UNIVERSITAS INDONESIA

PEMELAJARAN REPRESENTASI TRANSFORMASI-EKUIVARIAN DENGAN ESTIMASI INFORMASI TIMBAL BALIK BARBER-AGAKOV DAN *INFORMATION NOISE CONTRASTIVE*

TESIS

MARSHAL ARIJONA SINAGA
2006560983

FAKULTAS ILMU KOMPUTER
PROGRAM STUDI MAGISTER ILMU KOMPUTER
DEPOK
2022

UNIVERSITAS INDONESIA

PEMELAJARAN REPRESENTASI TRANSFORMASI-EKUIVARIAN DENGAN ESTIMASI INFORMASI TIMBAL BALIK BARBER-AGAKOV DAN *INFORMATION NOISE CONTRASTIVE*

TESIS
Diajukan sebagai salah satu syarat untuk memperoleh gelar
Magister

MARSHAL ARIJONA SINAGA
2006560983

FAKULTAS ILMU KOMPUTER
PROGRAM STUDI MAGISTER ILMU KOMPUTER
DEPOK
JANUARI 2022

HALAMAN PERSETUJUAN

Judul	:	Pemelajaran Representasi Transformasi-Ekuivarian dengan Estimasi Informasi Timbal Balik Barber-Agakov dan *Information Noise Contrastive*
Nama	:	Marshal Arijona Sinaga
NPM	:	2006560983

Laporan Tesis ini telah diperiksa dan disetujui.

10 Januari 2022

Prof. Basaruddin, Ph.D.	Adila Alfa Krisnadhi, Ph.D
Pembimbing Satu	Pembimbing Dua

HALAMAN PERNYATAAN ORISINALITAS

Tesis ini adalah hasil karya saya sendiri,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan dengan benar.

Nama	:	**Marshal Arijona Sinaga**
NPM	:	**2006560983**
Tanda Tangan	:	
Tanggal	:	**10 Januari 2022**

HALAMAN PENGESAHAN

Tesis ini diajukan oleh	:	
Nama	:	Marshal Arijona Sinaga
NPM	:	2006560983
Program Studi	:	Magister Ilmu Komputer
Judul Tesis	:	Pemelajaran Representasi Transformasi-Ekuivarian dengan Estimasi Informasi Timbal Balik Barber-Agakov dan *Information Noise Contrastive*

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Magister pada Program Studi Magister Ilmu Komputer, Fakultas Ilmu Komputer, Universitas Indonesia.

DEWAN PENGUJI

Pembimbing Satu	:	Prof. Basaruddin, Ph.D.	( )
Pembimbing Dua	:	Adila Alfa Krisnadhi, Ph.D	( )
Penguji	:	Prof. Dr. Eng. Wisnu Jatmiko, S.T., M.Kom.	( )
Penguji	:	Ari Saptawijaya, Ph.D	( )
Penguji	:	Fariz Darari, Ph.D	( )

Ditetapkan di : Depok
Tanggal : 22 Desember 2021

KATA PENGANTAR

Segala puji dan syukur saya panjatkan kepada Tuhan Yesus Kristus karena kasih dan anugerah-Nya saya dapat menyelesaikan tesis ini.

Penulisan tesis ini ditujukan untuk memenuhi salah satu syarat untuk menyelesaikan pendidikan Program Magister Ilmu Komputer, Universitas Indonesia. Penyelesaian tesis ini tidak terlepas dari dukungan beberapa pihak. Karenanya, saya ingin mengucapkan terimakasih kepada pihak - pihak berikut:

1. Prof.Basarrudin, Ph.D dan Bapak Adila Alfa Krisnadhi, Ph.D sebagai dosen pembimbing tesis saya yang selalu sabar membimbing saya dalam menyelesaikan tesis ini.

2. Prof. Wisnu Jatmiko, Bapak Fariz Darari, Ph.D, dan Bapak Ari Saptawijaya, Ph.D selaku dosen penguji yang telah memberikan masukan untuk membuat tesis ini menjadi lebih baik.

3. Tokopedia-UI AI Center yang telah menyediakan sumber daya komputasi untuk keperluan penelitian ini.

4. Orang tua dan adik saya yang memberi dukungan, semangat, dan doa kepada saya dalam menyelesaikan tesis ini.

Saya juga ingin mengucapkan terimakasih kepada setiap pihak yang tidak bisa disebutkan satu per satu yang turut membantu saya menyelesaikan tugas akhir ini. Akhir kata, saya berharap tesis ini dapat memberi kontribusi pada perkembangan ilmu pengetahuan, khususnya di bidang pemelajaran representasi. *Soli Deo gloria*.

Depok, 10 Januari 2022

Marshal Arijona Sinaga

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama	:	Marshal Arijona Sinaga
NPM	:	2006560983
Program Studi	:	Magister Ilmu Komputer
Fakultas	:	Ilmu Komputer
Jenis Karya	:	Tesis

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty Free Right*)** atas karya ilmiah saya yang berjudul:

Pemelajaran Representasi Transformasi-Ekuivarian dengan Estimasi Informasi Timbal Balik Barber-Agakov dan *Information Noise Contrastive*

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyatan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada tanggal : 10 Januari 2022
Yang menyatakan

(Marshal Arijona Sinaga)

ABSTRAK

Nama	:	Marshal Arijona Sinaga
Program Studi	:	Magister Ilmu Komputer
Judul	:	Pemelajaran Representasi Transformasi-Ekuivarian dengan Estimasi Informasi Timbal Balik Barber-Agakov dan *Information Noise Contrastive*
Pembimbing	:	Prof. Basaruddin, Ph.D. dan Adila Alfa Krisnadhi, Ph.D

Jaringan saraf konvolusi telah terbukti bekerja dengan baik pada tugas klasifikasi citra. Hasil tersebut didukung oleh sifat transformasi-ekuivarian yang merupakan salah satu karakteristik jaringan saraf konvolusi. Namun, sifat tersebut hanya terbatas pada transformasi translasi. Penelitian ini memperkenalkan transformasi-ekuivarian variasional (TEV), suatu model representasi tidak terawasi yang bersifat transformasi-ekuivarian terhadap transformasi yang lebih general. Pada pengimplementasiannya, TEV memanfaatkan *predictive-transformation*, suatu model pemelajaran terawasi sendiri yang berperan sebagai bias induktif. Proses optimisasi TEV melibatkan 2 estimasi batas bawah informasi timbal balik: metode estimasi Barber-Agakov (selanjutnya dinamakan model TEVBA) dan *information noise contrastive* (selanjutnya dinamakan model TEVInfoNCE). Model representasi TEV diuji berdasarkan rata-rata rasio kesalahan pada serangkaian tugas klasifikasi citra. Pada penelitian ini, perseptron lapis banyak, K-*nearest neighbor*, dan *multinomial logistic regression* dipilih sebagai *classifier* untuk tugas klasifikasi citra menggunakan *dataset* CIFAR-10 dan STL-10. Hasil menunjukkan TEVBA dan TEVInfoNCE mengungguli model *baseline* untuk masing-masing *classifier* pada kedua *dataset*. Secara spesifik, TEVBA secara konsisten menghasilkan rata-rata rasio kesalahan terendah untuk klasifikasi pada kedua *dataset*.

Kata kunci:
Pemelajaran representasi, Transformasi-ekuivarian, Informasi timbal balik, Barber-Agakov, *Information noise contrastive*

ABSTRACT

Name	:	Marshal Arijona Sinaga
Study Program	:	Magister Ilmu Komputer
Title	:	Transformation-Equivariant Representation Learning with Barber-Agakov and Information Noise Contrastive Mutual Information Estimation
Counsellor	:	Prof. Basaruddin, Ph.D.and Adila Alfa Krisnadhi, Ph.D

Convolution neural network (CNN) has shown promising results on various image classification tasks. One of the reasons due to the ability of CNN to extract representation that is equivariant to transformations. However, the notion only holds for the translation transformation. This research introduces variational transformation equivariant (VTE), a more general unsupervised transformation-equivariant representation model. During the implementation, VTE utilizes the Predictive-transformation, a self-supervised learning model that acts as inductive bias. The optimization of VTE involves two lower bound mutual information methods: Barber-Agakov and information noise contrastive (InfoNCE). The VTE models are evaluated based on the average error rate on image classification tasks on CIFAR-10 and STL-10 datasets. We utilize multi-layer perceptron, K-nearest neighbor, and multinomial logistic regression as the classifiers. Results show VTE with Barber-Agakov and VTE with InfoNCE outperform the baseline model for each classifier on both datasets. Specifically, VTEBA consistently achieves the lowest average error rate for both datasets.

Key words:
Representation learning, *Transformation-equivariant*, Mutual information, Barber-Agakov, Information noise contrastive

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR ISTILAH DAN AKRONIM

Bias Induktif	*Inductive bias*
Bobot	*Weight*
Entropi	*Entropy*
Estimasi batas atas	*Upper bound estimation*
Estimasi batas bawah	*Lower bound estimation*
Fungsi estimator	*Estimator function*
Fungsi objektif	*Objective function*
Inferensi variasional	*Variational Inference*
Informasi timbal balik	*Mutual information*
Jaringan saraf atensi	*Self-attention neural network*
Jaringan saraf berulang	*Recurrent neural network*
Jaringan saraf konvolusi	*Convolution neural network*
Jaringan saraf tiruan	*Artificial neural network*
Jumlah informasi	*Amount of Information*
Lapisan tersembunyi	*Hidden layer*
Model berbasis energi	*Energy based model*
Nilai galat	*Error rate*
Pemelajaran dalam	*Deep learning*
Pemelajaran kontrastif	*Contrastive learning*
Pemelajaran mesin	*Machine learning*
Pemelajaran representasi	*Representation learning*
Pemelajaran terawasi	*Supervised learning*
Pemelajaran terawasi sendiri	*Self-supervised learning*
Pemelajaran tidak terawasi	*Unsupervised learning*
Pengekstraksi fitur	*Feature extractor*
Perseptron lapis banyak	*Multi-layer perceptron*
Rasio densitas	*Density ratio*
Transformasi-ekuivarian	*Transformation-equivariant*
Transformasi-invarian	*Transformation-invariant*
Variabel acak	*Random variable*

Akronim

AET	*Autoencoding transformation*
AVT	*Autoencoding variational transformation*
BA	Barber-Agakov
GAN	*Generative adversarial network*
InfoNCE	*Information noise contrastive estimation*
ITB	Informasi timbal balik
JSK	Jaringan saraf konvolusi
JST	Jaringan saraf tiruan
PLB	Perseptron lapis banyak
TEV	*Transformasi-ekuivarian variasional*
VAE	*Variational autoencoder*
VA	Variabel acak

DAFTAR SIMBOL

Variabel

$\hat{\mathbf{x}}$ Sampel citra asli sebelum ditransformasi

$\mathbf{x}$ Sampel citra hasil transformasi

$\mathbf{t}$ Sampel transformasi citra (direpresentasikan oleh matriks)

$\hat{\mathbf{t}}$ Sampel hasil rekonstruksi transformasi citra (direpresentasikan oleh matriks)

$\hat{\mathbf{z}}$ Representasi citra $\hat{\mathbf{x}}$

$\mathbf{z}$ Representasi citra $\mathbf{x}$

$\mathbf{X}, \mathbf{Z}, \hat{\mathbf{Z}}, \mathbf{T}$ Variabel acak

$\mathcal{X}, \mathcal{T}, \mathcal{Z}$ Ruang sampel $\mathbf{X}, \mathbf{T}$, dan $\mathbf{Z}$

$\mathbf{u}$ Sampel transformasi representasi

$\mathbf{h}_l$ Lapisan ke-l pada perseptron lapis banyak

$\mathbf{b}_l$ Vektor bias perseptron pada lapis ke-l

α *Learning rate*

$\boldsymbol{\mu}$ *Mean* distribusi Gaussian multivariat

σ Standar deviasi distribusi Gaussian multivariat

$\boldsymbol{\theta}$ Parameter *encoder* TEV

$\hat{\boldsymbol{\theta}}$ Parameter *encoder predictive-transformation*

$\check{\boldsymbol{\phi}}$ Parameter *decoder predictive-transformation*

$\boldsymbol{\phi}$ Parameter *decoder* TEVBA

$\hat{\boldsymbol{\phi}}, \tilde{\boldsymbol{\phi}}, \boldsymbol{\phi}'$ Parameter fungsi estimator g, g_1 dan g_2 pada TEVInfoNCE

$\mathbf{B}$ *Mini-batch*

Distribusi Probabilitas

p Distribusi asli

q Distribusi variasional

$\mathcal{N}$ Distribusi normal

Fungsi

$\mathcal{L}, \ell, J$ Fungsi objektif

D *Decoder*

E *Encoder*

f^* Fungsi asli

f_l, f_{PLB} Perseptron lapis banyak

$g_{\hat{\phi}}, g_{1\tilde{\phi}}, g_{2\phi'}$ Fungsi estimator pada TEVInfoNCE

o Fungsi non-linear pada JST seperti Sigmoid, ReLU, dll

r Fungsi transformasi citra yang melibatkan $\hat{\mathbf{x}}$, dan $\mathbf{t}$

s Fungsi transformasi representasi yang melibatkan $\hat{\mathbf{z}}$ dan $\mathbf{u}$

v Fungsi yang memetakan transformasi pada level citra menuju transformasi pada level representasi

Operator

$\circ$ Komposisi fungsi

$\nabla_{\boldsymbol{\theta}}$ Gradien terhadap parameter $\boldsymbol{\theta}$

$\odot$ Perkalian Hadamard, yaitu perkalian vektor/matriks yang sifatnya *element-wise*

H Entropi

I Informasi timbal balik

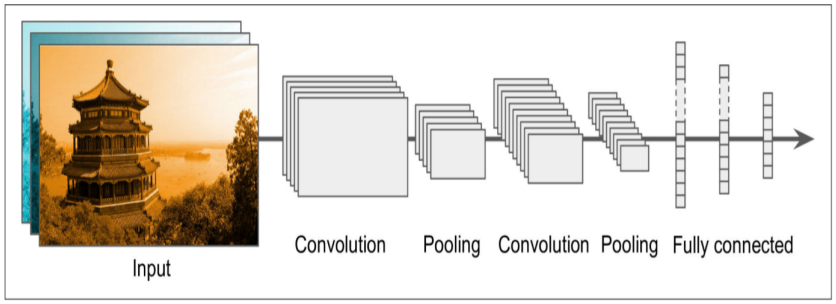
KL Kullback-Leibler *divergence*

$\mathbb{E}$ Ekspektasi

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Pemelajaran dalam (*deep learning*) telah menunjukkan hasil yang menjanjikan untuk klasifikasi citra (He et al., 2016; Krizhevsky et al., 2012; Simonyan and Zisserman, 2015; Szegedy et al., 2015). Secara umum, model klasifikasi citra berbasis pemelajaran dalam terdiri dari dua bagian, yaitu jaringan saraf konvolusional (*convolution neural network*) dan perseptron lapis banyak (*multi-layer perceptron*). Jaringan saraf konvolusional (JSK) bertugas mengekstraksi representasi/fitur yang esensial dari citra. Sementara itu, perseptron lapis banyak (PLB) bertugas untuk memetakan representasi yang dihasilkan JSK menuju suatu kelas/label. Gambar 1.1 menunjukkan ilustrasi pemelajaran dalam. Kesuksesan pemelajaran dalam masing-masing didukung oleh representasi JSK yang bersifat ekuivarian dan luaran PLB yang bersifat invarian terhadap translasi (Cohen and Welling, 2016; Cohen et al., 2018; Hinton et al., 2011; Sabour et al., 2017).



Gambar 1.1: Ilustrasi pemelajaran dalam untuk klasifikasi citra (Géron, 2019). Arsitektur terdiri atas jaringan saraf konvolusi dan perseptron lapis banyak.

Secara umum, representasi citra dikatakan bersifat transformasi-ekuivarian (*transformation-equivariant*) jika terdapat suatu model pemelajaran yang menghasilkan representasi citra sedemikian sehingga representasi tersebut turut berubah seiring dengan transformasi yang diterapkan pada citra tersebut (Qi et al., 2020). Sebagai contoh, misalkan terdapat suatu citra $\hat{\mathbf{x}}$. Selanjutnya, terdapat model pemelajaran yang menerima citra $\hat{\mathbf{x}}$ dan menghasilkan $\hat{\mathbf{z}}$, yaitu representasi citra $\hat{\mathbf{x}}$. Kemudian, misalkan citra $\hat{\mathbf{x}}$ ditransformasi oleh $\mathbf{t}$, yaitu suatu operasi transformasi yang berlaku pada level citra (misalnya rotasi, *cropping*, *scaling*, dll) sehingga dihasilkan citra hasil transformasi $\mathbf{x}$. Lalu, model pemelajaran tersebut juga menerima $\mathbf{x}$ dan menghasilkan $\mathbf{z}$ yang merupakan representasi citra $\mathbf{x}$. Sifat transformasi-ekuivarian memungkinkan $\mathbf{z}$ dapat diperoleh berdasarkan $\hat{\mathbf{z}}$. Secara spesifik, transformasi-ekuivarian mengasumsikan terdapat $\mathbf{u}$, yaitu suatu operasi transformasi yang

berlaku pada level representasi citra. Nilai **u** bersifat unik untuk setiap **t** sedemikian sehingga menerapkan operasi **u** pada $\hat{\mathbf{z}}$ akan menghasilkan **z**. Pada JSK, sifat ekuivarian hanya berlaku untuk translasi. Hal tersebut dikarenakan kernel yang digunakan pada JSK berukuran jauh lebih kecil dibandingkan citra yang masukan. Ukuran kernel yang lebih kecil menyebabkan kernel dapat digeser secara vertikal dan horizontal di sepanjang citra. Pergeseran tersebut secara alamiah menyebabkan JSK mampu menyesuaikan representasi citra seiring dengan translasi yang diterapkan padanya.

Selanjutnya, representasi citra dikatakan bersifat transformasi-invarian (*transformation-invariant*) jika terdapat suatu model pemelajaran yang menghasilkan representasi citra sedemikian sehingga representasi tersebut tidak mengalami perubahan meskipun citra tersebut ditransformasi (Zaheer, 2018). Pada kasus transformasi-invarian, representasi citra $\hat{\mathbf{x}}$ sama dengan representasi citra **x**. Model pemelajaran tidak mempertimbangkan transformasi **t** yang diterapkan pada $\hat{\mathbf{x}}$. Jaringan PLB secara khusus bersifat invarian terhadap translasi (Qi et al., 2019). Sifat tersebut dikarenakan semua komponen luaran PLB melibatkan semua komponen masukan, yang direpresentasikan oleh matriks bobot (*weight*).

Adanya sifat transformasi-ekuivarian memungkinkan model pemelajaran menghasilkan representasi yang *robust* terhadap berbagai tugas yang belum pernah dikerjakan sebelumnya (Qi et al., 2019). Secara intuitif, sifat transformasi-ekuivarian menyebabkan representasi yang dihasilkan mampu mempertahankan informasi yang esensial mengenai objek yang terdapat pada citra. Akibatnya, representasi yang dihasilkan mampu beradaptasi terhadap berbagai transformasi yang diterapkan pada citra. Karena itu, sifat ekuivarian sangat krusial dalam mendukung kinerja pemelajaran dalam. Namun, sifat ekuivarian pada JSK hanya terbatas pada translasi saja. Karena itu, penelitian dilakukan untuk mengembangkan model pemelajaran transformasi-ekuivarian yang tidak terbatas pada transformasi translasi saja.

Cohen and Welling (2016); Lenssen et al. (2018) berusaha mengembangkan varian JSK yang mampu mempertahankan representasi yang bersifat transformasi-ekuivarian untuk beberapa jenis transformasi. Untuk setiap jenis transformasi citra, varian JSK tersebut melibatkan suatu grup transformasi yang memetakan transformasi citra secara unik menuju suatu transformasi representasi. Namun, varian JSK tersebut hanya mampu menangani transformasi yang bersifat diskret. Cohen and Welling (2017) menggeneralisasi penelitian sebelumnya dengan memperkenalkan suatu fungsi yang memetakan transformasi citra menuju transformasi representasi. Namun, pemetaan tersebut bersifat linear sehingga model yang dihasilkan kurang fleksibel. Akibatnya, representasi yang dihasilkan kurang maksimal dalam mengeksploitasi sifat transformasi-ekuivarian (Qi et al., 2020).

Adanya pertumbuhan data tidak teranotasi secara masif mendorong penelitian representasi transformasi-ekuivarian beralih menggunakan pendekatan pemelajaran tidak terawasi (*unsupervised learning*). Penelitian terkini menggunakan suatu *autoencoder* yang dinamakan *autoencoding variational transformation* (AVT). *Autoencoder* tersebut memodelkan fungsi pemetaan yang diajukan Cohen and Welling (2017) dengan

memaksimalkan informasi timbal balik (*mutual information*) antara 3 variabel acak (*random variable*), yaitu citra sebelum transformasi $\hat{\mathbf{X}}$, operasi transformasi $\mathbf{T}$, dan representasi citra hasil transformasi $\mathbf{X}$. Informasi timbal balik (ITB) yang tinggi menunjukkan bahwa terdapat dependensi yang tinggi antara ketiga variabel tersebut, khususnya antara. Secara tidak langsung, dapat dipandang bahwa terdapat fungsi (linear maupun non-linear) yang memetakan relasi ketiga variabel tersebut. Model AVT bertujuan menemukan parameter *autoencoder* $\boldsymbol{\theta}$ yang memaksimalkan ITB $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T})$:

$$\max_{\boldsymbol{\theta}} I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T}) = \max_{\boldsymbol{\theta}} \left(I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}) + I_{\boldsymbol{\theta}}(\mathbf{Z}; \mathbf{T}|\hat{\mathbf{Z}}) \right) \tag{1.1}$$

Pada implementasinya, AVT hanya memaksimalkan $I_{\boldsymbol{\theta}}(\mathbf{Z}; \mathbf{T}|\hat{\mathbf{Z}})$ untuk mengurangi kompleksitas model.

Sampai saat ini, belum ada penelitian lebih lanjut untuk menginvestigasi apakah $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}})$ dapat dijadikan sebagai fungsi objektif (*objective function*) untuk melatih model representasi transformasi-ekuivarian. Secara intuitif, suku $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}})$ juga mengandung informasi mengenai $\mathbf{T}$ melalui $\mathbf{Z}$. Karena alasan tersebut, suku $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}})$ memiliki potensi untuk menghasilkan model pemelajaran yang memiliki kompleksitas lebih ringan tanpa menghilangkan esensi pemelajaran. Namun, percobaan awal menunjukkan bahwa pemelajaran berdasarkan ITB $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}})$ tidak dapat berdiri sendiri. Dibutuhkan suatu model pemelajaran tambahan yang bertindak sebagai bias induktif (*inductive bias*) bagi model tersebut.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang, permasalahan pada penelitian ini dapat dirumuskan sebagai berikut:

1. Bagaimana membangun model pemelajaran yang dapat berperan sebagai bias induktif bagi model alternatif AVT ?
2. Bagaimana mengembangkan model dan algoritma pemelajaran representasi transformasi-ekuivarian melalui fungsi objektif alternatif AVT ? (dengan melibatkan model pemelajaran pada poin sebelumnya)
3. Apakah hasil evaluasi model representasi alternatif pada klasifikasi citra lebih baik dibandingkan dengan AVT ?

1.3 Tujuan dan Kontribusi Penelitian

Penelitian ini bertujuan menginvestigasi model representasi alternatif AVT dengan memaksimalkan ITB $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}})$. Selanjutnya, model ini dinamakan

transformasi-ekuivarian variasional (TEV). Pada pengimplementasiannya, TEV memanfaatkan *predictive-transformation*, yaitu suatu model pemelajaran terawasi sendiri (*self-supervised learning*) yang berperan sebagai bias induktif bagi TEV. Sama seperti AVT, TEV memanfaatkan metode estimasi batas bawah untuk memaksimalkan ITB. Pada penelitan ini, $I_{\boldsymbol{\theta}}(\mathbf{Z};\hat{\mathbf{Z}})$ diestimasi menggunakan metode estimasi batas bawah ITB Barber-Agakov (BA) dan metode *information noise contrastive estimation* (InfoNCE) (Ma and Collins, 2018; Oord et al., 2018). Model yang diajukan kemudian diuji melalui serangkaian klasifikasi citra. Metode evaluasi ini mengikuti (Qi et al., 2019; Zhang et al., 2019) untuk mendapatkan hasil yang komparatif. Kontribusi dari penelitian ini adalah sebagai berikut:

- Penelitian ini memperkenalkan *predictive-transformation*, suatu model pemelajaran terawasi sendiri yang berperan sebagai bias induktif untuk melatih model representasi alternatif AVT.
- Penelitian ini mengajukan suatu model representasi alternatif AVT yang dinamakan transformasi-ekuivarian variasional, beserta dengan algoritma pemelajarannya. Model yang diajukan memanfaatkan metode estimasi batas bawah ITB Barber-Agakov (TEVBA) dan *information noise contrastive estimation* (TEVInfoNCE).
- Penelitian ini mengevaluasi model representasi TEV melalui serangkaian klasifikasi citra.

