b; 0, I)} \left\{ \log p[x, T_1^{-1}(S_1^{-1}(\eta_b)), T_2^{-1}(S_2^{-1}(\eta_s))] + \log |\det \mathcal{J}_{T_1^{-1}}(S_1^{-1}(\eta_b))| \right. \\ &\quad \left. + \log |\det \mathcal{J}_{T_2^{-1}}(S_2^{-1}(\eta_s))| - \log q(S_1^{-1}(\eta_b)) - \log q(S_2^{-1}(\eta_s)) \right. \\ &\quad \left. + 2 \cdot \alpha[q([S_1^{-1}(\eta_b|x)]) - q([S_2^{-1}(\eta_s|x)])]^2 \right\} \\ &= \mathbb{E}_{\mathcal{N}(\eta_b; 0, I)} \left\{ \nabla \mu_b [\log p[x, T_1^{-1}(S_1^{-1}(\eta_b)), T_2^{-1}(S_2^{-1}(\eta_s))] + \log |\det \mathcal{J}_{T_1^{-1}}(S_1^{-1}(\eta_b))| \right. \\ &\quad \left. + \log |\det \mathcal{J}_{T_2^{-1}}(S_2^{-1}(\eta_s))| - \log q(S_1^{-1}(\eta_b)) - \log q(S_2^{-1}(\eta_s)) \right. \\ &\quad \left. + 2 \cdot \alpha[q([S_1^{-1}(\eta_b|x)]) - q([S_2^{-1}(\eta_s|x)])]^2 \right\} \\ &= \mathbb{E}_{\mathcal{N}(\eta_b; 0, I)} \left\{ \nabla z_b \log p[x, T_1^{-1}(S_1^{-1}(\eta_b)), T_2^{-1}(S_2^{-1}(\eta_s))] \cdot \nabla \zeta T_1^{-1}(S_1^{-1}(\eta_b)) \right. \\ &\quad \left. + \nabla \zeta \log |\det \mathcal{J}_{T^{-1}}(S_1^{-1}(\eta_b))| - \nabla \zeta \log q(S_1^{-1}(\eta_b)|x) \right. \\ &\quad \left. + 2 \cdot \alpha[q([S_1^{-1}(\eta_b|x)]) - q([S_2^{-1}(\eta_s|x)])] \right\} \nabla \mu_b S_1^{-1}(\eta_b) \\ &= \mathbb{E}_{\mathcal{N}(\eta_b; 0, I)} \left\{ \nabla z_b \log p[x, T_1^{-1}(S_1^{-1}(\eta_b)), T_2^{-1}(S_2^{-1}(\eta_s))] \cdot \nabla \zeta T_1^{-1}(\zeta) \right. \\ &\quad \left. + \nabla \zeta \log |\det \mathcal{J}_{T^{-1}}(\zeta)| - \nabla \zeta \log q(\zeta|x) \right. \\ &\quad \left. + 2 \cdot \alpha[q([S_1^{-1}(\eta_b|x)]) - q([S_2^{-1}(\eta_s|x)])] \right\}. \quad (\text{B.5}) \end{aligned}$$

In a same vein, we can compute the gradients for other three variational parameters:

$$\begin{aligned} \nabla \mu_s(\mathcal{L}) = & \mathbb{E}_{\mathcal{N}(\eta_s; 0, I)} \left\{ \nabla z_s \log p[x, z_s, T_1^{-1}(S_1^{-1}(\eta_b))] \cdot \nabla \beta T_2^{-1}(\beta) + \nabla \beta \log |\det \mathcal{J}_{T^{-1}}(\beta)| \right. \\ & \left. - \nabla \beta \log q(\beta|x) + \nabla \beta 2 \cdot \alpha[q([S_2^{-1}(\eta_s|x)]) - q([S_1^{-1}(\eta_b|x)])] \right\}, \quad (\text{B.6}) \end{aligned}$$

$$\begin{aligned} \nabla \omega(\mathcal{L}) = & \mathbb{E}_{\mathcal{N}(\eta_b; 0, I)} \left\{ \nabla z_b \log p[x, z_b, T_2^{-1}(S_2^{-1}(\eta_s))] \cdot \nabla \zeta T_1^{-1}(\zeta) + \nabla \zeta \log |\det \mathcal{J}_{T^{-1}}(\zeta)| \right. \\ & \left. - \nabla \zeta \log q(\zeta|x) + \nabla \zeta 2 \cdot \alpha[q([S_1^{-1}(\eta_b|x)]) - q([S_2^{-1}(\eta_s|x)])] \right\} \eta_b^T(\exp(\omega)), \quad (\text{B.7}) \end{aligned}$$