1.4 Ruang Lingkup Penelitian

Berikut ini merupakan batasan-batasan yang ditetapkan pada penelitian ini:

- Penelitian ini menggunakan *balanced dataset*, artinya proporsi jumlah data untuk setiap kelas/label sama. Berdasarkan alasan tersebut, eksperimen pada penelitian ini menggunakan *benchmark dataset* CIFAR-10 dan STL-10.
- Penelitian ini melibatkan dua metode estimasi batas bawah (*lower bound estimation*) ITB, yaitu estimasi Barber-Agakov dan *information noise contrastive estimation*.

1.5 Sistematika Penulisan

Tesis ini disusun dengan sistematika sebagai berikut. **Bab 2** menjelaskan literatur dan juga landasan teori yang berkaitan dengan penelitian ini. Literatur mencakup definisi dari ITB dan metode estimasinya, penjelasan singkat mengenai PLB dan JSK, serta pembahasan mengenai model representasi tidak terawasi. Model pemelajaran representasi TEV dibahas secara lengkap pada **Bab 3**. **Bab 4** berisi penjelasan lengkap mengenai pengaturan dan

hasil eksperimen dari TEV. Tesis ini ditutup oleh **Bab 5** yang berisi kesimpulan dan peluang mengenai penelitian mendatang. Kode sumber penelitian ini dapat diakses di https://github.com/MarshalArijona/VTE.

BAB 2
LANDASAN TEORI DAN PENELITIAN TERKAIT

Bab ini membahas konsep-konsep dasar (definisi ITB dan JST) yang dibutuhkan untuk mengerjakan penelitian ini. Selain itu, dibahas pula penelitian terkait estimasi ITB dan model representasi tidak terawasi.

2.1 Informasi Timbal Balik (ITB)

Sebelum membahas ITB, terlebih dahulu dijelaskan mengenai jumlah informasi, entropi, dan Kullback-Leibler *divergence* sebagai dasar perhitungan ITB. Setelahnya, dijelaskan mengenai definisi dan metode estimasi ITB.

Misalkan terdapat suatu VA $\mathbf{X}$ dengan probabilitas $p(\mathbf{x})$ untuk $\mathbf{X} = \mathbf{x}$. Secara matematis, jumlah informasi (*amount of information*) untuk $\mathbf{X} = \mathbf{x}$ dapat dituliskan sebagai $I(\mathbf{x}) = -\log p(\mathbf{x})$. Jumlah informasi dapat dipandang sebagai tingkat ketidakpastian $\mathbf{X} = \mathbf{x}$ (MacKay, 2003). Jumlah informasi bernilai tinggi jika nilai $\mathbf{x}$ yang muncul memiliki probabilitas $p(\mathbf{x})$ yang rendah. Sebaliknya, jika nilai $\mathbf{x}$ yang muncul memiliki probabilitas $p(\mathbf{x})$ yang tinggi, maka jumlah informasi menjadi rendah (Bishop, 2006). Satuan dari jumlah informasi bergantung pada basis logaritma yang digunakan. Variabel acak biner umumnya menggunakan basis 2 sehingga satuannya disebut sebagai bit. Basis lain yang umum digunakan adalah bilangan natural e yang satuannya disebut sebagai nat.

Ekspektasi jumlah informasi suatu VA disebut sebagai entropi (*entropy*) (Bishop, 2006). Secara matematis, entropi VA $\mathbf{X}$ dapat dituliskan sebagai $H(\mathbf{X}) = \mathbb{E}_{p(\mathbf{x})}[I(\mathbf{x})] = \mathbb{E}_{p(\mathbf{x})}[-\log p(\mathbf{x})]$. Untuk VA diskret dengan semua kemungkinan $\mathbf{x}_1, \cdots, \mathbf{x}_k, \cdots, \mathbf{x}_K$, entropi $\mathbf{X}$ didapat dengan menghitung $-\sum_{k=1}^{K} p(\mathbf{x}_k) \log p(\mathbf{x}_k)$. Untuk VA kontinu dengan *support* $\mathcal{X}$, nilai ekspektasi didapat dengan menghitung integral $-\int_{\mathcal{X}} p(\mathbf{x}) \log p(\mathbf{x}) d\mathbf{x}$.

Selanjutnya, Kullback-Leibler (KL) *divergence* $p(\mathbf{x})$ menuju suatu distribusi $q(\mathbf{x})$ didefinisikan sebagai rata-rata informasi yang dibutuhkan untuk memetakan $\mathbf{x}$ yang berasal dari $p(\mathbf{x})$ menggunakan distribusi $q(\mathbf{x})$ (Bishop, 2006). Secara matematis, KL *divergence* dapat dituliskan sebagai:

$$KL(p\,||\,q) = \mathbb{E}_{p(\mathbf{x})}\left[\log \frac{p(\mathbf{x})}{q(\mathbf{x})}\right] \tag{2.1}$$

Untuk VA diskret dengan semua kemungkinan $\mathbf{x}_1, \cdots, \mathbf{x}_k, \cdots, \mathbf{x}_K$, KL *divergence* $KL(p\,||\,q)$ dituliskan sebagai:

$$KL(p||q) = \sum_{k=1}^{K} p(\mathbf{x}_k) \log \frac{p(\mathbf{x}_k)}{q(\mathbf{x}_k)} \qquad \text{(Diskret)} \tag{2.2}$$

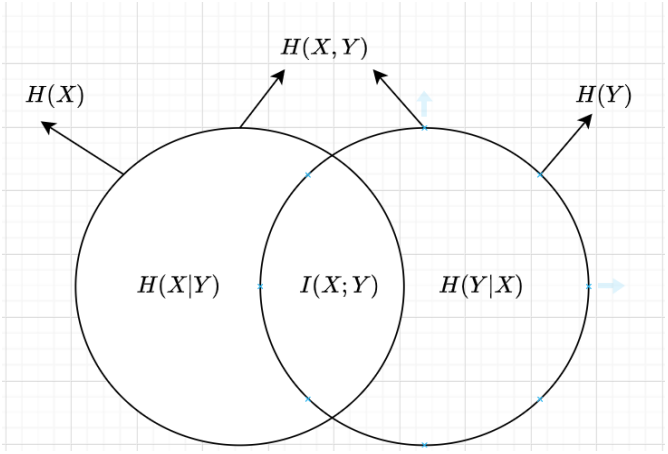
Untuk VA kontinu dengan *support* $\mathcal{X}$, KL *divergence* $KL(p||q)$ dituliskan sebagai:

$$KL(p||q) = \int_{\mathcal{X}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})} d\mathbf{x} \qquad \text{(Kontinu)} \tag{2.3}$$

KL *divergence* tidak memenuhi sifat komutatif, yaitu $KL(p||q) \neq KL(q||p)$ (Bishop, 2006). Selanjutnya, menggunakan ketidaksamaan Jensen dapat ditunjukkan bahwa $KL(p||q) \geq 0$ selalu terpenuhi (Bishop, 2006; MacKay, 2003). Ketika $p(\mathbf{x}) = q(\mathbf{x})$, maka $KL(p||q) = 0$. KL *divergence* dapat digunakan untuk mengukur perbedaan statistik antara dua distribusi probabilitas.

2.1.1 Definisi Informasi Timbal Balik

Misalkan terdapat distribusi *joint* $p(\mathbf{x},\mathbf{y})$ antara VA $\mathbf{X}$ dan $\mathbf{Y}$. Jika kedua VA saling independen, maka $p(\mathbf{x},\mathbf{y}) = p(\mathbf{x})p(\mathbf{y})$. Namun, jika kedua VA tidak independen, diperlukan suatu pengukuran untuk menghitung tingkat dependensi dari kedua VA tersebut. Hubungan antara dua VA yang tidak saling independen dapat digambarkan dalam bentuk diagram Venn seperti yang ditunjukkan pada Gambar 2.1. Area lingkaran yang saling tumpang tindih merupakan ITB $I(\mathbf{X};\mathbf{Y})$. Area lingkaran utuh di sebelah kiri merupakan entropi *marginal* $H(\mathbf{X})$. Area lingkaran di sebelah kiri tanpa area yang tertindih merupakan entropi bersyarat $H(\mathbf{X}|\mathbf{Y})$. Area lingkaran utuh di sebelah kanan merupakan entropi *marginal* $H(\mathbf{Y})$. Area lingkaran di sebelah kanan tanpa area yang tertindih merupakan entropi *bersyarat* $H(\mathbf{Y}|\mathbf{X})$. Area kedua lingkaran tanpa area yang tertindih merupakan entropi *joint* $H(\mathbf{X};\mathbf{Y})$.



Gambar 2.1: Skema diagram Venn yang menunjukkan relasi antara informasi yang dimiliki oleh VA **X** dan **Y**.

Tingkat dependensi antara dua VA dapat diukur menggunakan KL-*divergence* antara distribusi *joint* $p(\mathbf{x},\mathbf{y})$ dan perkalian antara distribusi *marginal* $p(\mathbf{x})$ dan $p(\mathbf{y})$. Secara matematis, dapat ditulis:

$$I(\mathbf{X};\mathbf{Y}) = KL(p(\mathbf{x},\mathbf{y}) \,||\, p(\mathbf{x})\, p(\mathbf{y})) \tag{2.4}$$

Persamaan 2.4 kemudian didefinisikan sebagai informasi timbal balik (*mutual information*) antara **X** dan **Y** (Bishop, 2006). Karena $KL(.\,\|\,.) \geq 0$, maka berlaku $I(\mathbf{X};\mathbf{Y}) \geq 0$. Ketika $I(\mathbf{X};\mathbf{Y}) = 0$, maka VA **X** dan **Y** saling independen. Informasi timbal balik dapat pula dituliskan dalam bentuk entropi. Berdasarkan Gambar 2.1, diperoleh:

$$I(\mathbf{X};\mathbf{Y}) = H(\mathbf{X}) + H(\mathbf{Y}) - H(\mathbf{X},\mathbf{Y}) \tag{2.5}$$
$$= H(\mathbf{Y}) - H(\mathbf{Y}|\mathbf{X}) \tag{2.6}$$
$$= H(\mathbf{X}) - H(\mathbf{X}|\mathbf{Y}) \tag{2.7}$$

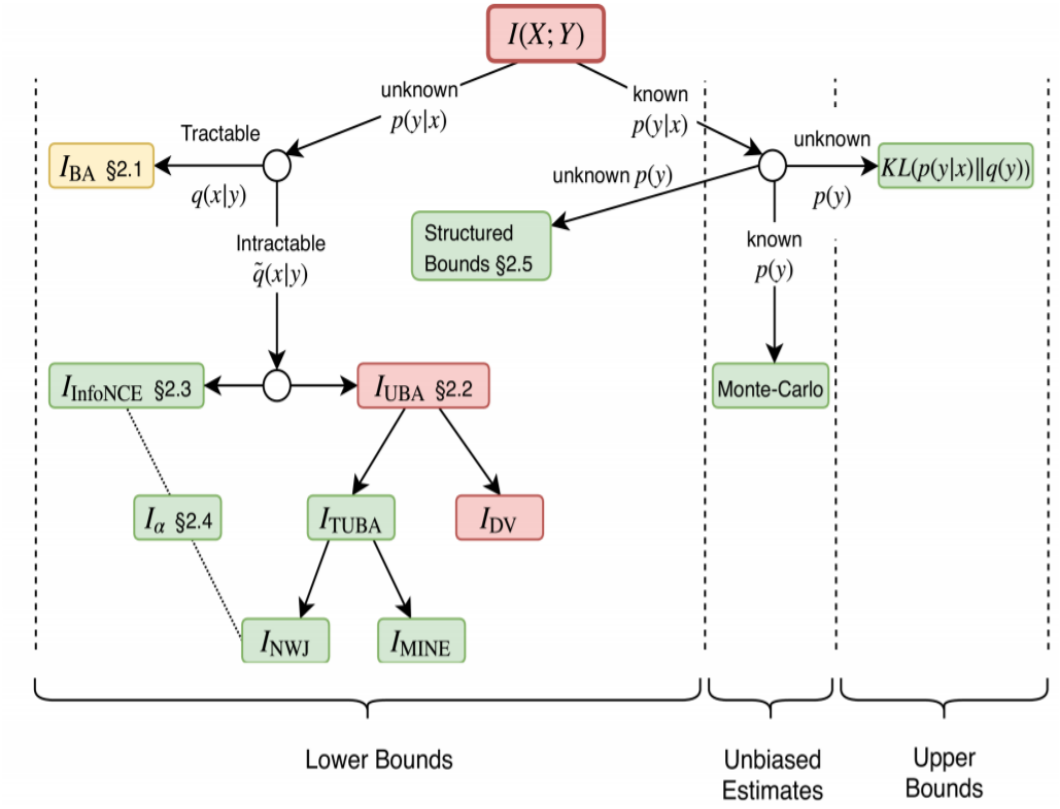
dengan $H(\mathbf{X})$ dan $H(\mathbf{Y})$ merupakan entropi *marginal* **X** dan **Y**, $H(\mathbf{Y}|\mathbf{X})$ dan $H(\mathbf{X}|\mathbf{Y})$ merupakan entropi bersyarat, dan $H(\mathbf{X},\mathbf{Y})$ merupakan entropi *joint* **X** dan **Y**. Kemudian, Persamaan 2.4 dapat dipandang sebagai ekspektasi terhadap $p(\mathbf{x},\mathbf{y})$. Dengan memanfaatkan definisi probilitas bersyarat, diperoleh:

$$I(\mathbf{X};\mathbf{Y}) = \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{x})p(\mathbf{y})}\right] = \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{x}|\mathbf{y})}{p(\mathbf{x})}\right] = \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{y}|\mathbf{x})}{p(\mathbf{y})}\right] \tag{2.8}$$

dengan $p(\mathbf{x}|\mathbf{y})$ merupakan probabilitas bersyarat **x** diberikan **y** dan $p(\mathbf{y}|\mathbf{x})$ merupakan probabilitas bersyarat **y** diberikan **x**. Berdasarkan Persamaan 2.8, dapat disimpulkan bahwa $I(\mathbf{X};\mathbf{Y}) = I(\mathbf{Y};\mathbf{X})$.

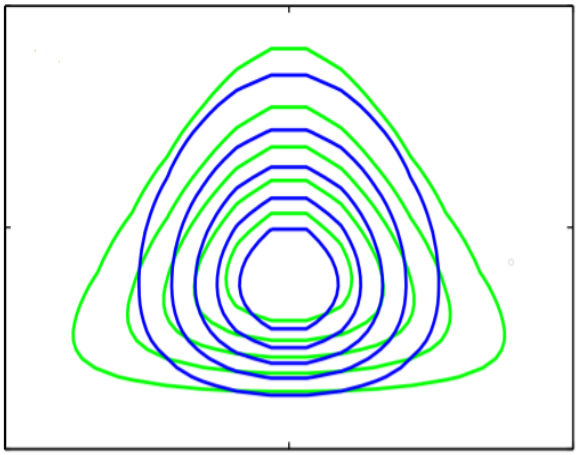
2.1.2 Estimasi Informasi Timbal Balik

Pada kenyataannya, perhitungan ITB secara langsung tidak selalu dapat dilakukan. Umumnya, akses $p(\mathbf{y}|\mathbf{x})$, $p(\mathbf{x})$, atau $p(\mathbf{y})$ tidak tersedia. Pada permasalahan pemelajaran mesin, sering kali hanya tersedia sampel data dari distribusi probabilitas tertentu. Penelitian dilakukan untuk mengestimasi ITB. Menurut Poole et al., 2019, metode estimasi ITB dapat dibagi menjadi 3 kelompok utama: estimasi batas bawah (*upper bound estimation*), estimasi batas atas (*lower bound estimation*), dan estimasi non-bias. Bagan pengelompokan metode estimasi ITB dapat dilihat pada Gambar 2.2. Bab ini menjelaskan metode estimasi batas bawah ITB Barber-Agakov (BA) (Agakov, 2004) dan *information noise contrastive estimation* (InfoNCE) (Oord et al., 2018), serta metode estimasi batas atas *variational bottleneck* (Alemi et al., 2017; Poole et al., 2019).



Gambar 2.2: Bagan yang mengelompokkan metode estimasi ITB menjadi metode estimasi batas atas, estimasi batas bawah, dan estimasi non-bias berdasarkan probabilitas yang terlibat (Poole et al., 2019).

Estimasi ITB BA memanfaatkan prinsip inferensi variasional (*variational inference*) yang bertujuan mengaproksimasi suatu distribusi menggunakan suatu distribusi yang dinamakan distribusi variasional. Distribusi variasional umumnya memiliki struktur yang lebih sederhana dibandingkan dengan distribusi yang ingin diestimasi Gambar 2.3 menunjukkan ilustrasi inferensi variasional.



Gambar 2.3: Ilustrasi yang menunjukkan inferensi variasional menurut Bishop, 2006. Distribusi asli ditunjukkan oleh kontur berwarna hijau sementara distribusi pengganti ditunjukkan oleh kontur berwarna biru. Distribusi variasional/pengganti dipilih sedemikian sehingga selisih area kedua kontur seminimal mungkin.

Estimasi batas bawah BA digunakan ketika $p(\mathbf{x}|\mathbf{y})$ tidak tersedia. Distribusi $p(\mathbf{x}|\mathbf{y})$ dapat diaproksimasi oleh distribusi variasional $q(\mathbf{x}|\mathbf{y})$ (Agakov, 2004). Penggunaan distribusi $q(\mathbf{x}|\mathbf{y})$ menghasilkan *gap* terhadap nilai ITB asli. *Gap* terhadap ITB asli dinyatakan oleh KL *divergence* antara distribusi asli $p(\mathbf{x}|\mathbf{y})$ dan $q(\mathbf{x}|\mathbf{y})$. Berdasarkan definisi ITB, estimasi batas bawah BA I_{BA} diturunkan sebagai berikut:

$$\begin{aligned} I(\mathbf{X};\mathbf{Y}) &= \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{x}|\mathbf{y})}{p(\mathbf{x})}\right] \\ &= \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{q(\mathbf{x}|\mathbf{y})}{p(\mathbf{x})}\right] + \mathbb{E}_{p(\mathbf{y})}\left[KL(p(\mathbf{x}|\mathbf{y}) \,||\, q(\mathbf{x}|\mathbf{y}))\right] \\ &\geq \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log q(\mathbf{x}|\mathbf{y})\right] + H(\mathbf{X}) \end{aligned} \tag{2.9}$$

dengan $H(\mathbf{X})$ menyatakan entropi *marginal* $\mathbf{X}$. Selanjutnya, estimasi batas bawah BA didefinisikan sebagai $I_{\text{BA}} = \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log q(\mathbf{x}|\mathbf{y})\right] + H(\mathbf{X})$. Semakin kecil nilai KL *divergence* antara $p(\mathbf{x}|\mathbf{y})$ dan $q(\mathbf{x}|\mathbf{y})$, maka semakin dekat nilai estimasi ITB dan berlaku juga sebaliknya. Estimasi batas bawah BA optimal ketika $p(\mathbf{x}|\mathbf{y}) = q(\mathbf{x}|\mathbf{y})$. Kesamaan tersebut menyebabkan $KL(q(\mathbf{x}|\mathbf{y})||p(\mathbf{x}|\mathbf{y})) = 0$. Aplikasi estimasi batas bawah BA diantaranya sebagai fungsi objektif (*objective function*) model pemelajaran representasi AVT (Qi et al., 2019) dan regulator pada *information generative adversarial network* (InfoGAN) (Chen et al., 2016).

Metode estimasi selanjutnya adalah estimasi batas bawah InfoNCE (Oord et al., 2018; Poole et al., 2019). Untuk suatu ITB $I(\mathbf{X};\mathbf{Y})$, metode ini mengestimasi rasio densitas (*density ratio*) $\frac{p(\mathbf{x}|\mathbf{y})}{p(\mathbf{x})}$ diberikan K pasang sampel $\{\mathbf{x}^{(j)},\mathbf{y}^{(j)}\}_{j=1}^{K}$. Rasio densitas tersebut diestimasi melalui suatu fungsi estimator (*estimator function*) f, yang luarannya

menjadi faktor pemangkatan bilangan natural e. Selanjutnya, estimasi rasio densitas untuk suatu pasangan $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ diekspresikan sebagai $\frac{p(\mathbf{x}|\mathbf{y})}{p(\mathbf{x})} \propto \exp(f(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}))$, dengan $\exp(f(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})) = e^{f(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})}$. Perlu diketahui bahwa estimasi rasio densitas tersebut tidak ternormalisasi, artinya $\int_{\mathcal{X}} \exp(f(\mathbf{x}, \mathbf{y}))d\mathbf{x} > 1$ dengan $\mathcal{X}$ menyatkan *support* VA $\mathbf{X}$. Upaya normalisasi dilakukan dengan memanfaatkan prinsip pemelajaran kontrastif (*contrastive learning*), yaitu dengan membagi setiap pasangan positif $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ terhadap seluruh pasangan negatif $(\mathbf{x}^{(j)}, \mathbf{y}^{(i)})$ untuk $1 \leq i, j \leq K$. Secara matematis, estimasi batas bawah InfoNCE I_{InfoNCE} dapat dituliskan sebagai:

$$I_{\text{InfoNCE}} = \mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K} \log \frac{\exp(f(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}))}{\sum_{j=1}^{K} \exp(f(\mathbf{x}^{(j)}, \mathbf{y}^{(i)}))}\right] \tag{2.10}$$

dengan ekspektasi terhadap $\prod_j p(\mathbf{x}^{(j)}, \mathbf{y}^{(j)})$. Bentuk estimasi batas bawah InfoNCE memiliki kemiripan dengan fungsi *softmax* dan model berbasis energi (*energy based model*) (Goodfellow et al., 2016). Estimasi ini memiliki variansi yang rendah namun memiliki bias yang cenderung tinggi (Poole et al., 2019). Fungsi estimator $f(\mathbf{x}, \mathbf{y})$ optimal ketika $f(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}|\mathbf{y}) + c(\mathbf{x})$ dengan $c(\mathbf{x})$ merupakan suatu fungsi yang hanya terikat pada $\mathbf{x}$ (Poole et al., 2019). Estimasi batas bawah InfoNCE memenuhi pertidaksamaan $I(\mathbf{X}; \mathbf{Y}) \geq \log(K) + I_{\text{InfoNCE}}$ (Oord et al., 2018). Pertidaksamaan tersebut mengindikasikan bahwa estimasi batas bawah InfoNCE bergantung pada jumlah sampel yang dilibatkan. Fungsi estimator $f(\mathbf{x}, \mathbf{y})$ dapat diganti dengan $p(\mathbf{x}|\mathbf{y})$ jika akses terhadap probabilitas tersebut diketahui. Estimasi InfoNCE tersebut dituliskan sebagai:

$$I_{\text{InfoNCE}} = \mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K} \log \frac{p(\mathbf{x}^{(i)}|\mathbf{y}^{(i)})}{\sum_{j=1}^{K} p(\mathbf{x}^{(j)}|\mathbf{y}^{(i)})}\right]$$

dengan ekspektasi terhadap $\prod_j p(\mathbf{x}^{(j)}, \mathbf{y}^{(j)})$.

Estimasi ITB yang terakhir adalah estimasi batas atas *variational bottleneck* (Alemi et al., 2017; Poole et al., 2019). Estimasi ini menggunakan pendekatan yang sama dengan estimasi batas bawah BA hanya saja menghasilkan nilai estimasi yang nilainya dijamin lebih besar sama dengan nilai ITB yang sebenarnya. Estimasi ini mengasumsikan bahwa terdapat akses terhadap $p(\mathbf{x}|\mathbf{y})$, namun tidak terdapat akses terhadap $p(\mathbf{y})$. Akibatnya, distribusi pengganti $q(\mathbf{y})$ dibutuhkan untuk mengestimasi $p(\mathbf{y})$. Estimasi batas atas *variational*

bottleneck I_{VB} dapat dituliskan sebagai:

$$\begin{aligned} I(\mathbf{X};\mathbf{Y}) &= \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{y}|\mathbf{x})}{p(\mathbf{y})}\right] \\ &= \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{y}|\mathbf{x})\,q(\mathbf{y})}{p(\mathbf{y})\,q(\mathbf{y})}\right] \\ &= \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{y}|\mathbf{x})}{q(\mathbf{y})}\right] - KL(p(\mathbf{y})\,||\,q(\mathbf{y})) \\ &\leq \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{y}|\mathbf{x})}{q(\mathbf{y})}\right] \end{aligned} \tag{2.11}$$

Selanjutnya, estimasi batas atas *variational bottleneck* dapat dituliskan sebagai $I_{\text{VB}} = \mathbb{E}_{p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{y}|\mathbf{x})}{q(\mathbf{y})}\right]$. Estimasi tersebut akan optimal apabila $p(\mathbf{y}) = q(\mathbf{y})$. Kesamaan tersebut menyebabkan $KL(p(\mathbf{y})\,||\,q(\mathbf{y})) = 0$. Namun, mengestimasi $p(\mathbf{y})$ tanpa mengetahui karakteristik distribusi $p(\mathbf{y})$ akan menyebabkan estimasi menjadi sulit, khususnya jika dimensionalitas $\mathbf{y}$ tinggi (Cheng et al., 2020). Metode estimasi ini diaplikasikan untuk fungsi objektif *deep variational information bottleneck* (Alemi et al., 2017). Pada penelitian tersebut, dipilih distribusi standar Gaussian $q(\mathbf{y}) = \mathcal{N}(\mathbf{y};\mathbf{0},\mathbb{I})$.

Berikut metode lain yang dapat mengestimasi ITB. *Contrastive log-ratio upper bound* (CLUB) menggunakan prinsip pemelajaran kontrastif untuk menurunkan batas atas ITB (Cheng et al., 2020). Sementara itu, estimasi batas atas L1-*out* mengubah bentuk estimasi BA sebagai suatu estimasi Monte-Carlo (Poole et al., 2019). Estimasi Nguyen-Wright-Jordan (NWJ) melibatkan bentuk variasional dari KL divergence untuk menurunkan batas bawah ITB (Poole et al., 2019). *Mutual information neural estimation* (MINE) mengestimasi ITB melalui representasi Donsker-Varadhan (Belghazi et al., 2018). Estimasi *smoothed mutual information lower bound* (SMILE) menggunakan teknik *clipping* untuk mengurangi variansi dari estimasi ITB NWJ (Song and Ermon, 2020).

2.2 Jaringan Saraf Tiruan (JST)

Bagian ini membahas JST dan 2 contoh model JST yaitu perseptron lapis banyak (PLB) dan jaringan saraf konvolusional (JSK). Pada bagian ini dijelaskan 2 model jaringan saraf tiruan yaitu perseptron lapis banyak (PLB) dan jaringan saraf konvolusional (JSK). Terdapat model jaringan saraf tiruan lain seperti jaringan saraf berulang (*recurrent neural network*) dan jaringan saraf berbasis atensi (*attention neural network*) (Goodfellow et al., 2016).

Jaringan saraf tiruan (*artificial neural network*) digunakan untuk mengaproksimasi sembarang fungsi $f^*(\mathbf{x})$ dengan $\mathbf{x}$ merupakan masukan bagi fungsi f^*. Kemudian, JST mendefinisikan suatu fungsi $f_{\text{JST}}(\mathbf{x},\boldsymbol{\theta})$ dengan $\boldsymbol{\theta}$ merepresentasikan parameter dari JST. Selanjutnya, JST mengoptimisasi $\boldsymbol{\theta}$ sedemikian sehingga fungsi f_{JST} dapat mengaproksimasi f^* seakurat mungkin. Karena memiliki kapasitas untuk mengaproksimasi semua jenis fungsi, JST dikatakan sebagai aproksimator universal (Goodfellow et al., 2016).

Proses optimisasi pada JST untuk permasalahan regresi maupun klasifikasi mengikuti

prinsip *empirical risk minimization* (Goodfellow et al., 2016). Misalkan terdapat himpunan data pemelajaran $\mathbf{X} = \{\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(K)}\}$. Selanjutnya, didefinisikan risk J sebagai kesalahan antara luaran fungsi f_{JST} terhadap luaran f^*. Optimisasi JST berusaha mencari nilai $\boldsymbol{\theta}$ sedemikian sehingga ekspektasi *risk* J terhadap $\mathbf{X}$ seminimal mungkin. Nilai $\boldsymbol{\theta}$ umumnya dioptimisasi menggunakan metode iteratif *batch gradient descent* (GD) yang melibatkan $\mathbf{x}^{(i)} \in \mathbf{X}$ untuk $i = 1, ..., K$. Pada aplikasinya, optimisasi yang digunakan sering kali merupakan *minibatch stochastic gradient descent* (SGD) yang membagi $\mathbf{X}$ kedalam *mini batch* $\mathbf{B} = \{\mathbf{B}^{(1)}, \cdots, \mathbf{B}^{(N)}\}$. Metode ini mengoptimisasi model untuk setiap *mini batch* berdasarkan *risk* $J^{(i)}$ pada *batch* ke-i. Untuk suatu *time-step* t pada *mini-batch* ke-i, optimisasi θ melalui SGD didefinisikan sebagai $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)} - \eta \nabla_{\boldsymbol{\theta}} J^{(i)}$ dengan η menyatakan *learning rate*. Notasi $-\nabla_{\boldsymbol{\theta}} J^{(i)}$ menyatakan gradien J terhadap parameter $\boldsymbol{\theta}$. Gradien tersebut menyatakan arah perubahan terbesar terhadap penurunan nilai $J^{(i)}$. Notasi η menyatakan skala perubahan $\boldsymbol{\theta}$ (Bishop, 2006). Nilai $\nabla_{\boldsymbol{\theta}} J^{(i)}$ didapat dengan melakukan *back-propagation* yaitu menghitung turunan fungsi $J^{(i)}$ terhadap setiap komponen parameter $\boldsymbol{\theta}$. Turunan untuk setiap komponen parameter didapat menggunakan sifat *chain-rule*.

2.2.1 Perseptron Lapis Banyak (PLB)

Perseptron lapis banyak (*multi-layer perceptron*) dapat dipandang sebagai komposisi $L \in \mathbb{N}$ fungsi

$$f_{\text{PLB}}(\mathbf{x}, \boldsymbol{\theta}) = (f_L \circ f_{L-1} \circ ... \circ f_1)(\mathbf{x})$$

untuk setiap f_l merupakan suatu fungsi yang terparameterisasi oleh $\boldsymbol{\theta}_l = \{\mathbf{W}_l, \mathbf{b}_l\}$ yaitu matriks bobot (*weight*) dan bias pada lapisan ke-l. Secara matematis, untuk setiap l berlaku:

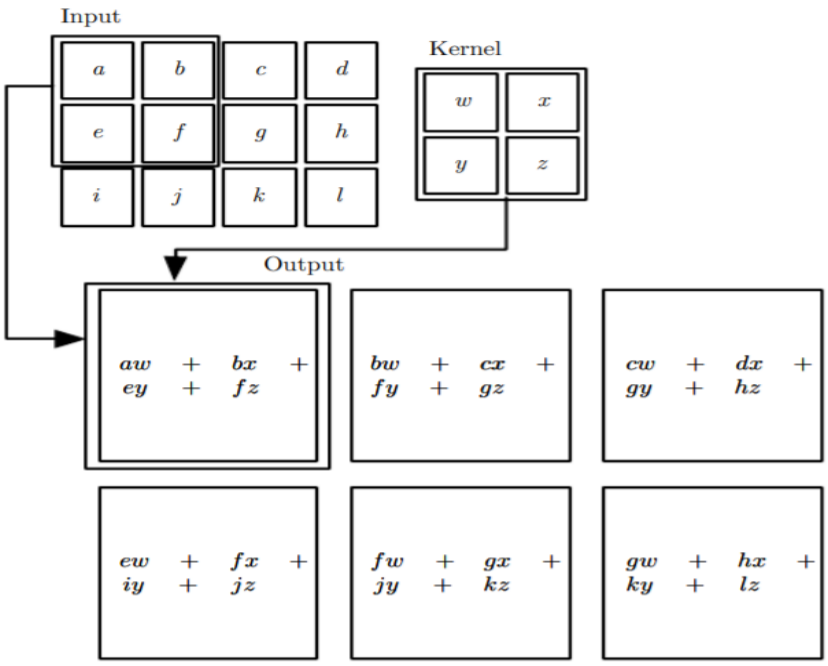
$$f_l(\mathbf{h}_{l-1}) = \mathbf{h}_l = o(\mathbf{W}_l^T \mathbf{h}_{l-1} + \mathbf{b}_l)$$

dengan $\mathbf{h}_0 = \mathbf{x}$ dan o merupakan suatu fungsi non-linear, seperti *rectified linear unit* $o(\mathbf{x}^{(i)}) = \max(0, \mathbf{x}^{(i)})$ (ReLU) (Nair and Hinton, 2010). Lapisan $\mathbf{h}_L$ merupakan keluaran akhir dari PLB. Untuk permasalahan klasifikasi, umumnya lapisan $\mathbf{h}_L$ menggunakan fungsi non-linear *softmax* $o(\mathbf{x}^{(i)}) = e^{\mathbf{x}^{(i)}} / \sum_j e^{\mathbf{x}^{(j)}}$ (klasifikasi biner) atau fungsi Sigmoid $o(\mathbf{x}^{(i)}) = 1/(1 + e^{-\mathbf{x}^{(i)}})$ (klasifikasi multi-kelas), sementara untuk permasalahan regresi tidak menggunakan fungsi non-linear sama sekali (Worrall, 2019). Nilai L menyatakan jumlah fungsi yang membentuk f_{PLB} yang disebut sebagai kedalaman. Kemudian dimensionalitas $\mathbf{h}_1, \cdots, \mathbf{h}_{L-1}$ disebut sebagai lebar setiap lapisan jaringan. Kedalaman dan lebar kemudian merepresentasikan ukuran dari suatu PLB (Kristiadi, 2019).

2.2.2 Jaringan Saraf Konvolusional (JSK)

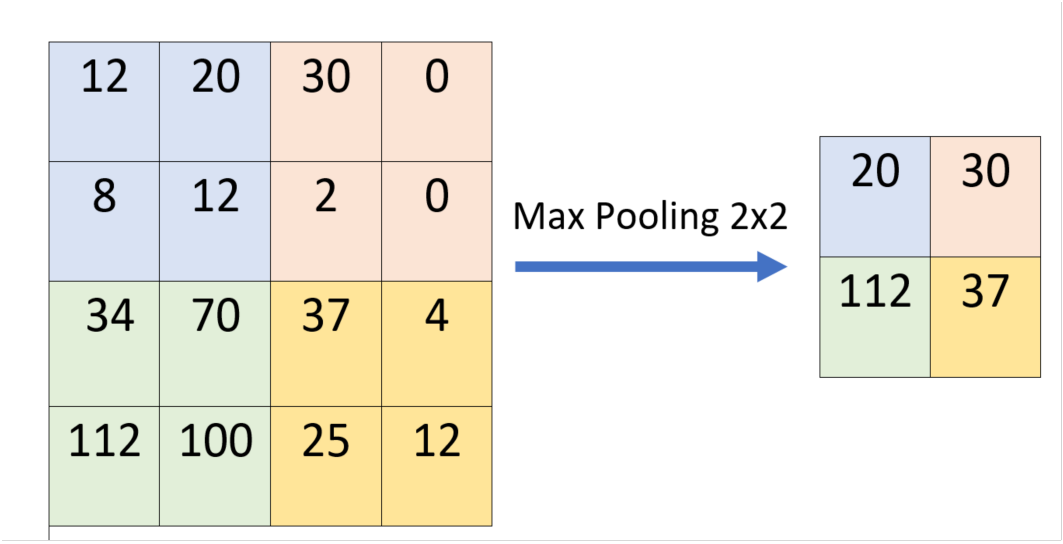
Jaringan saraf konvolusional (*convolution neural network*) merupakan JST yang menggunakan operasi konvolusi sebagai pengganti operasi perkalian matriks bobot pada setidaknya salah satu lapisannya (Goodfellow et al., 2016). Pada operasi konvolusi, matriks bobot digantikan oleh suatu matriks yang berukuran relatif lebih kecil dibandingkan ukuran data masukan yang disebut sebagai kernel. Kernel tersebut kemudian digeser di sepanjang area data masukan. Kernel tersebut dapat dipandang sebagai grid yang memindai suatu data yang juga berbentuk grid namun dengan ukuran yang lebih besar.

Operasi konvolusi pada JSK merupakan operasi perkalian linear antara kernel terhadap area masukan yang sedang ditempati oleh kernel tersebut. Gambar 2.4 menunjukkan ilustrasi operasi konvolusi pada JSK. Terdapat perbedaan antara operasi konvolusi pada JSK dan operasi konvolusi yang umumnya digunakan pada pengolahan sinyal. Operasi konvolusi pada JSK tidak memutar kernel seperti operasi konvolusi biasa. Pemutaran kernel bertujuan untuk menjamin sifat komutatif perkalian linear antara kernel dan area parsial masukan. Sifat komutatif pada operasi konvolusi tidak dibutuhkan pada JSK. Operasi konvolusi pada JSK disebut sebagai *cross-correlation*. Namun, penamaan ini jarang digunakan sehingga operasi tersebut tetap dinamakan konvolusi. Pemelajaran JSK bertujuan untuk mengoptimisasi bobot setiap kernel yang digunakan. Proses optimisasi tetap memanfaatkan *back-propagation* dan SGD terhadap setiap komponen pembentuk kernel. Sama seperti PLB, luaran dari JSK memerlukan operasi non-linear seperti ReLU sehingga mampu menghasilkan representasi data yang lebih kompleks. Selain itu, JSK memiliki suatu operasi non-linear lain yang tidak ada pada model PLB yang disebut sebagai *pooling*.



Gambar 2.4: Operasi konvolusi 2 dimensi pada JSK (Goodfellow et al., 2016). Operasi konvolusi dilakukan dengan menggeser kernel di sepanjang baris dan kolom masukan. Pada ilustrasi ini luaran dibatasi hanya untuk area yang ditempati oleh kernel secara utuh.

Operasi *pooling* juga memanfaatkan suatu kernel yang bertugas memindai masukan. Hanya saja, kernel tersebut tidak memiliki bobot seperti JSK. Kernel pada operasi *pooling* bertugas untuk mendapatkan deskripsi statistik dari regional masukan yang ditempati oleh kernel. Terdapat dua operasi *pooling* yang umum digunakan yaitu *max-pooling* dan *average-pooling*. Operasi *max-pooling* memilih komponen masukan terbesar dari regional komponen yang ditempati oleh kernel. Sementara itu, operasi *average-pooling* menghitung rata-rata dari setiap komponen masukan yang ditempati oleh kernel. Lapisan *pooling* menjamin representasi yang dihasilkan bersifat *invariant* terhadap sejumlah kecil operasi translasi (Goodfellow et al., 2016). Gambar 2.5 menunjukkan ilustrasi *max-pooling*.



Gambar 2.5: Ilustrasi *max-pooling* pada JSK. kernel berukuran 2×2 memindai masukan berukuran 4×4. Kernel memilih komponen terbesar untuk setiap regional masukan yang ditempati oleh kernel.

Terdapat 3 karakteristik dari JSK yang dapat meningkatkan performa dari pemelajaran mesin: interaksi berjarak, penggunaan parameter bersama, dan representasi yang bersifat ekuivarian (Goodfellow et al., 2016). Pada model JST, perkalian setiap komponen matriks bobot melibatkan semua komponen data masukan. Hal ini berarti setiap komponen luaran memiliki interaksi dengan semua komponen masukan. Metode ini berbeda dengan JSK yang memanfaatkan kernel yang ukurannya lebih kecil dibandingkan dengan ukuran masukannya. Ketika melakukan operasi konvolusi pada data citra, kernel hanya menempati sebagian kecil area citra tersebut. Artinya, perkalian matriks bobot hanya melibatkan area yang sedang ditempati oleh kernel. Hal ini berimplikasi pada penghematan memori komputasi serta peningkatan efisiensi secara statistik (Goodfellow et al., 2016). Karakter yang berikutnya adalah penggunaan parameter bersama untuk lebih dari satu fungsi pada model jaringan saraf tiruan Goodfellow et al. (2016). Setiap kernel akan digeser di sepanjang area citra menggunakan bobot kernel yang sama. Hal ini berbeda dengan jaringan saraf tiruan biasa yang mana setiap komponen matriks bobot hanya ditugaskan untuk satu kali perhitungan untuk semua komponen masukan. Adanya sifat penggunaan parameter bersama, memunculkan karakter terakhir yaitu sifat ekuivarian terhadap translasi Goodfellow et al. (2016). Suatu fungsi dikatakan bersifat ekuivarian jika masukan mengalami sejumlah perubahan maka fungsi akan merespon dengan memberi luaran yang juga berubah mengikuti perubahan masukan. Operasi konvolusi secara alamiah tidak bersifat ekuivarian terhadap transformasi lainnya sehingga diperlukan upaya lebih lanjut untuk menangani transformasi yang lebih kompleks.

2.3 Representasi Tidak Terawasi

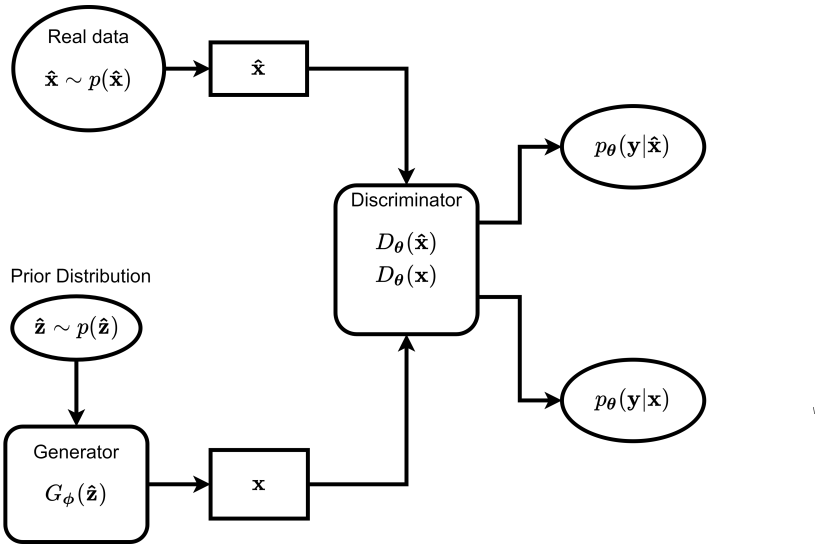
Pemelajaran mesin seringkali berurusan dengan data berdimensi tinggi. Dimensionalitas yang tinggi seringkali mempersulit model pemelajaran mesin untuk mempelajari distribusi data tersebut. Dimensionalitas yang semakin tinggi menyebabkan data tersebar terlalu *sparse* satu dengan yang lainnya. Kondisi ini dinamakan *curse of dimensionality* (Goodfellow et al., 2016). Pemelajaran representasi (*representation learning*) bertujuan untuk menemukan struktur data yang lebih sederhana dan mampu mempelajari distribusi data tersebut. Representasi tersebut dapat dimanfaatkan untuk menyelesaikan berbagai permasalahan PM, terutama untuk data yang sudah teranotasi. Kehadiran model representasi pada pemelajaran mesin dapat mempercepat proses inferensi dan menghasilkan suatu model yang memiliki interpretasi yang baik (Zaheer, 2018). Bagian ini secara khusus menjelaskan penelitian terkait model representasi citra tidak terawasi berbasis jaringan saraf tiruan. Pada bagian ini dijelaskan dua model representasi citra yaitu representasi klasik (GAN, *autoencoder*, dan pemelajaran terawasi sendiri) dan model representasi transformasi-ekuivarian (*transformation-equivariant*). Model representasi pertama berbasis *data-likelihood*, artinya model representasi tersebut hanya mempertimbangkan kemunculan sampel data sebagai kriteria pembelajarannya. Representasi transformasi-ekuivarian merespon luaran representasi yang dihasilkan terhadap transformasi yang diterapkan pada citra masukan (Qi et al., 2020). Sebelum kehadiran model representasi berbasis jaringan saraf tiruan, sudah ada beberapa model representasi tidak terawasi yang bersifat linear maupun non-linear seperti *principled componenet analysis* (PCA), *independent component analysis* (ICA), dan K-means (Bishop, 2006). Penelitian ini secara khusus membahas model representasi untuk data berupa citra.

2.3.1 GAN, *Autoencoder*, dan Pemelajaran Terawasi Sendiri

Pada bagian ini dijelaskan 3 model pemelajaran yaitu *generative adversarial netowork* (GAN), *autoencoder*, dan pemelajaran terawasi sendiri.

Generative adversarial network (GAN) merupakan model generatif berbasis jaringan saraf tiruan yang terdiri dari generator G dan diskriminator D (Goodfellow et al., 2020). Generator G mempelajari distribusi citra p_d dengan cara membangkitkan citra $\mathbf{x}$ yang mengikuti suatu distribusi p_g sedemikian sehingga p_g secara statistik semirip mungkin dengan p_d. Generator G membangkitkan citra dengan cara mentransformasi suatu *noise* $\hat{\mathbf{z}}$ yang dipilih secara acak dari suatu distribusi *prior* $p_{\hat{\mathbf{z}}}$. Transformasi tersebut melibatkan fungsi $G_{\phi}(\hat{\mathbf{z}})$ yang bersifat *differentiable* dan direpresentasikan oleh jaringan saraf tiruan yang terparameterisasi oleh ϕ. Diskriminator D mentransformasi citra $\mathbf{x}$ yang berasal dari p_g maupun $\hat{\mathbf{x}}$ yang berasal dari p_d ke dalam suatu nilai skalar. Nilai ini menyatakan probabilitas $\mathbf{x}$ maupun $\hat{\mathbf{x}}$ berasal dari p_d. Transformasi diskriminator melibatkan fungsi $D_{\boldsymbol{\theta}}(\mathbf{x})$ dan $D_{\boldsymbol{\theta}}(\hat{\mathbf{x}})$ yang sifatnya *differentiable* dan direpresentasikan oleh jaringan saraf

tiruan yang terparameterisasi oleh $\boldsymbol{\theta}$. Gambar 2.6 menunjukkan Ilustrasi model GAN.



Gambar 2.6: Model GAN yang terdiri dari generator dan diskriminator, diajukan oleh Goodfellow et al. (2020)

Generator dan diskriminator bekerja secara adversarial (berlawanan). Diskriminator D dioptimisasi dengan cara memaksimalkan probabilitas $\hat{\mathbf{x}}$ berasal dari p_d dan meminimalkan $\mathbf{x}$ berasal dari p_d. Optimisasi generator G secara tidak langsung memaksimalkan probabilitas $\mathbf{x}$ berasal dari p_d dengan cara membangkitkan citra dari distribusi p_g yang mendekati p_d. Secara matematis, generator G berusaha meminimalkan $1 - D(G(\hat{\mathbf{z}}))$. Diskriminator dan generator dapat dipandang sebagai model *min-max* 2 pemain dengan fungsi $V(D,G)$ (Goodfellow et al., 2016). Kemudian kita dapat menuliskan fungsi objektif GAN $V(D,G)$ sebagai:

$$\min_G \max_D V(D,G) = \mathbb{E}_{p_d(\hat{\mathbf{x}})}[\log D(\hat{\mathbf{x}})] + \mathbb{E}_{p_{\mathbf{z}}(\hat{\mathbf{z}})}[\log(1 - D(G(\hat{\mathbf{z}})))] \tag{2.12}$$

operasi logaritma pada fungsi tersebut bertujuan untuk menjaga stabilitas numerik luaran fungsi tersebut. Pada model ini, vektor *noise* berperan sebagai representasi dari distribusi data pada dimensi tinggi. Pemelajaran adversarial memaksa distribusi representasi mengikuti distribusi prior yang merupakan distribusi asal dari vektor *noise*.