$$\begin{aligned} \nabla L(\mathcal{L}) = \mathbb{E}_{\mathcal{N}(\eta_s; 0, I)} \{ & \nabla z_s \log p[x, z_s, T_1^{-1}(S_1^{-1}(\eta_b))] \cdot \nabla \beta T_2^{-1}(\beta) + \nabla \beta \log |\det \mathcal{J}_{T^{-1}}(\beta)| \\ & - \nabla \zeta \log q(\beta|x) + \nabla \beta 2 \cdot \alpha [q([S_2^{-1}(\eta_s|x)]) - q([S_1^{-1}(\eta_b|x)])] \} \eta_s^T. \quad (\text{B.8}) \end{aligned}$$

Obtaining these expectation encapsulated expressions on the gradients of parameters, we can now use the Markov Chain Monte Carlo (MCMC) to simulate these expectation terms [81], i.e., $\mathbb{E}_{q(\eta)}[f(\eta)] = \int f(\eta)q(\eta)d\eta \approx \frac{1}{S} \sum_{s=1}^S f(\eta_s)$.

B.0.3 Pseudo Code for the algorithm

These approximated gradients can be readily fed to some off-the-shelf gradient ascent algorithms, e.g., Adam [77] and rmsprop [82] to update parameters. Pooling all ingredients together, the recipe of learning EVE is summarised in Algorithm 3.

Algorithm 3 the EVE compatible ADVI algorithm

Require: Dataset $X : x^{(i)}$; Model: $p(x, z_b, z_s)$; VSE: $q(z_b, z_s|x; \phi_s, \phi_b)$
 Initialise the parameter: $\mu_b; \mu_s; \omega; L$
 Set the hyper-parameter α
repeat
 while fix μ_s, L **do**
 Draw one sample from η_b , where $\eta_b \sim \mathcal{N}(0, I)$
 Approximate $\nabla_{\mu_b} \mathcal{L}; \nabla_{\omega} \mathcal{L}$ according to Equation B.5 and B.7
 Update the $\mu_b; \omega$ via a off-the-shelf gradient ascent algorithm, e.g., Adam, using the previous approximated gradients
 end while
 while fix μ_b, ω **do**
 Draw one sample from η_s , where $\eta_s \sim \mathcal{N}(0, I)$.
 Approximate $\nabla_{\mu_s} \mathcal{L}; \nabla_L \mathcal{L}$ according to Equation B.6 and B.8
 Update the $\mu_s; L$ via a off-the-shelf gradient ascent algorithm, e.g., Adam, using the previous approximated gradients
 end while
until the elbo is converged to a certain level
Return the optimal $\mu_b; \mu_s; \omega; L$

Publications

- Bai, W., Quan, C., & Luo, W. (2017). Hard Label Relaxation in Biased Pictorial Sentiment Discrimination. *International Journal of Advanced Intelligence*, volume9, Number 4(pp. 507-521). (Published) **[Best Paper Award]**
- Bai, W., Quan, C., & Luo, W. (2017). Label Relaxation – An Efficient Solution to Ameliorate Biased Sentimental Discrimination. 12th IEEE *Proceedings of the Natural Language Processing and Knowledge Engineering*, Chengdu, China, 7-10. (Published)
- Bai, W., Quan, C., & Luo, Z. (2018). Uncertainty Flow Facilitates Zero-Shot Multi-Label Learning in Affective Facial Analysis. *Applied Sciences*, 8(2), 300. (Published)
- Bai, W., Quan, C., & Luo, Z. W. (2019). Adaptive Generative Initialization in Transfer Learning. *Computer and Information Science*, (pp. 63-74). Springer, Chapter 6. (Published)
- Bai, W., Quan, C., & Luo, Z. W. (2018, June). Adaptive Generative Initialization in Transfer Learning. *In International Conference on Computer and Information Science*. (Published)
- Bai, W., Quan, C., & Luo, Z. W. (2019, January). Learning Flexible Latent Representations via Encapsulated Variational Encoder. *In 33rd AAAI (Association for the Advancement of Artificial Intelligence)*. Short Paper.(Accepted)
- Bai, W., Quan, C., & Luo, W. (2019). Data-Driven Dimensional Expression Generation via Encapsulated Variational Auto-Encoders. *Cognitive Computation*. (Under Review)

Doctor Thesis, Kobe University

“Learning Complex and Continuous Affective
Representations of Facial Expressions”, 113 pages

Submitted on January, 25th, 2019

The date of publication is printed in cover of repository
version published in Kobe University Repository Kernel.

© BAI WENJUN
All Right Reserved, 2019