Meskipun GAN mampu membangkitkan citra yang realistis, GAN memiliki dua masalah utama yaitu *mode collapse* (Arjovsky et al., 2017) dan *vanishing gradient* (Arjovsky and Bottou, 2017). *Mode collapse* merupakan fenomena dimana GAN membangkitkan citra yang bersifat homogen. Hal ini terjadi karena pemelajaran yang dilakukan GAN bersifat trivial sehingga tidak mampu menangkap distribusi citra secara utuh. Masalah ini kemudian berusaha diselesaikan dengan cara mengganti fungsi objektif GAN menggunakan Wasserstein *distance*. Model ini dinamakan sebagai Wasserstein GAN. Permasalahan berikutnya adalah *vanishing gradient*, yaitu kondisi dimana gradien dari fungsi objektif bernilai nol (Arjovsky and Bottou, 2017). Kondisi

tersebut menyebabkan tidak ada aliran informasi yang dapat dipergunakan oleh generator untuk proses pemelajaran. Secara intuitif proses pemelajaran diskriminator lebih mudah dibandingkan dengan generator dikarenakan ukuran dimensi luaran generator jauh lebih besar dibandingkan dengan dimensi luaran diskriminator yang berupa skalar. Kedua masalah tersebut muncul dikarenakan pemelajaran GAN sulit mencapai titik konvergensi dan relatif tidak stabil (Arjovsky and Bottou, 2017).

Autoencoder Model representasi berikutnya adalah *autoencoder*. *Autoencoder* terdiri dari 2 bagian: *encoder* dan *decoder* (Goodfellow et al., 2016). Jaringan *encoder* mentransformasi citra $\hat{\mathbf{x}}$ menjadi sebuah representasi $\hat{\mathbf{z}}$. Jaringan *decoder* menerima representasi $\hat{\mathbf{z}}$ dan mentransformasi representasi tersebut. Luaran *decoder* berupa rekonstruksi dari citra $\hat{\mathbf{x}}$. Fungsi objektif pada *autoencoder* pada umumnya meminimalkan kesalahan rekonstruksi antara luaran *decoder* dan data asli. Kesalahan rekonstruksi dapat direpresentasikan menggunakan fungsi objektif *mean squared error* maupun *binary cross entropy*.

Permasalahan utama pada *autoencoder* adalah pemelajaran yang bersifat trivial (Steck, 2020). *Autoencoder* cenderung mempelajari suatu fungsi identitas untuk memaksimalkan rekonstruksi citra yang merupakan objektif pemelajaran. Fungsi identitas memberi indikasi bahwa *autoencoder* hanya menghafal representasi citra yang pernah ditemui selama proses pemelajaran. Untuk mengatasi masalah tersebut, variasi pengembangan *autoencoder* telah banyak diajukan. Salah satunya adalah *variational autoencoder* (VAE) yang dikembangkan sebagai suatu model probabilistik. Model ini memaksimalkan *evidence lower bound* (ELBO) dari distribusi data $p(\hat{\mathbf{x}})$ (Kingma and Welling, 2014). *Autoencoder* juga telah dikembangkan dengan menggunakan pendekatan pemelajaran adversarial. Model tersebut dinamakan *adversarial autoencoder* (Makhzani et al., 2015). Model ini menggunakan diskriminator sebagai jaringan tambahan yang bertujuan untuk meminimalkan probabilitas representasi $\hat{\mathbf{z}}$ berasal dari distribusi *prior*. Kehadiran diskriminator berupaya untuk meregularisasi representasi sedemikian sehingga KL-*divergence* antara distribusi posterior representasi dan distribusi *prior* seminimal mungkin. *Denoising autoencoder* didesain untuk merekonstruksi citra masukan yang sebelumnya diberi *noise* menjadi citra asli yang jernih (Vincent et al., 2008). Tujuan dari pemelajaran tersebut untuk menghasilkan suatu model representasi yang mampu menghiraukan variansi dari komponen yang tidak signifikan. Dengan motivasi yang sama dengan *denoising autoencoder*, *contrastive autoencoder* memperkenalkan suatu *penalty-term* pada fungsi objektifnya (Rifai et al., 2011). Terlepas dari banyaknya pendekatan yang diajukan, permasalahan pemelajaran pada *autoencoder* belumlah terselesaikan sepenuhnya.

Pemelajaran terawasi sendiri (*Self-supervised learning*) Model pemelajaran dalam bekerja dengan sangat baik apabila terdapat label yang dapat dimanfaatkan sebagai sinyal pemelajaran. Sayangnya, pemelajaran representasi sering kali berurusan dengan data yang tidak dianotasi. Pemelajaran terawasi sendiri mensintesis informasi dari citra masukan sehingga dapat digunakan sebagai sinyal pemelajaran (label pengganti). Terdapat

berbagai macam pendekatan yang telah diajukan sebagai model pemelajaran terawasi sendiri. Noroozi and Favaro (2016) membagi citra menjadi beberapa potongan kemudian mengacak posisi potongan tersebut. Kemudian, suatu JST khusus yang dinamakan jaringan *context-free* dilatih untuk memprediksi indeks potongan yang telah diacak. Doersch et al. (2015) juga melatih model representasi dengan terlebih dahulu mengubah citra menjadi beberapa potongan. Kemudian, JSK dilatih untuk memperediksi posisi relatif dari sebuah potongan diberikan potongan yang lain. Potongan yang kedua berperan sebagai konteks informasi bagi JSK. Dosovitskiy et al. (2014) menggunakan kelas pengganti dengan terlebih dahulu menerapkan berbagai macam transformasi pada citra. Gidaris et al. (2018) menerapkan transformasi rotasi secara acak pada setiap citra. Selanjutnya, dibangun suatu prediktor yang bertugas untuk memprediksi rotasi yang diterapkan pada masing-masing citra. Noroozi et al. (2017) mengajukan suatu model representasi yang dilatih untuk menghitung jumlah fitur dari suatu citra yang sebelumnya telah ditransformasi oleh beberapa operasi transformasi. Pada penelitian tersebut, transformasi dibatasi hanya untuk *scaling* dan *tiling*. Representasi yang dihasilkan oleh pemelajaran terawasi sendiri terbatas pada pemilihan kelas pengganti yang dipilih untuk diprediksi.

2.3.2 Representasi Tidak Terawasi Transformasi-Ekuivarian

Model representasi transformasi-ekuivarian (*transformation-equivariant*) diawali oleh jaringan kapsul (Hinton et al., 2011, 2018; Lenssen et al., 2018; Wang and Liu, 2018). Jaringan kapsul terdiri dari beberapa JST terpisah yang masing-masing bertugas untuk menemukan representasi bagian/regional tertentu dari suatu citra. Representasi yang dihasilkan oleh masing-masing kapsul bersifat ekuivarian terhadap berbagai macam transformasi. Kendala utama dari jaringan kapsul adalah belum adanya metode yang secara tepat mampu mengendalikan sifat transformasi-ekuivarian dari representasi yang dihasilkan (Qi et al., 2020).

Beberapa penelitian berusaha mengembangkan jaringan saraf konvolusional khusus yang bersifat transformasi-ekuivarian terhadap berbagai transformasi. Konvolusi grup-ekuivarian memperkenalkan operasi konvolusi khusus yang melibatkan grup yang dinamakan $p4$ dan $p4m$ untuk menjamin sifat transformasi-ekuivarian terhadap transformasi rotasi, translasi, dan refleksi. Akibatnya, jaringan ini mampu menangkap struktur objek yang lebih kompleks dari suatu citra. (Cohen and Welling, 2017) mendefinisikan representasi transformasi-ekuivarian sebagai *steerability*. Konsep *steerability* diimplementasikan dengan membangun JSK yang dilengkapi oleh *filter bank*. *Filter bank* secara sederhana dapat dipandang sebagai suatu transformasi linear terhadap representasi citra. Jaringan konvolusional yang dilengkapi dengan *filter bank* dinamakan *steerable*-CNN.

Autoencoding transformation (AET) mengadopsi arsitektur *autoencoder* untuk membangun model representasi transformasi-ekuivarian. Misalkan terdapat citra $\hat{\mathbf{x}} \sim p(\hat{\mathbf{x}})$

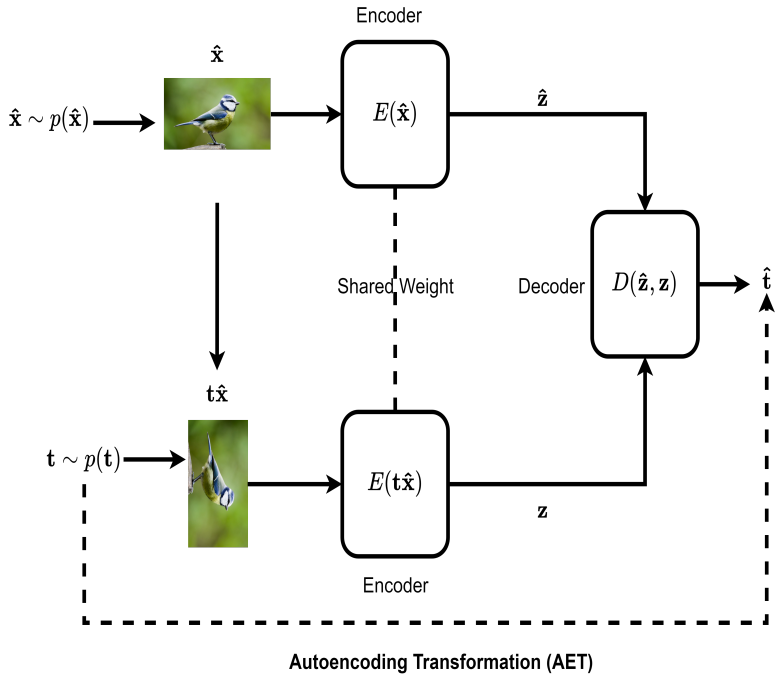
dan suatu transformasi $\mathbf{t} \sim p(\mathbf{t})$. Selanjutnya, transformasi $\mathbf{t}$ diterapkan pada citra $\hat{\mathbf{x}}$ sehingga menghasilkan $\mathbf{t}\hat{\mathbf{x}}$. Pada penelitian ini, transformasi $\mathbf{t}$ dipandang sebagai matriks sehingga memungkinkan terjadinya perkalian matriks antara $\mathbf{t}$ dan vektor $\hat{\mathbf{x}}$. *Encoder* E menghasilkan luaran berupa representasi $\hat{\mathbf{z}} = E(\hat{\mathbf{x}})$ dan $\mathbf{z} = E(\mathbf{t}\hat{\mathbf{x}})$. Berbeda dengan *data-likelihood autoencoder* yang merekonstruksi data $\hat{\mathbf{x}}$, luaran AET berupa $\hat{\mathbf{t}}$ yang merupakan rekonstruksi dari transformasi $\mathbf{t}$. Representasi $\hat{\mathbf{z}}$ dan $\mathbf{z}$ menjadi masukan bagi *decoder* D. Secara matematis, pemelajaran AET dapat dituliskan sebagai:

$$\min_{E,D} \mathbb{E}_{p(\hat{\mathbf{x}})p(\mathbf{t})}\, \ell(\mathbf{t}, \hat{\mathbf{t}}) \tag{2.13}$$

dengan

$$\hat{\mathbf{t}} = D(\hat{\mathbf{z}}, \mathbf{z})$$

Euclidean *distance* dipilih sebagai fungsi ℓ. Gambar 2.7 menunjukkan ilustrasi model representasi AET.



Gambar 2.7: Ilustrasi model representasi AET. *Encoder* menerima citra $\hat{\mathbf{x}}$ dan $\mathbf{t}\hat{\mathbf{x}}$ sebagai masukan. *Encoder* E menghasilkan representasi $\hat{\mathbf{z}}$ dan $\mathbf{z}$. *Decoder* D menghasilkan $\hat{\mathbf{t}}$ yang merupakan rekonstruksi dari transformasi $\mathbf{t}$.

Terdapat tiga cara yang dapat digunakan untuk memilih transformasi yang akan diterapkan pada data. Cara yang pertama menggunakan transformasi yang terparameterisasi. Secara matematis, ruang sampel transformasi yang terparameterisasi oleh $\boldsymbol{\theta}$ dapat ditulis sebagai $\mathcal{T} = \{\mathbf{t}_{\boldsymbol{\theta}} | \boldsymbol{\theta} \sim p(\theta)\}$, dengan parameter $\boldsymbol{\theta}$ disampel dari $p(\theta)$ (Zhang et al., 2019). Beberapa transformasi yang dapat direpresentasikan diantaranya

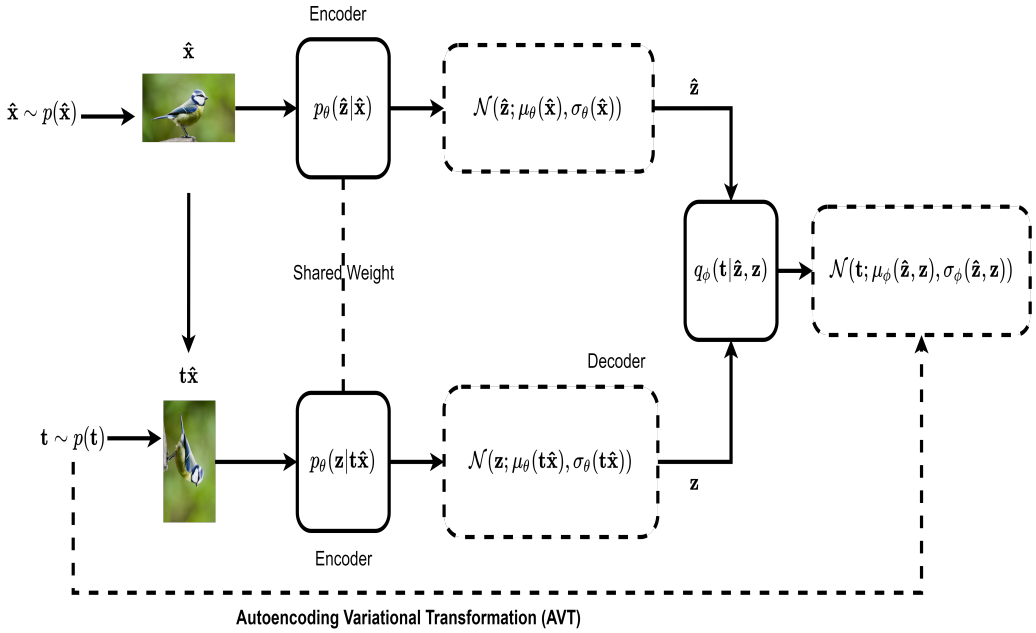
transformasi *affine* dan transformasi proyektif. Cara yang kedua memanfaatkan model *local* GAN (Qi et al., 2018). Terdapat generator $G(\hat{\mathbf{x}}, \hat{\mathbf{z}})$ yang menerima dua masukan berupa citra $\hat{\mathbf{x}}$ dan vektor *noise* $\hat{\mathbf{z}}$. *Noise* $\hat{\mathbf{z}}$ berperan sebagai parameter transformasi citra $\hat{\mathbf{x}}$. Jenis transformasi ini lebih variatif dibandingkan dengan transformasi pada umumnya yang hanya sedikit mengubah struktur geometris dari data. Hal tersebut dikarenakan GAN mampu menangkap struktur citra pada level semantik sehingga rentang perubahan citra jauh lebih lebar (Radford et al., 2016). Cara yang terakhir memanfaatkan transformasi non-parametrik $\mathbf{t} \sim \mathcal{T}$, artinya tidak ada parameter yang mengikat $\mathcal{T}$. Notasi $\mathcal{T}$ menyatakan himpunan transformasi $\mathbf{t}$ yang dapat diterapkan pada $\hat{\mathbf{x}}$.

Autoencoding variational transformation (AVT) mengembangkan AET dari perspektif teori informasi (Qi et al., 2019). Model AVT memaksimalkan ITB $I_{\boldsymbol{\theta}}(\mathbf{T}; \mathbf{Z}|\hat{\mathbf{Z}})$. Informasi timbal balik $I_{\boldsymbol{\theta}}(\mathbf{T}; \mathbf{Z}|\hat{\mathbf{Z}})$ merupakan suku kedua dari Persamaan 1.1. Perhitungan ITB secara langsung tidak memungkinkan karena distribusi $p_{\boldsymbol{\theta}}(\mathbf{t}|\mathbf{z}, \hat{\mathbf{z}})$ tidak diketahui. Karena itu, AVT mengestimasi ITB dengan menggunakan estimasi batas bawah BA. Estimasi batas bawah BA memperkenalkan suatu distribusi variasional $q_{\boldsymbol{\phi}}$ yang terparameterisasi oleh $\boldsymbol{\phi}$ untuk mengaproksimasi $p_{\boldsymbol{\theta}}(\mathbf{t}|\mathbf{z}, \hat{\mathbf{z}})$. Estimasi batas bawah BA memaksimalkan distribusi $q_{\boldsymbol{\phi}}(\mathbf{t}|\mathbf{z}, \hat{\mathbf{z}})$ terhadap parameter $\boldsymbol{\theta}$ dan $\boldsymbol{\phi}$. Parameter $\boldsymbol{\theta}$ mengikat distribusi $p_{\boldsymbol{\theta}}(\hat{\mathbf{z}}|\hat{\mathbf{x}})$ dan $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$. Representasi $\hat{\mathbf{z}}$ dan $\mathbf{z}$ nantinya akan dibangkitkan dari kedua distribusi tersebut. Secara matematis, fungsi objektif AVT dapat dituliskan sebagai:

$$\max_{\boldsymbol{\theta}, \boldsymbol{\phi}} \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{t}, \mathbf{z}, \hat{\mathbf{z}})} \log q_{\boldsymbol{\phi}}(\mathbf{t}|\mathbf{z}, \hat{\mathbf{z}}) \tag{2.14}$$

Pada penelitian tersebut, diasumsikan bahwa $p_{\boldsymbol{\theta}}(\hat{\mathbf{z}}|\hat{\mathbf{x}}) = \mathcal{N}(\hat{\mathbf{z}}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}), \sigma_{\boldsymbol{\theta}}(\hat{\mathbf{x}})$, $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{t}\hat{\mathbf{x}}), \sigma_{\boldsymbol{\theta}}(\mathbf{t}\hat{\mathbf{x}}))$, dan $q_{\boldsymbol{\phi}}(\mathbf{t}|\hat{\mathbf{z}}, \mathbf{z}) = \mathcal{N}(\mathbf{t}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\hat{\mathbf{z}}, \mathbf{z}), \sigma_{\boldsymbol{\phi}}(\hat{\mathbf{z}}, \mathbf{z}))$ (berasal dari keluarga distribusi Gaussian multivariat terfaktorisasi). Notasi $\boldsymbol{\mu}$ menyatakan vektor *mean* dan σ menyatakan vektor standar deviasi.

Sama seperti AET, AVT terdiri dari *encoder* dan *decoder*. *Encoder* $E_{\boldsymbol{\theta}}$ terparameterisasi oleh $\boldsymbol{\theta}$, sementara *decoder* $D_{\boldsymbol{\phi}}$ terparameterisasi oleh $\boldsymbol{\phi}$. *Encoder* $E_{\boldsymbol{\theta}}$ menerima masukan citra $\hat{\mathbf{x}}$ dan citra transformasi $\mathbf{t}\hat{\mathbf{x}}$. Berbeda dengan AET, luaran dari *encoder* berupa parameter distribusi representasi $p_{\boldsymbol{\theta}}(\hat{\mathbf{z}}|\hat{\mathbf{x}})$ dan $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ ($\boldsymbol{\mu}_{\boldsymbol{\theta}}$ dan $\boldsymbol{\sigma}_{\boldsymbol{\theta}}$). Sementara itu, luaran dari *decoder* $D_{\boldsymbol{\phi}}$ berupa parameter distribusi transformasi $q_{\boldsymbol{\phi}}(\mathbf{t}|\hat{\mathbf{z}}, \mathbf{z})$ ($\boldsymbol{\mu}_{\boldsymbol{\phi}}$ dan $\boldsymbol{\sigma}_{\boldsymbol{\phi}}$). Gambar 2.8 menunjukkan ilustrasi model representasi AVT.



Gambar 2.8: Ilustrasi model representasi AVT. *Encoder* menerima masukan $\hat{\mathbf{x}}$ dan $\mathbf{t}\hat{\mathbf{x}}$ dan menghasilkan luaran berupa parameter distribusi $p_{\boldsymbol{\theta}}(\hat{\mathbf{z}}|\hat{\mathbf{x}})$ dan $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{t}, \hat{\mathbf{x}})$. Luaran *decoder D* berupa parameter distribusi $q_{\boldsymbol{\phi}}(\mathbf{t}|\mathbf{z}, \hat{\mathbf{z}})$.

Model representasi AVT menerapkan *reparameterization trick* untuk membangkitkan sampel $\hat{\mathbf{z}}$ dan $\mathbf{z}$ dari $p_{\boldsymbol{\theta}}(\hat{\mathbf{z}}|\hat{\mathbf{x}})$ dan $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ (Qi et al., 2019). Teknik *sampling* tersebut memungkinkan *back-propagation* dapat dilakukan. Bentuk estimasi non-bias dari fungsi objektif AVT untuk K sampel $(\mathbf{t}^{(i)}, \mathbf{z}^{(i)}, \hat{\mathbf{z}}^{(i)})$ dapat dituliskan sebagai:

$$\max_{\boldsymbol{\theta}, \boldsymbol{\phi}} \frac{1}{K} \sum_{i=1}^{K} \log \mathcal{N}(\mathbf{t}^{(i)}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\hat{\mathbf{z}}^{(i)}, \mathbf{z}^{(i)}), \boldsymbol{\sigma}_{\boldsymbol{\phi}}(\hat{\mathbf{z}}^{(i)}, \mathbf{z}^{(i)})) \tag{2.15}$$

dengan

$$\hat{\mathbf{z}}^{(i)} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(i)}) + \boldsymbol{\sigma}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(i)}) \odot \hat{\boldsymbol{\epsilon}}^{(i)} \qquad \textit{reparameterization-trick}$$

dan

$$\mathbf{z}^{(i)} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) + \boldsymbol{\sigma}_{\boldsymbol{\theta}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) \odot \boldsymbol{\epsilon}^{(i)} \qquad \textit{reparameterization-trick}$$

Notasi $\boldsymbol{\epsilon}$ dan $\hat{\boldsymbol{\epsilon}}$ menyatakan *noise* yang digunakan untuk melakukan *reparameterization trick* dengan $\boldsymbol{\epsilon}^{(i)}, \hat{\boldsymbol{\epsilon}}^{(i)} \sim \mathcal{N}(\boldsymbol{\epsilon}; 0, \mathbb{I})$ untuk $i = 1, \cdots, K$. Berbeda dengan AVT, penelitian ini membangun model representasi transformasi-ekuivarian dengan memaksimalkan ITB $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}})$ sebagai kriteria pemelajaran.

BAB 3
TRANSFORMASI-EKUIVARIAN VARIASIONAL

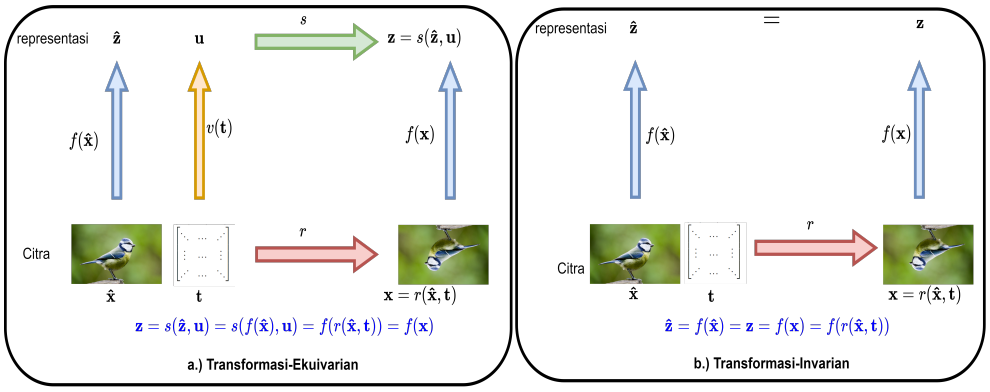
3.1 Definisi Representasi Transformasi-Ekuivarian

Autoencoding variational transformation dibangun berdasarkan definisi transformasi-ekuivarian yang diperkenalkan oleh Cohen and Welling (2017).

Definisi 3.1.1 (Transformasi-ekuivarian) *Misalkan* $\hat{\mathbf{x}} \sim \mathcal{X}$ *menyatakan citra yang awal yang belum ditransformasi. Terdapat fungsi* $f : \mathcal{X} \to \mathcal{Z}$ *yang memetakan citra* $\mathbf{x}$ *menuju suatu representasi* $\hat{\mathbf{z}} \in \mathcal{Z}$ *dengan* $\hat{\mathbf{z}} = f(\mathbf{x})$. *Fungsi* f *merepresentasikan model pemelajaran yang berperan mengekstraksi representasi citra. Selanjutnya, misalkan* $\mathbf{t} \in \mathcal{T}$ *menyatakan operasi transformasi pada citra, seperti rotasi, translasi, dll. Terdapat fungsi* $r : \mathcal{X} \times \mathcal{T} \to \mathcal{X}$ *yang memetakan citra* $\mathbf{x}$ *dan transformasi* $\mathbf{t}$ *sehingga menghasilkan citra hasil transformasi* $\mathbf{x} \in \mathcal{X}$ *dengan* $\mathbf{x} = r(\hat{\mathbf{x}}, \mathbf{t}) = \mathbf{t}\hat{\mathbf{x}}$. *Kemudian,* $\mathbf{z} \in \mathcal{Z}$ *dengan* $\mathbf{z} = f(\mathbf{x})$ *menyatakan representasi citra* $\mathbf{x}$. *Representasi* $\hat{\mathbf{z}}$ *dan* $\mathbf{z}$ *dikatakan bersifat transformasi-ekuivarian jika:*

- *Terdapat fungsi* $v : \mathcal{T} \to \mathcal{U}$ *yang memetakan transformasi citra* $\mathbf{t}$ *menuju suatu transformasi pada level representasi* $\mathbf{u} \in \mathcal{U}$ *dengan* $\mathbf{u} = v(\mathbf{t})$.
- *Terdapat fungsi* $s : \mathcal{Z} \times \mathcal{U} \to \mathcal{Z}$ *yang memetakan representasi* $\hat{\mathbf{z}}$ *dan transformasi representasi* $\mathbf{u}$ *menuju representasi* $\mathbf{z} = s(\hat{\mathbf{z}}, \mathbf{u}) = \mathbf{u}\hat{\mathbf{z}}$ *sedemikian sehingga* $\mathbf{z} = s(\hat{\mathbf{z}}, \mathbf{u}) = f(r(\mathbf{x}, \mathbf{t}))$.

Gambar 3.1 a.) menunjukkan ilustrasi transformasi-ekuivarian sesuai dengan **Definisi 3.1.1**. Berdasarkan fungsi s, representasi $\mathbf{z}$ dapat ditentukan berdasarkan representasi $\hat{\mathbf{z}}$ dan transformasi $\mathbf{t}$ tanpa memerlukan akses terhadap $\mathbf{x}$. Cohen and Welling (2017) secara spesifik menamakan **Definisi 3.1.1** sebagai *steerability*. Berbeda dengan transformasi-ekuivarian, representasi transformasi-invarian representasi memenuhi $\mathbf{z} = f(\mathbf{x}) = f(r(\hat{\mathbf{x}}, \mathbf{t})) = f(\mathbf{t}\hat{\mathbf{x}}) = f(\hat{\mathbf{x}}) = \hat{\mathbf{z}}$. Fungsi f tidak mempertimbangkan transformasi $\mathbf{t}$ yang diterapkan pada $\hat{\mathbf{x}}$ sehingga representasi $\hat{\mathbf{z}}$ bernilai sama dengan $\mathbf{z}$. Ilustrasi representasi transformasi-invarian ditunjukkan oleh Gambar 3.1 b.). Permasalahan utama dari definisi tersebut terletak pada fungsi v yang memetakan transformasi $\mathbf{t}$ yang bersifat linear. Kondisi tersebut mengurangi fleksibilitas $\mathbf{u}$ sehingga mempengaruhi representasi yang dihasilkan.



Gambar 3.1: a.) Ilustrasi transformasi-ekuivarian yang menjadi fokus pada penelitian ini (Zaheer, 2018). **b.)** Ilustrasi representasi transformasi-invarian.

Qi et al. (2020) menggeneralisasi konsep *steerability* dengan memodelkan **Definisi 3.1.1** sebagai suatu ITB. Misalkan VA $\mathbf{Z}, \hat{\mathbf{Z}}$, dan $\mathbf{T}$ masing-masing mewakili pelbagai kemungkinan representasi $\mathbf{z}, \hat{\mathbf{z}}$, dan transformasi $\mathbf{t}$. *Steerability* dapat dimodelkan melalui ITB $I(\mathbf{Z}, \hat{\mathbf{Z}})$ (Qi et al., 2020). Secara matematis, dapat dituliskan:

$$I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T}) = I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T}, \mathbf{X}) - I_{\boldsymbol{\theta}}(\mathbf{Z}; \mathbf{X} | \hat{\mathbf{Z}}, \mathbf{T}) \leq I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T}, \mathbf{X}) \tag{3.1}$$

Notasi $\boldsymbol{\theta}$ menyatakan parameter model pemelajaran (misalnya *autoencoder*). Dekomposisi ITB $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T})$ didapat melalui aturan rantai. Memaksimalkan $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T})$ menyebabkan batas bawah terhadap $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T}, \mathbf{X})$ semakin ketat. Upaya memaksimalkan $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T})$ dilakukan dengan memilih parameter $\boldsymbol{\theta}$ yang memaksimalkan ITB tersebut:

$$\boldsymbol{\theta}^* = \max_{\boldsymbol{\theta}} I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T}) \tag{3.2}$$

Dengan menggunakan aturan rantai distribusi *joint*, $I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T})$ dapat diuraikan menjadi:

$$\begin{aligned} I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}, \mathbf{T}) &= \mathop{\mathbb{E}}_{p_{\boldsymbol{\theta}}(\mathbf{z}, \hat{\mathbf{z}}, \mathbf{t})} \left[\frac{p_{\boldsymbol{\theta}}(\mathbf{z}, \hat{\mathbf{z}}, \mathbf{t})}{p_{\boldsymbol{\theta}}(\mathbf{z})\, p_{\boldsymbol{\theta}}(\hat{\mathbf{z}}, \mathbf{t})} \right] = \mathop{\mathbb{E}}_{p_{\boldsymbol{\theta}}(\mathbf{z}, \hat{\mathbf{z}}, \mathbf{t})} \left[\frac{p_{\boldsymbol{\theta}}(\hat{\mathbf{z}}) p_{\boldsymbol{\theta}}(\mathbf{z}|\hat{\mathbf{z}}) p_{\boldsymbol{\theta}}(\mathbf{t}|\mathbf{z}, \hat{\mathbf{z}})}{p_{\boldsymbol{\theta}}(\mathbf{z})\, p_{\boldsymbol{\theta}}(\mathbf{t}|\hat{\mathbf{z}}) p_{\boldsymbol{\theta}}(\hat{\mathbf{z}})} \right] \\ &= \mathop{\mathbb{E}}_{p_{\boldsymbol{\theta}}(\mathbf{z}, \hat{\mathbf{z}}, \mathbf{t})} \left[\frac{p_{\boldsymbol{\theta}}(\mathbf{z}|\hat{\mathbf{z}})}{p_{\boldsymbol{\theta}}(\mathbf{z})} \right] + \mathop{\mathbb{E}}_{p_{\boldsymbol{\theta}}(\mathbf{z}, \hat{\mathbf{z}}, \mathbf{t})} \left[\frac{p_{\boldsymbol{\theta}}(\mathbf{t}|\mathbf{z}, \hat{\mathbf{z}})}{p_{\boldsymbol{\theta}}(\mathbf{t}|\hat{\mathbf{z}})} \right] \\ &= I_{\boldsymbol{\theta}}(\mathbf{Z}; \hat{\mathbf{Z}}) + I_{\boldsymbol{\theta}}(\mathbf{Z}; \mathbf{T} | \hat{\mathbf{Z}}) \end{aligned} \tag{3.3}$$

sehingga didapatkan Kesamaan 1.1. Memodelkan *steerability* dengan memaksimalkan ITB secara tidak langsung dapat memetakan relasi VA $\hat{\mathbf{Z}}$, $\mathbf{Z}$, dan $\mathbf{T}$ dengan lebih fleksibel. Karena itu, mendefinisikan representasi transformasi-ekuivarian sebagai suatu ITB secara teoritis lebih baik jika dibandingkan dengan *filter bank* yang diajukan Cohen and Welling (2017).

Autoencoding variational transformation memaksimalkan $I_{\boldsymbol{\theta}}(\mathbf{Z};\, \mathbf{T}|\hat{\mathbf{Z}})$ dengan menggunakan estimasi batas bawah Barber-Agakov. **Pada penelitian ini, representasi transformasi-ekuivarian direalisasikan dengan memaksimalkan** $I_{\boldsymbol{\theta}}(\mathbf{Z};\, \hat{\mathbf{Z}})$. Model tersebut dinamakan transformasi-ekuivarian variasional (TEV). Model representasi tersebut memaksimalkan ITB melalui estimasi batas bawah BA dan estimasi batas bawah InfoNCE. Kedua metode estimasi dipilih karena keduanya hanya membutuhkan sampel dari distribusi probabilitas *joint* VA yang terlibat. Pada kasus TEV dan AVT, sampel berasal dari distribusi probabilitas *joint* $p(\mathbf{z}, \hat{\mathbf{z}}, \mathbf{t})$. Sebagai pembanding, metode estimasi batas bawah Nguyen-Wright-Jordan dan Donsker-Varadhan membutuhkan sampel dari distribusi probabilitas *joint* dan probabilitas *marginal* VA yang terlibat (Poole et al., 2019).

3.2 *Predictive-transformation* sebagai Bias Induktif

Percobaan awal mengindikasikan bahwa memaksimalkan $I_{\boldsymbol{\theta}}(\mathbf{Z};\, \hat{\mathbf{Z}})$ tanpa bantuan bias induktif gagal dalam menghasilkan representasi yang bermakna. Bias induktif (*inductive bias*) dapat dipandang sebagai suatu sinyal/informasi yang digunakan oleh model untuk proses pemelajaran. Pada kasus $I_{\boldsymbol{\theta}}(\mathbf{Z};\, \mathbf{T}|\hat{\mathbf{Z}})$, transformasi $\mathbf{t}$ bertindak sebagai bias induktif yang mempengaruhi $\mathcal{Z}$. Karena itu, salah satu dari $\hat{\mathbf{z}}$ atau $\mathbf{z}$ haruslah dapat bertindak sebagai bias induktif. Hal ini menjadi masalah karena $\mathbf{z}$ dan $\hat{\mathbf{z}}$ merupakan variabel yang ingin dimodelkan oleh objektif $I_{\boldsymbol{\theta}}(\mathbf{Z};\, \hat{\mathbf{Z}})$.

Masalah sebelumnya dapat diatasi dengan terlebih dahulu mempelajari salah satu representasi $\hat{\mathbf{z}}$ atau $\mathbf{z}$ secara terpisah. Representasi tersebut dapat memanfaatkan model representasi berbasis *data-likelihood* (*autoencoder* dan GAN) maupun model representasi terawasi sendiri. Kemudian, representasi yang telah dipelajari sebelumnya, akan bertindak sebagai bias induktif untuk optimisasi $I_{\boldsymbol{\theta}}(\mathbf{Z};\, \hat{\mathbf{Z}})$.

Pada penelitian ini, diperkenalkan model representasi *predictive-transformation* yang bertugas untuk mempelajari representasi $\mathbf{z}$. Model ini terinspirasi dari pemelajaran terawasi sendiri yang diperkenalkan oleh Gidaris et al. (2018). Model tersebut menerima masukan berupa citra $\hat{\mathbf{x}}$ yang sudah dirotasi secara acak. Model tersebut kemudian diminta untuk memprediksi operasi rotasi $\mathbf{t}$ yang diterapkan pada citra tersebut ($p(\mathbf{t}|\mathbf{t}\hat{\mathbf{x}})$). Setelah selesai dilatih, lapisan tersembunyi $\mathbf{z}$ pada model ini dapat dimanfaatkan sebagai fitur untuk keperluan tugas pemelajaran lain yang umumnya teranotasi. *Predictive-transformation* memperluas operasi rotasi menjadi transformasi proyektif (dijelaskan lebih detail pada bab selanjutnya). Tugas memprediksi transformasi $\mathbf{t}$ dapat dipandang sebagai upaya memaksimalkan ITB $I_{\hat{\boldsymbol{\theta}}}(\mathbf{T};\mathbf{Z})$, dengan $\hat{\boldsymbol{\theta}}$ menyatakan parameter model pemelajaran. Fungsi objektif tersebut mirip dengan fungsi objektif AVT yang memaksimalkan $I_{\boldsymbol{\theta}}(\mathbf{T};\mathbf{Z}|\hat{\mathbf{Z}})$. Perbedaan terletak pada ketiadaan $\hat{\mathbf{z}}$ pada fungsi objektif *predictive-transformation*. Perhitungan ITB pada *predictive-transformation* membutuhkan distribusi $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ dan $p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z})$. Distribusi $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ ditentukan berdasarkan model pemelajaran terparameterisasi oleh $\hat{\boldsymbol{\theta}}$ yang mengekstraksi representasi citra $\mathbf{t}\hat{\mathbf{x}}$. Namun, model pemelajaran tersebut

tidak dapat secara bersamaan merepresentasikan $p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z})$ melalui parameter $\hat{\boldsymbol{\theta}}$. Selain itu, *predictive-transformation* hanya memiliki akses terhadap sampel $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z},\mathbf{t})$.

Masalah di atas kemudian diatasi dengan mengestimasi batas bawah $I_{\hat{\boldsymbol{\theta}}}(\mathbf{T};\mathbf{Z})$ melalui estimasi batas bawah BA (dapat pula diestimasi melalui metode lain seperti InfoNCE) (Agakov, 2004). Metode estimasi ini memperkenalkan suatu distribusi variasional $q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})$ yang bertugas untuk mengaproksimasi $p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z})$ dengan $\check{\boldsymbol{\phi}}$ menyatakan parameter distribusi q. KL *divergence* $\mathrm{KL}(p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z})\|q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z}))$ dilibatkan untuk mengukur perbedaan statistik antara $p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z})$ dan $q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})$. Secara matematis, estimasi batas bawah BA $I_{\hat{\boldsymbol{\theta}}}(\mathbf{T};\mathbf{Z})$ diturunkan sebagai berikut:

$$
\begin{aligned}
I_{\hat{\boldsymbol{\theta}}}(\mathbf{T};\mathbf{Z}) &= H(\mathbf{T}) - H(\mathbf{T}|\mathbf{Z}) \\
&= H(\mathbf{T}) + \mathbb{E}_{p_{\hat{\boldsymbol{\theta}}}(\mathbf{t},\mathbf{z})}\left[\log p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z})\right] \\
&= H(\mathbf{T}) + \mathbb{E}_{p_{\hat{\boldsymbol{\theta}}}(\mathbf{t},\mathbf{z})}\left[\log q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})\right] \\
&\quad + \mathbb{E}_{p(\mathbf{z})}\left[\mathrm{KL}(p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z})\,\|\,q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z}))\right] \\
&\geq H(\mathbf{T}) + \mathbb{E}_{p_{\hat{\boldsymbol{\theta}}}(\mathbf{t},\mathbf{z})}\left[\log q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})\right]
\end{aligned} \tag{3.4}
$$

Selanjutnya, estimasi ITB *predictive-transformation* didefinisikan sebagai $\mathcal{L}_{\text{pred}} = H(\mathbf{T}) + \mathbb{E}_{p_{\hat{\boldsymbol{\theta}}}(\mathbf{t},\mathbf{z})}\left[\log q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})\right]$. Dengan memaksimalkan $\mathbb{E}_{p_{\hat{\boldsymbol{\theta}}}(\mathbf{t},\mathbf{z})}\left[\log q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})\right]$ maka akan meminimalkan $\mathrm{KL}(p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z})\,\|\,q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z}))$. Ketika $\mathrm{KL}(p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z})\,\|\,q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})) = 0$ maka $p_{\hat{\boldsymbol{\theta}}}(\mathbf{t}|\mathbf{z}) = q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})$. Berdasarkan pertidaksamaan tersebut, $H(\mathbf{T})$ tidak terparameterisasi oleh $\hat{\boldsymbol{\theta}}$ maupun $\check{\boldsymbol{\phi}}$. Dengan memanfaatkan sifat kemonotonan logaritma, optimisasi *predictive-transformation* dapat dituliskan sebagai:

$$
\hat{\boldsymbol{\theta}}^*, \check{\boldsymbol{\phi}}^* = \max_{\hat{\boldsymbol{\theta}},\check{\boldsymbol{\phi}}} \mathbb{E}_{p_{\hat{\boldsymbol{\theta}}}(\mathbf{t},\mathbf{z})}\left[\log q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})\right] = \min_{\hat{\boldsymbol{\theta}},\check{\boldsymbol{\phi}}} \mathbb{E}_{p_{\hat{\boldsymbol{\theta}}}(\mathbf{t},\mathbf{z})}\left[-\log q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})\right] \tag{3.5}
$$

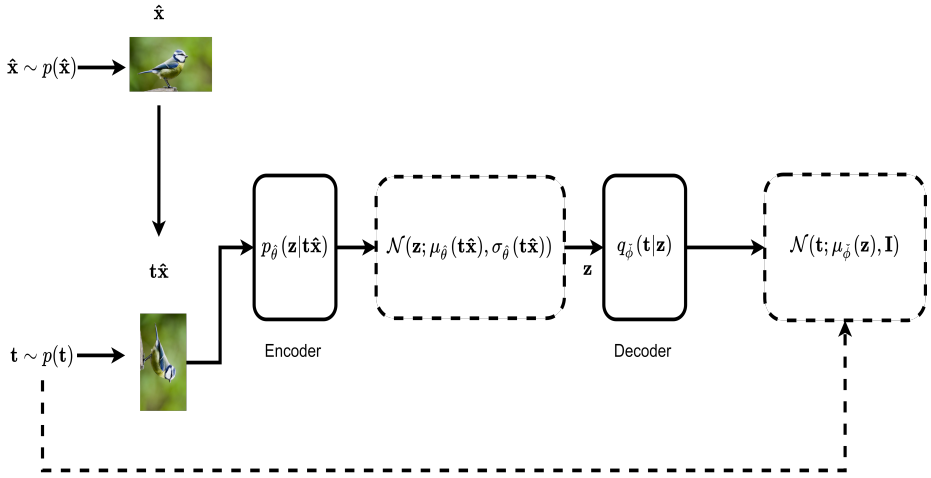
Setelah selesai dilatih, *predictive-transformation* akan dilibatkan untuk melatih TEV. *Predictive-transformation* akan bertugas untuk menginferensi representasi $\mathbf{z}$ diberikan $\mathbf{t}\hat{\mathbf{x}}$. Pemaparan lebih lanjut terdapat pada sub-bab selanjutnya.

Penelitian ini, mengikuti asumsi Qi et al. (2019) bahwa $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ berasal dari suatu keluarga distribusi Gaussian multivariat terfaktorisasi, artinya $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}}) = \prod_j^M p_{\hat{\boldsymbol{\theta}}_j}(\mathbf{z}_j|\mathbf{t}\hat{\mathbf{x}})$, dengan j menyatakan komponen ke-j. Notasi M menyatakan dimensionalitas representasi $\mathbf{z}$. Distribusi Gaussian multivariat terfaktorisasi menjamin *covariance* distribusi $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ berupa matriks diagonal berukuran $M \times M$. *Covariance* tersebut dapat direpresentasikan sebagai suatu vektor $\boldsymbol{\sigma}_{\hat{\theta}}$ berukuran M. Akhirnya, $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ dapat dituliskan sebagai $\mathcal{N}(\mathbf{z};\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}},\boldsymbol{\sigma}_{\hat{\boldsymbol{\theta}}})$. Notasi $\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}$ menyatakan vektor *mean* berukuran M. Informasi timbal balik $\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}$ dan $\boldsymbol{\sigma}_{\hat{\boldsymbol{\theta}}}$ terparameterisasi oleh $\hat{\boldsymbol{\theta}}$.

Distribusi $q_{\check{\boldsymbol{\phi}}}(\mathbf{t}|\mathbf{z})$ juga diasumsikan berasal dari keluarga distribusi Gaussian

multivariat terfaktorisasi. *Mean* ditribusi $q_{\check{\phi}}(\mathbf{t}|\mathbf{z})$ dinyatakan oleh vektor $\boldsymbol{\mu}_{\check{\phi}}$ berukuran L, dengan L menyatakan dimensionalitas dari operasi transformasi $\mathbf{t}$. Sementara itu, *covariance* distribusi dipilih berupa matriks identitas $\mathbb{I}$ berukuran $L \times L$. Hal ini berarti *variance* masing-masing komponen $q_{\check{\phi}_j}(\mathbf{t}_j|\mathbf{z})$ bernilai 1. Pemilihan *covariance* tersebut bertujuan untuk mengurangi kompleksitas model. Selanjutnya, matriks $\mathbb{I}$ dapat direpresentasikan sebagai suatu vektor-1 berukuran L. Dengan demikian, $q_{\check{\phi}}(\mathbf{t}|\mathbf{z})$ dapat ditulis sebagai $\mathcal{N}(\mathbf{t}; \boldsymbol{\mu}_{\check{\phi}}, \mathbf{1})$. *Mean* $\boldsymbol{\mu}_{\check{\phi}}$ dan standar-deviasi $\boldsymbol{\sigma}_{\check{\phi}}$ terparameterisasi oleh $\check{\phi}$.

Predictive-transformation mengadopsi arsitektur *autoencoder*. *Encoder* $E_{\hat{\boldsymbol{\theta}}}$ merepresentasikan $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ sementara *decoder* $D_{\check{\phi}}$ menyatakan $q_{\check{\phi}}(\mathbf{t}|\mathbf{z})$. Parameter distribusi $\hat{\boldsymbol{\theta}}$ dan $\check{\phi}$ merupakan parameter dari *encoder* E dan *decoder* D, artinya karakteristik distribusi p dan q ditentukan oleh *encoder* dan *decoder*. Gambar 3.2 menunjukkan ilustrasi *predictive-transformation*.



Gambar 3.2: Ilustrasi *predictive-transformation* dengan batas bawah Barber-Agakov. Berbeda dengan AVT, *decoder* pada *predictive-transformation* hanya menerima $\mathbf{z}$ sebagai masukan.

Berdasarkan Gambar 3.2, luaran dari *encoder* E dan *decoder* D berupa parameter distribusi. Luaran tersebut tidak dapat digunakan secara langsung untuk melatih *predictive-transformation*. Dibutuhkan sampel dari distribusi $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ untuk melakukan inferensi terhadap $q_{\check{\phi}}(\mathbf{t}|\mathbf{z})$ (prinsip *empirical risk minimization*). Namun, melakukan *sampling* secara langsung dari distribusi $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$ menyebabkan model kehilangan informasi mengenai $\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}$ dan $\boldsymbol{\sigma}_{\hat{\boldsymbol{\theta}}}$. Kedua informasi ini dibutuhkan untuk dapat melakukan *back-propagation* yang diperlukan pada saat proses optimisasi. Karena itu, diperkenalkan metode *reparameterization-trick* untuk melakukan *sampling* terhadap $\mathbf{z}$. Metode ini dipopulerkan oleh Kingma and Welling (2014) untuk melatih *variational autoencoder*. Secara matematis, *reparameterization-trick* untuk representasi $\mathbf{z}$ dapat dituliskan sebagai:

$$\mathbf{z}^{(i)} = \boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) + \boldsymbol{\sigma}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) \odot \boldsymbol{\epsilon}^{(i)} \tag{3.6}$$

Algorithm 1: *Predictive-transformation* dengan batas bawah Barber-Agakov

Required : $\{\hat{\mathbf{x}}^{(1)},...,\hat{\mathbf{x}}^{(K)}\}$ dan α ;
- Definisikan $\hat{\mathcal{L}}_{\text{pred}} = 0$;
for $i = 1,2,...,K$ **do**
  - Pilih $\mathbf{t}^{(i)} \sim p(\mathbf{t})$;
  - Pilih $\boldsymbol{\epsilon}^{(i)} \sim \mathcal{N}(\boldsymbol{\epsilon};\mathbf{0},\mathbb{I})$;
  - $\mathbf{z}^{(i)} \leftarrow \boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) + \boldsymbol{\sigma}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) \odot \boldsymbol{\epsilon}^{(i)}$ (*reparameterization-trick*);
  - $\hat{\mathcal{L}}_{\text{pred}} \leftarrow \hat{\mathcal{L}}_{\text{pred}} + \log \mathcal{N}(\mathbf{t}^{(i)}; \boldsymbol{\mu}_{\check{\boldsymbol{\phi}}}(\mathbf{z}^{(i)}),\mathbb{I})$;
end
- $\check{\boldsymbol{\phi}} \leftarrow \check{\boldsymbol{\phi}} - \alpha\nabla_{\check{\boldsymbol{\phi}}} \frac{-\hat{\mathcal{L}}_{\text{pred}}}{K}$;
- $\hat{\boldsymbol{\theta}} \leftarrow \hat{\boldsymbol{\theta}} - \alpha\nabla_{\hat{\boldsymbol{\theta}}} \frac{-\hat{\mathcal{L}}_{\text{pred}}}{K}$;

dengan $1 \leq i \leq K$ menyatakan indeks sampel berukuran K. Notasi $\odot$ menyatakan operasi perkalian Hadamard. *Noise* $\boldsymbol{\epsilon}^{(i)}$ merupakan suatu vektor acak berukuran M yang disampel dari distribusi Gaussian normal $\mathcal{N}(\boldsymbol{\epsilon};\mathbf{0},\mathbb{I})$. Notasi $\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}$ dan $\boldsymbol{\sigma}_{\hat{\boldsymbol{\theta}}}$ menyatakan parameter ditribusi $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}^{(i)}|\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}$ untuk setiap setiap i. Kondisi ketika parameter $\boldsymbol{\theta}$ berperan sebagai parameter global dinamakan sebagai *amortized-inference* (Kingma and Welling, 2014). Karena *decoder* $D_{\check{\boldsymbol{\phi}}}$ menerima masukan dari setiap sampel $\mathbf{z}^{(i)}$, maka parameter $\check{\boldsymbol{\phi}}$ juga merupakan *amortized inference*. Berdasarkan Persamaan 3.6, sampel yang dibangkitkan mewarisi informasi mengenai $\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)})$ dan $\boldsymbol{\sigma}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)})$ secara eksplisit. Dengan demikian, *back-propagation* dapat dilakukan untuk setiap parameter jaringan. Untuk K sampel pasangan $(\mathbf{t},\mathbf{z})$, $\mathcal{L}_{\text{pred}}$ diestimasi melalui estimasi Monte-Carlo. Sama seperti kebanyakan model pemelajaran dalam, *predictive-transformation* dioptimisasi menggunakan metode iteratif *stochastic gradient descent* (Goodfellow et al., 2016). Langkah-langkah untuk melatih *predictive-transformation* dalam satu *mini-batch* untuk satu iterasi dapat dilihat pada Algoritma 1.

3.3 TEV dengan Batas Bawah Barber-Agakov

Model representasi TEV dilatih dengan memanfaatkan *predictive-transformation* yang bertugas untuk menyediakan representasi $\mathbf{z}$. *Predictive-transformation* tidak lagi dioptimisasi pada saat melatih TEV. Proses pemelajaran TEV juga terkendala oleh proses perhitungan ITB secara eksak. Pada penelitian ini, diperkenalkan dua metode untuk mengestimasi batas bawah $I_{\boldsymbol{\theta}}(\mathbf{Z};\hat{\mathbf{Z}})$: Barber-Agakov (TEVBA) (Agakov, 2004) dan InfoNCE (TEVInfoNCE) (Oord et al., 2018). Sub-bab ini fokus membahas TEV dengan estimasi batas bawah BA.

Estimasi batas bawah BA untuk ITB $I_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{Z};\hat{\mathbf{Z}})$ dapat diturunkan melalui cara yang sama dengan *predictive-transformation*. Perbedaan terletak pada parameter dan variabel

yang terlibat pada proses penurunan batas bawah. Pada kasus TEVBA terdapat dua parameter yang terlibat yaitu $\hat{\boldsymbol{\theta}}$ yang bertugas menginferensi $\mathbf{z}$ dan $\boldsymbol{\theta}$ yang bertugas untuk menginferensi $\hat{\mathbf{z}}$. Selanjutnya, diperkenalkan distribusi variasional $q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}})$ yang bertujuan untuk mengaproksimasi $p_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{z}|\hat{\mathbf{z}})$. Secara matematis, estimasi batas bawah BA untuk $I_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{Z};\hat{\mathbf{Z}})$ dapat diturunkan sebagai berikut:

$$\begin{aligned}
I_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{Z};\hat{\mathbf{Z}}) &= H(\mathbf{Z}) - H(\mathbf{Z}|\hat{\mathbf{Z}}) \\
&= H(\mathbf{Z}) + \mathbb{E}_{p_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{t},\hat{\mathbf{z}},\mathbf{z})}\left[\log p_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{z}|\hat{\mathbf{z}})\right] \\
&= H(\mathbf{Z}) + \mathbb{E}_{p_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{t},\hat{\mathbf{z}},\mathbf{z})}\left[\log q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}})\right] \\
&\quad + \mathbb{E}_{p(\hat{\mathbf{z}})}\left[\mathrm{KL}(p_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{z}|\hat{\mathbf{z}}) \,||\, q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}}))\right] \\
&\geq H(\mathbf{Z}) + \mathbb{E}_{p_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{t},\hat{\mathbf{z}},\mathbf{z})}\left[\log q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}})\right]
\end{aligned} \tag{3.7}$$

Selanjutnya, estimasi batas bawah BA untuk $I_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{Z};\hat{\mathbf{Z}})$ didefinisikan sebagai $\mathcal{L}_{\text{TEVBA}} = H(\mathbf{Z}) + \mathbb{E}_{p_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{t},\hat{\mathbf{z}},\mathbf{z})}\left[\log q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}})\right]$. Karena $H(\mathbf{Z})$ tidak terparameterisasi oleh $\boldsymbol{\theta}$ maupun $\boldsymbol{\phi}$, maka proses optimisasi cukup memaksimalkan $\mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{z},\hat{\mathbf{z}})}\left[q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}})\right]$ terhadap parameter $\boldsymbol{\theta}$ dan $\boldsymbol{\phi}$.

$$\boldsymbol{\theta}^*, \boldsymbol{\phi}^* = \max_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathbb{E}_{p_{\hat{\boldsymbol{\theta}},\boldsymbol{\theta}}(\mathbf{t},\hat{\mathbf{z}},\mathbf{z})}\left[\log q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}})\right] = \min_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathbb{E}_{p_{\hat{\boldsymbol{\theta}},\boldsymbol{\theta}}(\mathbf{t},\hat{\mathbf{z}},\mathbf{z})}\left[-\log q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}})\right] \tag{3.8}$$

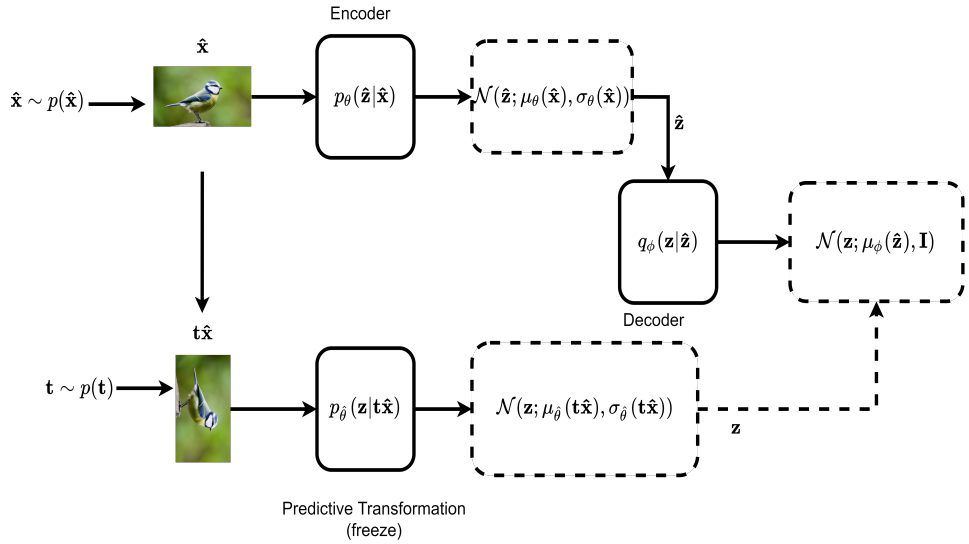
Model TEVBA menggunakan asumsi yang sama dengan *predictive-transformation*. Distribusi $p_{\boldsymbol{\theta}}(\hat{\mathbf{z}}|\hat{\mathbf{x}})$ mengikuti distribusi Gaussian multivariat terfaktorisasi $\mathcal{N}(\hat{\mathbf{z}};\boldsymbol{\mu}_{\theta},\boldsymbol{\sigma}_{\theta})$, sementara $q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}})$ mengikuti distribusi Gaussian $\mathcal{N}(\mathbf{z};\boldsymbol{\mu}_{\boldsymbol{\phi}},\mathbf{1})$. Parameter distribusi $\boldsymbol{\mu}_{\boldsymbol{\theta}},\boldsymbol{\sigma}_{\boldsymbol{\theta}}$ dan $\boldsymbol{\mu}_{\boldsymbol{\phi}}$ berupa vektor berukuran M.

Model TEVBA juga mengadopsi arsitektur *autoencoder*. *Encoder* $E_{\boldsymbol{\theta}}$ merepresentasikan $p_{\boldsymbol{\theta}}(\hat{\mathbf{z}}|\hat{\mathbf{x}})$, sementara *decoder* $D_{\boldsymbol{\phi}}$ merepresentasikan $q_{\boldsymbol{\phi}}(\mathbf{z}|\hat{\mathbf{z}})$. Terdapat pula *encoder* dari *predictive-transformation* $E_{\hat{\boldsymbol{\theta}}}$ yang merepresentasikan $p_{\hat{\boldsymbol{\theta}}}(\mathbf{z}|\mathbf{t}\hat{\mathbf{x}})$. *Encoder* $E_{\hat{\boldsymbol{\theta}}}$ tidak dioptimisasi tahap ini. Ilustrasi mengenai TEVBA dapat dilihat pada Gambar 3.3

Algorithm 2: Transformasi-ekuivarian variasional (TEV) dengan estimasi batas bawah Barber-Agakov

Required : $\{\hat{\mathbf{x}}^{(1)},...,\hat{\mathbf{x}}^{(K)}\}$ dan α ;
definisikan $\hat{\mathcal{L}}_{\text{TEVBA}} = 0$;
for $i = 1,2,...,K$ **do**
- Pilih $\mathbf{t}^{(i)} \sim p(\mathbf{t})$;
- Pilih $\boldsymbol{\epsilon}^{(i)} \sim \mathcal{N}(\boldsymbol{\epsilon};\mathbf{0},\mathbb{I})$;
- $\mathbf{z}^{(i)} \leftarrow \boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) + \boldsymbol{\sigma}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) \odot \boldsymbol{\epsilon}^{(i)}$ (*reparameterization-trick*) ;
- Pilih $\hat{\boldsymbol{\epsilon}}^{(i)} \sim \mathcal{N}(\hat{\boldsymbol{\epsilon}};\mathbf{0},\mathbb{I})$;
- $\hat{\mathbf{z}}^{(i)} \leftarrow \boldsymbol{\mu}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(i)}) + \boldsymbol{\sigma}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(i)}) \odot \hat{\boldsymbol{\epsilon}}^{(i)}$ (*reparameterization-trick*);
- $\hat{\mathcal{L}}_{\text{TEVBA}} \leftarrow \hat{\mathcal{L}}_{\text{TEVBA}} + \log \mathcal{N}(\mathbf{z}^{(\mathbf{i})}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\hat{\mathbf{z}}^{(i)}),\mathbb{I})$;

end
- $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \alpha\nabla_{\boldsymbol{\phi}} \frac{-\hat{\mathcal{L}}_{\text{TEVBA}}}{K}$;
- $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha\nabla_{\boldsymbol{\theta}} \frac{-\hat{\mathcal{L}}_{\text{TEVBA}}}{K}$;



Gambar 3.3: Ilustrasi TEV dengan batas bawah Barber-Agakov. *Encoder* $E_{\boldsymbol{\theta}}$ menerima masukan berupa citra $\mathbf{z}$, sementara *Encoder* $E_{\hat{\boldsymbol{\theta}}}$ menerima masukan berupa citra transformasi $\mathbf{t}\hat{\mathbf{x}}$.

Proses *sampling* $\hat{\mathbf{z}}$ berdasarkan *reparameterization-trick* yang sudah dijelaskan pada sub-bab sebelumnya. *Reparameterization-trick* untuk representasi $\mathbf{z}^{(i)}$ dapat dituliskan sebagai:

$$\hat{\mathbf{z}}^{(i)} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(i)}) + \boldsymbol{\sigma}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(i)}) \odot \hat{\boldsymbol{\epsilon}}^{(i)} \tag{3.9}$$

dengan $\hat{\boldsymbol{\epsilon}}^{(i)}$ merupakan vektor acak yang dipilih dari distribusi Gaussian normal $\mathcal{N}(\hat{\boldsymbol{\epsilon}};\mathbf{0},\mathbb{I})$ untuk $1 \leq i \leq K$ merupakan indeks untuk sampel berukuran K. Langkah-langkah untuk melatih TEVBA untuk satu *mini-batch* dalam satu iterasi dapat dilihat pada Algoritma 2.

3.4 TEV dengan Batas Bawah InfoNCE

Sub-bab ini membahas TEV dengan estimasi batas bawah InfoNCE. Berdasarkan Persamaan 2.8, ITB $I(\mathbf{X},\mathbf{Y})$ dapat didefinisikan sebagai ekspektasi dari rasio densitas $\frac{p(\mathbf{x}|\mathbf{y})}{p(\mathbf{x})}$, untuk VA $\mathbf{X}$ dan $\mathbf{Y}$. Pada kasus TEV, $I_{\theta,\hat{\theta}}(\mathbf{Z};\hat{\mathbf{Z}})$, didefinisikan sebagai:

$$I_{\theta,\hat{\theta}}(\mathbf{Z};\hat{\mathbf{Z}}) = \mathbb{E}_{p_{\theta,\hat{\theta}}(\mathbf{t},\hat{\mathbf{z}},\mathbf{z})}\left[\log \frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}|\mathbf{z})}{p(\hat{\mathbf{z}})}\right] \tag{3.10}$$

Misalkan terdapat K sampel pasangan $\{(\hat{\mathbf{z}}^{(j)},\mathbf{z}^{(j)})\}_{j=1}^{K}$. Berdasarkan (Oord et al., 2018), rasio densitas untuk suatu pasangan $(\hat{\mathbf{z}}^{(i)},\mathbf{z}^{(i)})$, $1 \leq i,j \leq K$ proporsional terhadap suatu fungsi eksponensial yang melibatkan fungsi estimator $g_{\hat{\phi}}$:

$$\exp(g_{\hat{\phi}}(\hat{\mathbf{z}}^{(i)},\mathbf{z}^{(i)})) \propto \frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}\,|\,\mathbf{z}^{(i)})}{p(\hat{\mathbf{z}}^{(i)})} \tag{3.11}$$

Notasi $\propto$ menyatakan proporsionalitas antara kedua suku. Fungsi estimator $g_{\hat{\phi}}$ direpresentasikan oleh suatu pemelajaran dalam yang terparameterisasi oleh $\hat{\phi}$. Luaran dari fungsi estimator g menjadi faktor pemangkatan terhadap bilangan natural e. Fungsi estimator $g_{\hat{\phi}}$ dapat pula digantikan oleh dua fungsi estimator $g_{1\tilde{\phi}}$ dan $g_{2\phi'}$ yang masing-masing terparameterisasi oleh $\tilde{\phi}$ dan ϕ'. Luaran dari fungsi estimator g_1 dan g_2 dikombinasikan melalui operasi perkalian Hadamard $\odot$. Secara matematis:

$$\exp(g_{1\tilde{\phi}}(\hat{\mathbf{z}}^{(i)}) \odot g_{2\phi'}(\mathbf{z}^{(i)})) \propto \frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}\,|\,\mathbf{z}^{(i)})}{p(\hat{\mathbf{z}}^{(i)})} \tag{3.12}$$

Rasio densitas memiliki sifat tidak ternormalisasi, artinya hasil integralnya tidak sama dengan 1 (Oord et al., 2018). Karena itu, dibutuhkan suatu faktor normalisasi $Z = \sum_j \exp(f_{\hat{\phi}}(\hat{\mathbf{z}}^{(j)},\mathbf{z}^{(i)}))$ untuk suatu i dan $1 \leq j \leq K$. Dari sudut pandang model berbasis energi, Z disebut sebagai fungsi partisi (Goodfellow et al., 2016). Dengan menggabungkan informasi sebelumnya, ITB $I_{\theta}(\mathbf{Z};\hat{\mathbf{Z}})$ dengan estimasi batas bawah InfoNCE $\mathcal{L}_{\text{TEVInfoNCE}}$ didefinisikan sebagai:

$$I_{\theta}(\hat{\mathbf{Z}};\mathbf{Z}) \geq \mathop{\mathbb{E}}_{\prod_j p_{\theta,\hat{\theta}}(\mathbf{t}^{(j)},\hat{\mathbf{z}}^{(j)},\mathbf{z}^{(j)})}\left[\frac{1}{K}\sum_{i=1}^{K}\log\frac{\exp(g_{\hat{\phi}}(\hat{\mathbf{z}}^{(i)},\mathbf{z}^{(i)}))}{\sum_{j=1}^{K}\exp(g_{\hat{\phi}}(\hat{\mathbf{z}}^{(j)},\mathbf{z}^{(i)}))}\right] \tag{3.13}$$

$$\geq \mathop{\mathbb{E}}_{\prod_j p_{\theta,\hat{\theta}}(\mathbf{t}^{(j)},\hat{\mathbf{z}}^{(j)},\mathbf{z}^{(j)})}\left[\frac{1}{K}\sum_{i=1}^{K}\log\frac{\exp(g_{1\tilde{\phi}}(\hat{\mathbf{z}}^{(i)}) \odot g_{2\phi'}(\mathbf{z}^{(i)}))}{\sum_{j=1}^{K}\exp(g_{1\tilde{\phi}}(\hat{\mathbf{z}}^{(j)}) \odot g_{2\phi'}(\mathbf{z}^{(i)}))}\right] \tag{3.14}$$

$$\tag{3.15}$$

Pasangan $(\hat{\mathbf{z}}^{(i)},\mathbf{z}^{(i)})$ disebut sebagai pasangan positif, sementara $(\hat{\mathbf{z}}^{(j)},\mathbf{z}^{(i)})$ disebut sebagai pasangan negatif, untuk $i \neq j; 1 \leq i,j \leq K$. Pertidaksamaan 3.13 dinamakan TEVInfoNCE

concatenated, sementara pertidaksamaan Pertidaksamaan 3.14 dinamakan TEVInfoNCE *separated*. Dengan memanfaatkan kemonotonan logaritma, optimisasi TEVInfoNCE dituliskan sebagai:

$$\boldsymbol{\theta}^*, \hat{\phi}^* = \max_{\boldsymbol{\theta}, \hat{\phi}} \mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K} \log \frac{\exp(g_{\hat{\phi}}(\hat{\mathbf{z}}^{(i)}, \mathbf{z}^{(i)}))}{\sum_{j=1}^{K} \exp(g_{\hat{\phi}}(\hat{\mathbf{z}}^{(j)}, \mathbf{z}^{(i)}))}\right] \tag{3.16}$$

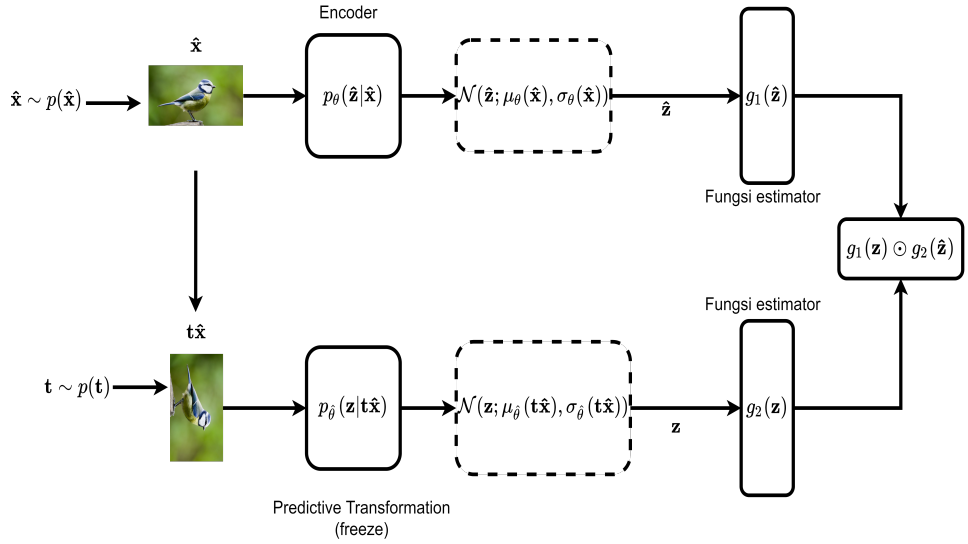
$$= \min_{\boldsymbol{\theta}, \hat{\phi}} \mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K} -\log \frac{\exp(g_{\hat{\phi}}(\hat{\mathbf{z}}^{(i)}, \mathbf{z}^{(i)}))}{\sum_{j=1}^{K} \exp(g_{\hat{\phi}}(\hat{\mathbf{z}}^{(j)}, \mathbf{z}^{(i)}))}\right] \tag{3.17}$$

$$\boldsymbol{\theta}^*, \tilde{\phi}^*, \phi'^* = \max_{\boldsymbol{\theta}, \tilde{\phi}, \phi'} \mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K} \log \frac{\exp(g_{1\tilde{\phi}}(\hat{\mathbf{z}}^{(i)}) \odot g_{2\phi'}(\mathbf{z}^{(i)}))}{\sum_{j=1}^{K} \exp(g_{1\tilde{\phi}}(\hat{\mathbf{z}}^{(j)}) \odot g_{2\phi'}(\mathbf{z}^{(i)}))}\right] \tag{3.18}$$

$$= \min_{\boldsymbol{\theta}, \tilde{\phi}, \phi'} \mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K} -\log \frac{\exp(g_{1\tilde{\phi}}(\hat{\mathbf{z}}^{(i)}) \odot g_{2\phi'}(\mathbf{z}^{(i)}))}{\sum_{j=1}^{K} \exp(g_{1\tilde{\phi}}(\hat{\mathbf{z}}^{(j)}) \odot g_{2\phi'}(\mathbf{z}^{(i)}))}\right] \tag{3.19}$$

dengan ekspektasi terhadap $\prod_j p_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(j)}, \hat{\mathbf{z}}^{(j)}, \mathbf{z}^{(j)})$. Model TEVInfoNCE sangat bergantung pada jumlah sampel $\{(\hat{\mathbf{z}}^{(j)}, \mathbf{z}^{(j)})\}_{j=1}^{K}$. Secara matematis, TEVInfoNCE memenuhi pertidaksamaan $I_{\boldsymbol{\theta},\hat{\boldsymbol{\theta}}}(\mathbf{Z}; \hat{\mathbf{Z}}) \geq \mathcal{L}_{\text{TEVInfoNCE}} + \log(K)$ (Oord et al., 2018). Pembuktian lengkap dapat dilihat pada **Appendix A.1**.

Model TEVInfoNCE sedikit berbeda dengan TEVBA. *Encoder* TEVInfoNCE memiliki peran yang sama dengan *encoder* TEVBA, yaitu membangkitkan $\hat{\mathbf{z}}$. Encoder *predictive-transformation* $E_{\hat{\boldsymbol{\theta}}}$ juga masih terlibat untuk membangkitkan $\mathbf{z}$. Desain *encoder* TEVInfoNCE sama seperti *encoder* TEVBA. Perbedaan terletak pada fungsi estimator yang menggantikan *decoder* pada TEVBA. Gambar 3.4 menunjukkan ilustrasi TEVInfoNCE. Arsitektur ini memiliki kemiripan dengan model pemelajaran terawasi sendiri SimCLR (Chen et al., 2020). Perbedaan terletak pada *encoder* SimCLR yang bersifat *siamese* (menerima masukan $\hat{\mathbf{z}}$ dan $\mathbf{z}$). Selain itu, *encoder* SimCLR bersifat deterministik, sementara *encoder* TEVInfoNCE bersifat probabilistik.



Gambar 3.4: Ilustrasi TEVInfoNCE *separated*. *Encoder* bekerja seperti TEV dengan batas bawah Barber-Agakov. Fungsi estimator g_1 menerima $\hat{\mathbf{z}}$ sebagai masukan, sementara $\mathbf{z}$ menjadi masukan bagi fungsi estimator g_2. Alternatif lain yaitu TEVInfoNCE *concatenated* yang menggunakan satu fungsi estimator g dan menerima masukan berupa hasil penggabungan $\hat{\mathbf{z}}$ dan $\mathbf{z}$.

Encoder TEVInfoNCE masih bergantung pada *reparameterization-trick* untuk membangkitkan $\hat{\mathbf{z}}$. Proses optimisasi sama seperti TEVBA, yaitu melalui *stochastic gradient descent*. Algoritma lengkap TEV dengan batas bawah InfoNCE untuk suatu *mini-batch* dalam satu iterasi dapat dilihat pada Algoritma 3.

Algorithm 3: Transformasi-ekuivarian variasional dengan batas bawah InfoNCE

Required : $\{\hat{\mathbf{x}}^{(1)},...,\hat{\mathbf{x}}^{(K)}\}$ dan $\boldsymbol{\alpha}$;
- Definisikan $\mathcal{L}_{\text{TEVInfoNCE}} = 0$;
for $i = 1, 2, ..., K$ **do**
  - Pilih $\mathbf{t}^{(i)} \sim p(\mathbf{t})$;
  - Pilih $\boldsymbol{\epsilon}^{(i)} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbb{I})$;
  - $\mathbf{z}^{(i)} \leftarrow \boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) + \boldsymbol{\sigma}_{\hat{\boldsymbol{\theta}}}(\mathbf{t}^{(i)}\hat{\mathbf{x}}^{(i)}) \odot \boldsymbol{\epsilon}^{(i)}$ (*reparameterization-trick*);
  - Pilih $\hat{\epsilon}^{(i)} \sim \mathcal{N}(\hat{\epsilon}; \mathbf{0}, \mathbb{I})$;
  - $\hat{\mathbf{z}}^{(i)} \leftarrow \boldsymbol{\mu}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(i)}) + \boldsymbol{\sigma}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(i)}) \odot \hat{\boldsymbol{\epsilon}}^{(i)}$ (*reparameterization-trick*);
  - Definisikan $\sum_j = 0$;
  for *j = 1, 2, ..., K* **do**
    - Pilih $\hat{\epsilon}^{(j)} \sim \mathcal{N}(\hat{\epsilon}; \mathbf{0}, \mathbb{I})$;
    - $\hat{\mathbf{z}}^{(j)} \leftarrow \boldsymbol{\mu}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(j)}) + \boldsymbol{\sigma}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{(j)}) \odot \hat{\boldsymbol{\epsilon}}^{(j)}$ (*reparameterization-trick*);
    if *concatenated* **then**
      - $\sum_j \leftarrow \sum_j + \exp(g_{\hat{\phi}}(\hat{\mathbf{z}}^{(j)}, \mathbf{z}^{(i)}))$;
    else
      - $\sum_j \leftarrow \sum_j + \exp(g_{1\tilde{\phi}}(\hat{\mathbf{z}}^{(j)}) \odot g_{2\phi'}(\mathbf{z}^{(i)}))$;
    end
  end
  if *concatenated* **then**
    - $\mathcal{L}_{\text{TEVInfoNCE}} \leftarrow \mathcal{L}_{\text{TEVInfoNCE}} + \log \frac{\exp(g_{\hat{\phi}}(\hat{\mathbf{z}}^{(i)}, \mathbf{z}^{(i)}))}{\sum_j}$;
  else
    - $\mathcal{L}_{\text{TEVInfoNCE}} \leftarrow \mathcal{L}_{\text{TEVInfoNCE}} + \log \frac{\exp(g_{1\tilde{\phi}}(\hat{\mathbf{z}}^{(i)}) \odot g_{2\phi'}(\mathbf{z}^{(i)}))}{\sum_j}$;
  end
end
- $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \frac{-\mathcal{L}_{\text{TEVInfoNCE}}}{K})$;
if *concatenated* **then**
  - $\hat{\phi} \leftarrow \hat{\phi} - \alpha \nabla_{\hat{\phi}} \frac{-\mathcal{L}_{\text{TEVInfoNCE}}}{K}$;
else
  - $\tilde{\phi} \leftarrow \tilde{\phi} - \alpha \nabla_{\tilde{\phi}} \frac{-\mathcal{L}_{\text{TEVInfoNCE}}}{K}$;
  - $\phi' \leftarrow \phi' - \alpha \nabla_{\phi'} \frac{-\mathcal{L}_{\text{TEVInfoNCE}}}{K}$;
end

BAB 4
EKSPERIMEN

Eksperimen dilakukan untuk mengevaluasi model representasi TEV dan membandingkannya dengan model *baseline* AVT. Karena penelitian ini berfokus pada AVT, maka TEV tidak dibandingkan dengan model representasi lainnya. Metode evaluasi model representasi umumnya dengan menghadapkan model tersebut terhadap tugas pemelajaran yang sudah teranotasi. Model representasi berperan sebagai pengekstrasi fitur bagi masukan pada tugas pemelajaran tersebut. Sebagai perbandingan, Zhang et al. (2019) menghadapkan model representasi terhadap tugas klasifikasi. Sementara itu, Gidaris et al. (2018) memilih tugas segmentasi dan deteksi objek sebagai alat evaluasi bagi model representasi yang diajukan. Pada penelitian ini, TEV dihadapkan pada tugas klasifikasi, mengikuti Qi et al. (2019).

Eksperimen diawali dengan melatih *predictive transformation*. Selanjutnya, TEVBA dan TEVInfoNCE (*separated* dan *concatenated*) dilatih dengan melibatkan *predictive-transformation* yang menyediakan representasi citra hasil transformasi. Selain itu, ekperimen AVT (Qi et al., 2019) direproduksi dengan beberapa penyesuaian untuk mendapatkan hasil yang komparatif dengan model-model yang diajukan. Semua model yang sudah dilatih dijadikan sebagai pengekstraksi fitur (*feature extractor*) pada tugas klasifikasi citra. Masing-masing model dievaluasi berdasarkan rata-rata nilai galat (*error rate*) yang dihasilkan dari tugas klasifikasi. Pada penelitian ini, dipergunakan dua *benchmark dataset*: CIFAR-10 (Krizhevsky et al., 2009) dan STL-10 (Coates et al., 2011). Pemilihan *dataset* CIFAR-10 berdasarkan eksperimen utama AVT. Kemudian, *dataset* STL-10 dipilih karena karakteristiknya yang menyerupai *dataset* Imagenet namun dengan ukuran yang lebih kecil. Imagenet tercatat merupakan *dataset* citra terbesar saat ini, sehingga dipandang sebagai rujukan utama evaluasi pemelajaran mesin untuk domain citra. *Dataset* STL-10 dapat menjadi alternatif yang menjanjikan, terutama ketika ketersediaan sumber daya komputasi cukup terbatas.

4.1 Pengaturan Eksperimen

4.1.1 CIFAR-10

Dataset CIFAR-10 berjumlah 60000 yang terbagi dalam 10 kelas (setiap kelas terdiri dari 6000 data). Masing-masing data merupakan citra *red, green, blue* (RGB) berukuran 32×32 (Krizhevsky et al., 2009).

4.1.1.1 Arsitektur dan Implementasi

Arsitektur Semua *encoder* dari model yang diajukan (*predictive-transformation* dan TEV) mengadopsi arsitektur AVT, yaitu *network in network* (NIN). *Encoder* tersebut terdiri dari beberapa blok yang tersusun atas JSK, *batch normalization*, dan fungsi non-linear ReLU. Pada penelitian ini, dimensionalitas representasi $\mathbf{z}$ dan $\hat{\mathbf{z}}$ dipilih sebesar 512. Pemilihan dimensionalitas representasi berdasarkan percobaan empiris yang sebelumnya sudah dilakukan. *Decoder* maupun fungsi estimator diimplementasikan menggunakan PLB dan fungsi non-linear ReLU. Kompleksitas arsitektur *decoder* dirancang lebih sederhana dibandingkan *encoder* karena *decoder* menerima masukan yang lebih sederhana. Arsitektur lengkap model representasi pada eksperimen CIFAR-10 dapat dilihat pada **Appendix B.1**.

Arsitektur *encoder* dapat diimplementasikan tidak hanya menggunakan JSK. Sebagai contoh, arsitektur *encoder* dapat digantikan oleh PLB. Selanjutnya, citra masukan perlu direpresentasikan sebagai suatu vektor sehingga dapat dilewatkan melalui PLB. Pemilihan arsitektur, terutama pada *encoder* tentu berpengaruh terhadap representasi yang dihasilkan oleh model representasi. Secara teoritis, karakteristik JSK dalam menangani data berupa citra menyebabkan struktur representasi JSK lebih kaya secara geometris dibandingkan dengan representasi PLB. Karena alasan tersebut, *encoder* pada penelitian ini dikembangkan sebagai JSK.

Implementasi Semua model dilatih sebanyak 200 iterasi. Pemilihan nilai tersebut berdasarkan pengamatan yang dilakukan terhadap laju konvergensi dari setiap model. Masing-masing model dioptimisasi menggunakan *stochastic gradient descent* dengan *adaptive moment* (adam) sebagai optimator dan *learning rate* sebesar 1e-4. Untuk setiap iterasi, *dataset* dibagi menjadi beberapa *mini-batch* yang dapat menampung 256 data.

Operasi transformasi pada citra dipilih sedemikian mengikuti penelitian yang dilakukan terhadap AVT (Qi et al., 2019), yaitu menggunakan transformasi proyektif. Operasi transformasi tersebut terdiri dari 3 operasi transformasi. Operasi yang pertama berupa translasi pada keempat sudut citra secara vertikal dan horizontal dengan rentang nilai $[\pm 1.25]$ kali dari ukuran lebar dan tinggi citra. Operasi kedua berupa operasi *scaling* dengan rentang faktor $[0.8, 1.2]$ dari ukuran citra. Operasi ketiga berupa rotasi yang nilainya dipilih dari himpunan diskret $\{0°, 90°, 180°, 270°\}$. Gambar 4.1 menunjukkan sampel citra awal dan sampel citra yang sudah ditransformasi.

Gambar 4.1: Sampel pada baris pertama merupakan citra awal yang belum ditransformasi. Sampel pada baris kedua didapat dengan menerapkan transformasi proyektif terhadap sampel pada baris pertama. Komposisi transformasi dipilih secara acak dalam rentang yang telah ditentukan sebelumnya.

Semua model representasi diimplementasikan sebagai suatu program Python (versi 3.7) menggunakan *framework* PyTorch Lightning (versi 1.9). Selanjutnya, program yang sudah diimplementasi dijalankan menggunakan bantuan 1 GPU Tesla V-100 milik Tokopedia-UI AI Center. Model representasi TEVBA, *predictive-transformation* dan model *baseline* AVT menghabiskan waktu pemelajaran kurang lebih selama 3 jam. Sementara itu, model representasi TEVInfoNCE *concatenated* dan TEVInfoNCE *separated* menghabiskan waktu pemelajaran kurang lebih selama 9 jam. Kedua model tersebut membutuhkan waktu yang lebih lama dikarenakan proses pemelajaran yang melibatkan pasangan sampel negatif sebagai faktor normalisasi.

4.1.1.2 Evaluasi

Pada saat evaluasi, TEV dan AVT dijadikan sebagai pengekstraksi fitur terhadap citra. Ekstraksi fitur diawali dengan melewatkan citra melalui *encoder*. Berdasarkan luaran *encoder*, dibangkitkan representasi/fitur melalui *reparameterization trick*. Fitur yang dibangkitkan menjadi masukan bagi *classifier* yang digunakan pada tugas klasifikasi. Perlu ditekankan bahwa model representasi tidak lagi dioptimisasi pada saat evaluasi. Pada penelitian ini, digunakan 3 model *classifier*: PLB dan K-*nearest neighbor* (K-NN), dan *multinomial logistic regression*. Ketiga *classifier* tersebut memiliki karakteristik yang sangat berbeda sehingga dapat memberi hasil yang lebih komperhensif mengenai kemampuan model representasi sebagai pengekstraksi fitur. *Classifier* PLB dan *multinomial logistic regression* masing-masing merupakan model linear dan non-linear yang terparameterisasi oleh matriks bobot. Sementara itu, K-NN merupakan model non-parametrik. Klasifikasi menggunakan *dataset* yang sama, namun tanpa menerapkan operasi transformasi. Selain itu, klasifikasi diuji dengan berbagai ukuran *training set*. Pada penelitian ini, dipilih masing-masing 50000, 5000, dan 500 data. Apabila model representasi berhasil menangkap struktur citra secara keseluruhan dan diasumsikan bahwa pemelajaran *classifier* ideal (tidak mengalami *overfitting*), maka model representasi dapat bekerja dengan cukup baik meskipun jumlah data yang terlibat sedikit.

Model *classifier* yang pertama terdiri atas 3 lapisan PLB. Dua lapisan pertama berukuran 2048, yang masing-masing diikuti oleh *batch normalization* dan fungsi non-linear ReLU. Lapisan terakhir berupa PLB berukuran 10, merepresentasikan jumlah kelas pada *dataset* CIFAR-10. Fungsi *cross entropy* dipilih sebagai fungsi objektif untuk melatih *classifier*. *Classifier* PLB dilatih sebanyak 200 iterasi menggunakan SGD, menggunakan optimator *adaptive momentum*. Pemilihan iterasi berdasarkan observasi yang dilakukan terhadap laju konvergensi *classifier* PLB. Selain itu, *classifier* PLB menggunakan 10% data *training set* untuk validasi. Sebagai contoh, untuk data berukuran 5000, maka 4500 data dipergunakan untuk melatih *classifier*, sementara 500 data sisanya dipergunakan untuk melakukan validasi.

Classifier K-NN menetapkan kelas dari suatu *query* dengan terlebih dahulu memilih *K*

data dari *training set* berdasarkan Euclidean *distance* yang paling rendah. Kelas dari *query* ditentukan berdasarkan mayoritas kelas dari K data yang terpilih. Pada penelitian ini dipilih $K = 5$. *Classifier* K-NN disebut juga sebagai model non-parameterik karena tidak terdapat parameter yang mengikatnya.

Classifier multinomial logistic regression secara sederhana merupakan suatu model *linear regression* dengan tambahan fungsi *softmax*. Fungsi *softmax* berperan memodelkan probabilitas dari setiap kelas sehingga model regresi dapat melakukan klasifikasi. *Multinomial logistic regression* merupakan generalisasi dari model *logistic regression* yang hanya mampu menangani tugas klasifikasi biner. Pada penelitian ini, *multinomial logistic regression* mengoptimisasi *cross entropy* sebagai fungsi objektif sebanyak 100 iterasi. Permasalahan optimisasi diselesaikan melalui metode *stochastic average gradient descent* (SAG) (Schmidt et al., 2017). *Stochastic average gradient descent* dipilih karena mampu bekerja dengan optimal untuk data berdimensi tinggi dan berjumlah banyak. Selain itu, *multinomial logistic regression* ini diregularisasi oleh l^2 *norm*.

Classifier yang telah dilatih diminta untuk memprediksi 10000 data yang sebelumnya belum pernah digunakan selama pelatihan. Metrik yang digunakan untuk evaluasi adalah nilai galat, mengikuti penelitian AVT. Metrik ini tepat untuk digunakan mengingat *dataset* yang digunakan merupakan *dataset* benchmark yang dijamin memiliki proporsi data per kelas yang seimbang. Untuk kasus *dataset* yang tidak seimbang, metrik *confusion matrix* dapat mendeskripsikan hasil klasifikasi dengan lebih baik. Selanjutnya, *encoder* TEV merupakan model probabilistik karena luarannya merupakan parameter distribusi. Karena itu, performa *classifier* diukur berdasarkan rata-rata nilai galat dari 5 hasil prediksi terhadap *test set*. Nilai galat merupakan perbandingan jumlah data yang mengalami misklasifikasi terhadap jumlah semua data yang diuji. Informasi mengenai rata-rata nilai galat direpresentasikan oleh *mean* dan standar deviasi. Misalkan pada salah satu hasil prediksi terdapat 50 data yang dimisklasifikasi oleh *classifier*, maka nilai galat *classifier* pada prediksi tersebut adalah $\frac{50}{10000} = 0.005$. Rata-rata nilai galat yang rendah memberi indikasi bahwa *classifier* mampu membedakan data menurut kelasnya. Hal ini secara tidak langsung berimplikasi terhadap kesuksesan pengekstraksi fitur yang mampu memberikan representasi yang bermakna sebagai masukan terhadap *classifier*. Sebagai pengingat, nilai nilai galat yang ideal adalah 0.0, artinya *classifier* berhasil memprediksi seluruh data dengan tepat. Hasil klasifikasi model TEV kemudian dibandingkan dengan AVT yang merupakan *baseline*. Selain itu, *predictive-transformation* juga dievaluasi melalui tugas klasifikasi untuk membuktikan suatu hipotesis. Secara intuitif, *predictive-transformation* seharusnya tidak menghasilkan performa klasifikasi yang lebih baik dibandingkan AVT dan TEV mengingat *predictive-transformation* hanya mampu menangani representasi citra hasil transformasi.

4.1.2 STL-10

Dataset STL-10 didesain untuk keperluan pemelajaran tidak terawasi. Terdapat 100000 citra RGB tanpa label berukuran 96×96. Semua model representasi yang diajukan beserta dengan *baseline* akan dilatih menggunakan data tersebut. Selain itu, terdapat 5000 dan 8000 citra yang sudah dianotasi, masing-masing dipergunakan untuk melatih dan mengevaluasi *classifier*. Secara keseluruhan, terdapat 10 kelas/label pada *dataset* STL-10 Coates et al. (2011).

4.1.2.1 Arsitektur dan Implementasi

Arsitektur Pada eksperimen ini, *encoder* dari setiap model yang diajukan beserta dengan *baseline* mengadopsi arsitektur Alexnet (Krizhevsky et al., 2012). *Encoder* terdiri dari beberapa blok yang masing-masing terdiri atas JSK, *batch normalization*, fungsi non-linear ReLU, dan jaringan *pooling*. Dimensionalitas representasi $\mathbf{z}$ dan $\hat{\mathbf{z}}$ sama dengan eksperimen sebelumnya, yaitu 512. *Decoder* maupun fungsi estimator diimplementasikan dengan PLB yang berjumlah tiga lapis. Arsitektur lengkap dari model representasi yang digunakan pada eksperimen ini dapat dilihat pada **Appendix B.2**.

Implementasi Pemilihan parameter pada eksperimen ini mengadopsi eksperimen sebelumnya. Agar dapat menggunakan operasi transformasi yang sama dengan eksperimen sebelumnya, setiap citra diatur ulang ukurannya menjadi 32×32. Selanjutnya, ukuran *mini-batch* diubah menjadi 512 seiring dengan bertambahnya jumlah *dataset*. Sama seperti eksperimen sebelumnya, model representasi diimplementasikan dalam bentuk program Python menggunakan *framework* PyTorch. Program ini juga dijalankan dengan 1 GPU Tesla V-100. Penambahan ukuran *mini-batch* pada eksperimen ini menyebabkan waktu yang dibutuhkan untuk pemelajaran relatif sama dengan eksperimen sebelumnya.

4.1.2.2 Evaluasi

Evaluasi pada eksperimen ini masih berdasarkan tugas klasifikasi citra. Eksperimen ini melibatkan *training set* berupa citra teranotasi yang berjumlah 5000. Karena jumlah data yang teranotasi terbatas, evaluasi klasifikasi hanya melibatkan skenario dengan jumlah data tersebut. Skenario ini berbeda dengan eksperimen menggunakan *dataset* CIFAR-10 yang terdiri atas 3 skenario klasifikasi dengan ukuran *training set* yang berbeda. Ketiga skenario tersebut didukung oleh ketersediaan data teranotasi CIFAR-10 yang berjumlah 60000. Sama seperti eksperimen sebelumnya, *classifier* yang digunakan adalah PLB, K-NN, dan *multinomial logistic regression*. *Classifier* PLB menggunakan sebanyak 500 data dari *training set* untuk proses validasi, sementara sisanya digunakan untuk pemelajaran. Arsitektur dan implementasi *classifier* mengikuti eksperimen sebelumnya. *Classifier* yang telah menyelesaikan pemelajaran diminta untuk melakukan klasifikasi terhadap 8000 data teranotasi yang belum pernah ditemui sebelumnya oleh *classifier*. Rata-rata nilai galat digunakan untuk mengukur peforma dari *classifier*.

4.2 Hasil Eksperimen

4.2.1 CIFAR-10

Tabel 4.1 menunjukkan rata-rata nilai galat *classifier* PLB dari masing-masing model pada tugas klasifikasi citra CIFAR-10. Model TEVBA memberi rata-rata nilai galat paling rendah untuk klasifikasi menggunakan *training set* sebanyak 50000 data. Sementara itu, TEVInfoNCE *concatenated* mencatatkan rata-rata nilai galat tertinggi. Untuk klasifikasi menggunakan *training set* sebanyak 5000 data, rata-rata nilai galat terendah dicapai oleh TEVInfoNCE *separated* dan rata-rata nilai galat tertinggi dicapai oleh TEVInfoNCE *concatenated*. Untuk kasus terakhir (*training set* sebanyak 500 data), TEVInfoNCE *concatenated* meraih rata-rata nilai galat paling rendah, sementara AVT mencatat rata-rata nilai galat tertinggi.

Tabel 4.1: Rata-rata nilai galat *classifier* PLB dari setiap model representasi dengan berbagai ukuran data untuk *dataset* CIFAR-10.

Model	50K	5K	0.5K
AVT	0.147 ± 0.0001	0.355 ± 0.0018	0.797 ± 0.0021
Predictive transformation	0.142 ± 0.0006	0.383 ± 0.0028	0.655 ± 0.0036
TEVBA	$\mathbf{0.140 \pm 0.0000}$	0.354 ± 0.0001	0.508 ± 0.0006
TEVInfoNCE (sep.)	0.148 ± 0.0005	$\mathbf{0.310 \pm 0.0015}$	0.550 ± 0.0015
TEVInfoNCE (concat.)	0.152 ± 0.0000	0.362 ± 0.0016	$\mathbf{0.476 \pm 0.0006}$

Tabel 4.2 menunjukkan rata-rata nilai galat classifier K-NN dari masing-masing model pada tugas klasifikasi citra CIFAR-10. Hasil eksperimen menunjukkan TEVBA

Tabel 4.2: Rata-rata nilai galat *classifier* K-NN dari setiap model representasi dengan berbagai ukuran data untuk *dataset* CIFAR-10.

Model	50K	5K	0.5K
AVT	0.693 ± 0.050	0.743 ± 0.006	0.803 ± 0.009
Predictive transformation	0.527 ± 0.006	0.596 ± 0.002	0.69 ± 0.011
TEVBA	$\mathbf{0.396 \pm 0.003}$	$\mathbf{0.488 \pm 0.003}$	$\mathbf{0.578 \pm 0.008}$
TEVInfoNCE (sep.)	0.429 ± 0.030	0.508 ± 0.004	0.583 ± 0.004
TEVInfoNCE (concatenated)	0.448 ± 0.003	0.519 ± 0.001	0.596 ± 0.006

mengungguli model lainnya untuk setiap ukuran *dataset*. Model representasi TEVInfoNCE *separated* dan *concatenated* masing-masing mencatatkan posisi kedua dan ketiga sebagai model dengan rata-rata nilai galat terendah untuk setiap ukuran *training set*. Sementara itu, AVT mencatatkan rata-rata nilai galat tertinggi untuk semua ukuran *training set*.

Kemudian, Tabel 4.3 menunjukkan rata-rata nilai galat *classifier multinomial logistic regression* dari masing-masing model pada tugas klasifikasi citra CIFAR-10. Berdasarkan hasil eksperimen, TEVBA konsisten menjadi model dengan rata-rata nilai galat terendah untuk setiap ukuran *training set*. Selanjutnya, TEVInfoNCE *concatenated* dan *separated*

Tabel 4.3: Rata-rata nilai galat *classifier logistic regression* dari setiap model representasi dengan berbagai ukuran data untuk *dataset* CIFAR-10.

Model	50K	5K	0.5K
AVT	0.622 ± 0.0023	0.759 ± 0.0023	0.840 ± 0.0029
Predictive transformation	0.539 ± 0.0025	0.698 ± 0.0018	0.793 ± 0.0046
TEVBA	$\mathbf{0.391 \pm 0.0005}$	$\mathbf{0.457 \pm 0.0003}$	$\mathbf{0.603 \pm 0.0003}$
TEVInfoNCE (sep.)	0.439 ± 0.0010	0.550 ± 0.0014	0.681 ± 0.0012
TEVInfoNCE (concatenated)	0.423 ± 0.0015	0.523 ± 0.0011	0.656 ± 0.0028

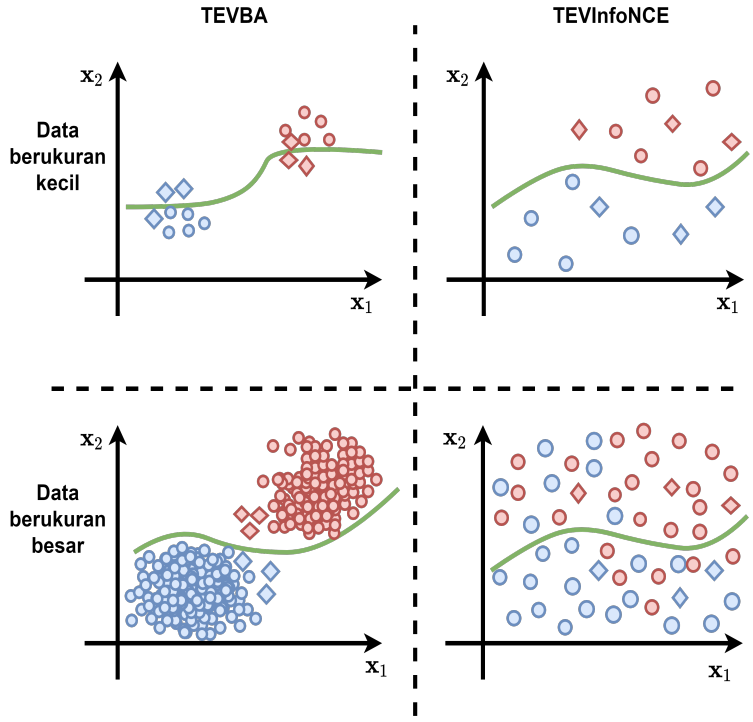
masing-masing menempati posisi kedua dan ketiga sebagai model dengan rata-rata nilai galat terendah. Sementara itu, AVT mencatatkan rata-rata nilai galat tertinggi untuk semua ukuran *training set*.

Secara keseluruhan, rata-rata nilai galat mengalami peningkatan seiring dengan berkurangnya ukuran *training set*. Berdasarkan Tabel 4.1 kolom 5K, model representasi mampu memberi hasil yang cukup baik meskipun hanya melibatkan sejumlah kecil *training set*. Hal ini ditunjukkan oleh rata-rata nilai galat dari semua model yang diujikan kurang dari 0.4. Berdasarkan data tersebut, dapat disimpulkan bahwa *classifier* rata-rata berhasil memprediksi lebih dari setengah data *test set* dengan benar. Namun, hasil klasifikasi untuk *training set* berukuran 500 menunjukkan hasil yang kurang menjanjikan karena rata-rata nilai galat untuk semua model lebih dari 0.5. Sementara itu, hasil klasifikasi menggunakan *classifier* K-NN dan *multinomial logistic regression* menunjukkan hasil yang kurang menjanjikan bahkan ketika menggunakan *training set* dengan ukuran 5000. Nilai tersebut memberi indikasi awal bahwa *classifier* yang digunakan berperan penting terhadap hasil klasifikasi.

Tabel 4.2 menunjukkan bahwa model representasi yang memberi rata-rata nilai galat terendah berbeda untuk setiap ukuran *training set*. Penjelasan mengenai hasil tersebut berkaitan dengan kecenderungan PLB mengalami *overfitting* dan karakteristik model representasi TEVBA dan TEVInfoNCE. *Overfitting* secara umum merupakan fenomena dimana hipotesis dari suatu model pemelajaran terlalu sesuai dengan data pada *training set*. Akibatnya, model pemelajaran memiliki performa yang baik pada saat proses pemelajaran, namun memiliki performa yang buruk pada saat evaluasi. Adanya *overfitting* mengindikasikan bahwa model pemelajaran gagal menggeneralisasi data. *Overfitting* umumnya disebabkan oleh jumlah data yang terlalu sedikit ketika proses pemelajaran. Hal ini sesuai dengan hasil yang didapat pada ketiga tabel yang menunjukkan penurunan performa seiring dengan berkurangnya jumlah data pada *training set*. *Classifier* PLB yang dilengkapi dengan fungsi non-linear di setiap lapisan menyebabkan PLB cenderung lebih mudah menyesuaikan data karena mampu merepresentasikan data dengan lebih kompleks.

Selanjutnya, TEVInfoNCE yang berbasis pemelajaran konstrastif cenderung menghasilkan representasi yang lebih *sparse* dibandingkan dengan TEVBA. Hal tersebut dikarenakan pemelajaran konstrastif secara intuitif mendorong sampel negatif menjauh

dari sampel positif. Untuk data dalam jumlah yang sedikit, *sparsity* dapat mengurangi efek *overfitting* karena regional representasi data untuk setiap kelas menjadi lebih luas. Ketika jumlah data besar, representasi yang dihasilkan TEVInfoNCE menyulitkan PLB dalam menemukan hipotesis yang dapat menyesuaikan data yang menyebar. Sementara itu, model TEVBA cenderung menghasilkan representasi yang terkonsentrasi pada regional ruang representasi tertentu terutama untuk citra yang memiliki similaritas yang tinggi. Hal tersebut dikarenakan fungsi objektif TEVBA tidak melibatkan sampel pasangan negatif sehingga tidak ada sinyal pemelajaran yang mengendalikan jarak geometris antar representasi. Untuk jumlah data yang sedikit, TEVBA menyebabkan PLB lebih mudah mengalami *overfitting*. Hal tersebut disebabkan hipotesis yang dihasilkan PLB cenderung lebih mudah untuk menyesuaikan data yang terkonsentrasi dalam jumlah yang sedikit. Sebaliknya, untuk data berukuran besar, PLB dapat lebih mudah menemukan hipotesis yang dapat menyesuaikan data dengan resiko *overfitting* yang lebih rendah. Gambar 4.2 menunjukkan ilustrasi klasifikasi data yang diekstraksi oleh TEVBA dan TEVInfoNCE menggunakan *classifier* PLB. Kolom pertama menunjukkan hasil klasifikasi untuk model TEVBA, sementara kolom kedua menunjukkan hasil klasifikasi untuk model TEVInfoNCE. Baris pertama menunjukkan hasil klasifikasi untuk skenario data berukuran kecil, sementara baris kedua menunjukkan hasil klasifikasi untuk skenario data berukuran besar. Ilustrasi ini melibatkan data yang terbagi ke dalam 2 kelas, yaitu kelas merah dan biru. *Training set* direpresentasikan oleh lingkaran, sementara *test set* direpresentasikan oleh bangun segi empat. Kurva berwarna hijau merepresentasikan hipotesis yang dihasilkan dari pemelajaran yang melibatkan *training set*.



Gambar 4.2: Ilustrasi klasifikasi data yang diekstraksi oleh TEVBA dan TEVInfoNCE menggunakan *classifier* PLB.

Berdasarkan Tabel 4.2 dan Tabel 4.3, hasil yang sama tidak berlaku untuk *classifier* K-NN dan *multinomial logistic regression*. Hal tersebut dikarenakan kompleksitas kedua *classifier* yang lebih rendah dibandingkan dengan PLB. Kompleksitas yang rendah cenderung menyebabkan *classifier* mengalami *underfitting*, yaitu *classifier* gagal menemukan hipotesis yang optimal selama pemelajaran berlangsung. Meskipun begitu, hasil klasifikasi menggunakan kedua *classifier* tersebut menunjukkan TEVBA menghasilkan rata-rata nilai galat terendah secara konsisten untuk setiap skenario. Klasifikasi menggunakan K-NN secara alamiah menguntungkan model representasi TEVBA. *Classifier* K-NN memprediksi kelas suatu *query* berdasarkan kelas mayoritas dari data yang berdekatan secara geometris dengan *query* tersebut. Representasi TEVBA yang cenderung terkonsentrasi untuk data yang memiliki similaritas tinggi akan memberi hasil klasifikasi yang lebih akurat dibandingkan dengan TEVInfoNCE yang menghasilkan representasi yang tersebar. *Classifier multinomial logistic regression* yang merupakan model linear juga cenderung lebih mudah menemukan hipotesis yang ideal untuk representasi yang terkonsentrasi pada suatu regional dibandingkan dengan representasi yang tersebar. Berdasarkan beberapa penjelasan sebelumnya, telah ditunjukkan bahwa jenis *classifier* berpengaruh terhadap hasil klasifikasi dari model representasi.

4.2.2 STL-10

Tabel 4.4 menunjukkan rata-rata nilai galat *classifier* PLB, K-NN, dan *multinomial logistic regression* dari masing-masing model representasi pada tugas klasifikasi citra STL-10. Hasil eksperimen menunjukkan TEVInfoNCE *concatenated* menjadi model representasi dengan rata-rata nilai galat terendah untuk semua *classifier*. Selanjutnya, TEVInfoNCE *separated* dan TEVBA masing-masing menempati urutan kedua dan ketiga sebagai model representasi dengan rata-rata nilai galat terendah untuk semua *classifier*. Rata-rata nilai galat tertinggi diraih oleh AVT untuk semua *classifier*.

Tabel 4.4: Rata-rata nilai galat tugas klasifikasi pada *dataset* STL-10.

Model	PLB	K-NN	Logistic regression
AVT	0.522 ± 0.0000	0.707 ± 0.004	0.632 ± 0.0000
Predictive transformation	0.492 ± 0.0009	0.711 ± 0.002	0.544 ± 0.0012
TEVBA	0.460 ± 0.0007	0.607 ± 0.004	0.54 ± 0.0002
TEVInfoNCE (sep.)	0.365 ± 0.0000	0.475 ± 0.005	0.477 ± 0.0000
TEVInfoNCE (concat.)	$\mathbf{0.363 \pm 0.0005}$	$\mathbf{0.473 \pm 0.003}$	$\mathbf{0.462 \pm 0.0003}$

Berdasarkan dua eksperimen yang dilakukan, model representasi AVT cenderung memberi rata-rata nilai galat yang lebih tinggi bila dibandingkan dengan model representasi yang diajukan (TEVBA dan TEVInfoNCE). Bahkan, *predictive-transformation* memberi rata-rata nilai galat yang lebih rendah dibandingkan AVT pada beberapa kasus. Hasil tersebut berbeda dengan hipotesis awal yang menduga bahwa AVT bekerja lebih baik dibandingkan dengan *predictive-transformation*. Hal ini dapat memberi indikasi bahwa menggabungkan representasi $\hat{\mathbf{z}}$ dan $\mathbf{z}$ secara bersamaan untuk menginferensi transformasi $\mathbf{t}$ dapat membatasi kemampuan *encoder* untuk mengekstraksi representasi $\hat{\mathbf{z}}$ dan $\mathbf{z}$ yang esensial. Di sisi lain, model representasi TEV mempelajari representasi $\mathbf{z}$ dan $\hat{\mathbf{z}}$ secara terpisah (dengan bantuan *predictive-transformation*). Pendekatan ini memungkinkan *predictive-transformation* dan TEV untuk mengeksploitasi struktur representasi $\hat{\mathbf{z}}$ dan $\mathbf{z}$ secara maksimal tanpa kehilangan sifat transformasi-ekuivarian. Visualisasi t-*distributed stochastic neighbor embedding* (t-SNE) juga dilakukan untuk mendapatkan interpretasi mengenai representasi yang dihasilkan oleh masing-masing model. Hasil visualisasi tersebut memberi interpretasi bahwa TEV mampu menghasilkan representasi yang lebih baik. Hasil visualisasi t-SNE untuk eksperimen CIFAR-10 dan STL-10 masing-masing dapat dilihat pada **Appendix C.1** dan **Appendix C.2**.

Kekurangan utama dari model yang diajukan terletak pada waktu komputasi yang lama. Hal ini disebabkan TEV membutuhkan *predictive-transformation* untuk memelajari struktur representasi $\mathbf{z}$. Akibatnya, proses pemelajaran TEV bertambah menjadi dua tahap (pemelajaran terhadap $\mathbf{z}$ dan pemelajaran terhadap $\hat{\mathbf{z}}$). Masing-masing tahap memakan waktu yang sama untuk menyelesaikan proses pemelajaran. Karenanya, rata-rata waktu komputasi TEV untuk proses pemelajaran bernilai dua kali lipat waktu komputasi AVT (dengan asumsi kompleksitas arsitektur AVT dan TEV sama). Sebagai pembanding,

pemelajaran pada *predictive-transformation*, dan TEV (TEVBA dan TEVInfoNCE) menghabiskan waktu 3 jam untuk eksperimen CIFAR-10 dan STL-10. Hal ini berarti TEV membutuhkan total 6 jam untuk menyelesaikan proses pemelajaran pada kedua *dataset*. Walaupun begitu, terdapat keuntungan menggunakan model representasi TEV, yaitu model ini bersifat modular. Telah diketahui bahwa *predictive-transformation* berperan menemukan struktur representasi $\mathbf{z}$ yang nantinya menjadi bias induktif bagi TEV. TEV dapat menggunakan model representasi lain untuk menggantikan *predictive-transformation*, seperti *autoencoder* atau model pemelajaran terawasi sendiri lainnya. Bias induktif yang digunakan tentu akan mempengaruhi struktur representasi yang dihasilkan.

BAB 5
KESIMPULAN

Kemampuan JSK untuk mengekstraksi representasi yang esensial dari suatu citra didukung oleh sifat transformasi-ekuivarian terhadap translasi. Sifat transformasi-ekuivarian untuk transformasi secara general menghasilkan suatu representasi yang turut berubah seiring dengan penerapan transformasi pada citra asal. Namun, sifat transformasi-ekuivarian pada JSK hanya terbatas pada transformasi berupa translasi. *State-of-the-art* membangun suatu model representasi tidak terawasi yang dinamakan *autoencoding transformation* (AET). Model representasi ini kemudian dikembangkan dari perspektif teori informasi, yang kemudian disebut sebagai *autoencoding variational transformation* (AVT).

Penelitian ini berusaha menginvestigasi fungsi objektif alternatif untuk membangun AVT. Berdasarkan percobaan awal, model alternatif membutuhkan suatu bias induktif sehingga proses pemelajaran dapat berhasil. Dengan alasan tersebut, penelitian ini memperkenalkan suatu model pemelajaran terawasi sendiri yang dinamakan *predictive-transformation* sebagai bias induktif bagi model alternatif AVT. Model representasi alternatif ini kemudian dinamakan sebagai transformasi-ekuivarian variasional (TEV). Fungsi objektif dari model TEV melibatkan metode estimasi informasi timbal balik (ITB). Pada penelitian ini dipilih 2 metode estimasi yaitu: metode estimasi batas bawah Barber-Agakov (BA) dan estimasi batas bawah *information noise contrastive estimation* (InfoNCE). Model TEV yang diestimasi dengan metode Barber-Agakov dinamakan TEV Barber-Agakov (TEVBA), sementara TEV yang diestimasi dengan metode InfoNCE dinamakan TEVInfoNCE.

Model representasi TEV dievaluasi melalui serangkaian tugas klasifikasi citra. Hasil menunjukkan TEVInfoNCE mengungguli model *baseline* untuk klasifikasi yang hanya melibatkan data dalam jumlah yang sedikit. Sementara itu, TEVBA memberi hasil yang paling baik untuk tugas klasifikasi yang melibatkan data dalam jumlah yang banyak. Hasil visualisasi t-SNE juga memberi indikasi bahwa model TEV menghasilkan struktur representasi yang lebih baik dibandingkan model AVT. Kelebihan model TEV terletak pada sifatnya yang modular sehingga *predictive-transformation* dapat digantikan oleh model representasi lainnya. Sementara itu, kelemahan utama dari model TEV terletak pada waktu dan sumber komputasi yang lebih boros dibandingkan dengan AVT. Semua temuan dan hasil penelitian ini telah diterima sebagai prosiding dalam konferensi *International Conference on Pattern Recognition Applications and Methods* (ICPRAM) 2022. Penelitian ini nantinya akan dipresentasikan pada konferensi tersebut pada tanggal 3-5 Februari 2022.

Hasil yang didapatkan pada penelitian ini membuka peluang bagi penelitian mendatang, diantaranya:

- Evaluasi terhadap model representasi dapat dicobakan pada tugas pemelajaran lainnya seperti deteksi objek maupun segementasi.
- Model representasi TEV dapat dikembangkan untuk memelajari struktur representasi data *sequential* seperti teks maupun dialog. Oord et al. (2018) berhasil membangun model representasi berbasis ITB untuk data *sequential* menggunakan estimasi InfoNCE. Hanya saja, model representasi tersebut tidak bersifat transformasi-ekuivarian.
- Fungsi objektif TEV dapat pula digunakan untuk mengembangkan model pemelajaran semi terawasi yang bersifat transformasi-ekuivarian. Sebagai perbandingan, penelitian yang dilakukan Qi et al. (2020) berhasil mengembangkan model pemelajaran semi terawasi yang bersifat transformasi-ekuivarian menggunakan fungsi objektif AVT.

DAFTAR REFERENSI

Agakov, D. B. F. (2004). The im algorithm: a variational approach to information maximization. In *Advances in neural information processing systems*. PMLR.

Alemi, A. A., Fischer, I., Dillon, J. V., and Murphy, K. (2017). Deep variational information bottleneck. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *In International conference on machine learning*. PMLR.

Belghazi, M. I., Baratin, A., Rajeshwar, S., Ozair, S., Bengio, Y., Courville, A., and Hjelm, D. (2018). Mutual information neural estimation. In *In International Conference on Machine Learning*. PMLR.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. E. (2020). A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. (2016). Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In Lee, D. D., Sugiyama, M., von Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2172–2180.

Cheng, P., Hao, W., Dai, S., Liu, J., Gan, Z., and Carin, L. (2020). Club: A contrastive log-ratio upper bound of mutual information. In *In International Conference on Machine Learning*, pages 1779–1788. PMLR.

Coates, A., Ng, A. Y., and Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In Gordon, G. J., Dunson, D. B., and Dudík, M., editors,

Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011, volume 15 of *JMLR Proceedings*, pages 215–223. JMLR.org.

Cohen, T. and Welling, M. (2016). Group equivariant convolutional networks. In Balcan, M. and Weinberger, K. Q., editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2990–2999. JMLR.org.

Cohen, T. S., Geiger, M., and Weiler, M. (2018). Intertwiners between induced representations (with applications to the theory of equivariant neural networks). *CoRR*, abs/1803.10743.

Cohen, T. S. and Welling, M. (2017). Steerable cnns. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Doersch, C., Gupta, A., and Efros, A. A. (2015). Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*. IEEE.

Dosovitskiy, A., Springenberg, J. T., Riedmiller, M., and Brox, T. (2014). Discriminative unsupervised feature learning with convolutional neural networks. In *NIPS*. PMLR.

Flexa, C., Gomes, W. C., Moreira, I., Alves, R., and Sales, C. (2021). Polygonal coordinate system: Visualizing high-dimensional data using geometric dr, and a deterministic version of t-sne. *Expert Syst. Appl.*, 175:114741.

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media.

Gidaris, S., Singh, P., and Komodakis, N. (2018). Unsupervised representation learning by predicting image rotations. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*. Cambridge: MIT press.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. (2020). Generative adversarial networks. *Commun. ACM*, 63(11):139–144.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Hinton, G. E., Krizhevsky, A., and Wang, S. D. (2011). Transforming auto-encoders. In Honkela, T., Duch, W., Girolami, M. A., and Kaski, S., editors, *Artificial Neural Networks and Machine Learning - ICANN 2011 - 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I*, volume 6791 of *Lecture Notes in Computer Science*, pages 44–51. Springer.

Hinton, G. E., Sabour, S., and Frosst, N. (2018). Matrix capsules with em routing. In *ICLR*.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In Bengio, Y. and LeCun, Y., editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Kristiadi, A. (2019). Predictive uncertainty quantification with compound density networks. Master's thesis.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Bartlett, P. L., Pereira, F. C. N., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114.

Lenssen, J. E., Fey, M., and Libuschewski, P. (2018). Group equivariant capsule networks. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8858–8867.

Ma, Z. and Collins, M. (2018). Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3698–3707. Association for Computational Linguistics.

MacKay, D. J. C. (2003). *Information theory, inference, and learning algorithms*. Cambridge University Press.

Makhzani, A., Shlens, J., Jaitly, N., and Goodfellow, I. J. (2015). Adversarial autoencoders. *CoRR*, abs/1511.05644.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *ICML*. PMLR.

Noroozi, M. and Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*. Springer.

Noroozi, M., Pirsiavash, H., and Favaro, P. (2017). Representation learning by learning to count. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5899–5907. IEEE Computer Society.

Oord, A. V. D., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv:1807.03748*.

Poole, B., Ozair, S., Van Den Oord, A., Alemi, A., and Tucker, G. (2019). On variational bounds of mutual information. pages 5171–5180.

Qi, G.-J., Zhang, L., Chen, C. W., and Tian, Q. (2019). Avt: Unsupervised learning of transformation equivariant representations by autoencoding variational transformations. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 8129–8138. IEEE.

Qi, G. J., Zhang, L., Hu, H., Edraki, M., Wang, J., and Hua, X. S. (2018). Global versus localized generative adversarial nets. In *In Proceedings of the IEEE conference on computer vision and pattern recognition*. IEEE.

Qi, G. J., Zhang, L., Lin, F., and Wang, X. (2020). Learning generalized transformation equivariant representations via autoencoding transformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Radford, A., Metz, L., and Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. In Bengio, Y. and LeCun, Y., editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. (2011). Contractive auto-encoders: Explicit invariance during feature extraction. In *ICML*. PMLR.

Sabour, S., Frosst, N., and Hinton, G. E. (2017). Dynamic routing between capsules. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3856–3866.

Schmidt, M., Roux, N. L., and Bach, F. R. (2017). Minimizing finite sums with the stochastic average gradient. *Math. Program.*, 162(1-2):83–112.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Song, J. and Ermon, S. (2020). Understanding the limitations of variational mutual information estimators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Steck, H. (2020). Autoencoders that don't overfit towards the identity. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9. IEEE Computer Society.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P. A. (2008). Extracting and composing robust features with denoising autoencoders. In *In Proceedings of the 25th international conference on Machine learning*. PMLR.

Wang, D. and Liu, Q. (2018). An optimization view on dynamic routing between capsules. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net.

Worrall, D. E. (2019). *Equivariance For Deep Learning And Retinal Imaging*. PhD thesis.

Zaheer, M. (2018). *diversity, interpretability, scalability*. PhD thesis.

Zhang, L., Qi, G. J., Wang, L., and Luo, J. (2019). Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2547–2555.

APPENDIX

APPENDIX A

A.1 Batas Bawah InfoNCE

Pembuktian ini untuk menunjukkan batas bawah InfoNCE bergantung pada jumlah sampel yang digunakan. Penurunan bukti berdasarkan Oord et al. (2018), dengan sedikit modifikasi. Pembuktian dilakukan hanya untuk kasus TVEInfoNCE *concatenated*.

Nilai ideal untuk $\exp(f_{\hat{\phi}}(\hat{\mathbf{z}}^{(i)}, \mathbf{z}^{(i)}))$ adalah $\frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}|\hat{\mathbf{z}}^{(i)})}{p(\hat{\mathbf{z}}^{(i)})}$. Pembuktian dilakukan dengan mensubstitusi bentuk tersebut ke dalam Pertidaksamaan 3.13. Untuk mempermudah notasi, $\mathop{\mathbb{E}}_{\prod_j p_{\theta,\hat{\theta}}(\mathbf{t}^{(j)},\hat{\mathbf{z}}^{(j)},\mathbf{z}^{(j)})}$ dituliskan sebagai $\mathbb{E}$.

$$
\begin{aligned}
\mathcal{L}_{\text{TVEInfoNCE}} &= \mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K}\log\frac{\frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}|\mathbf{z}^{(i)})}{p(\hat{\mathbf{z}}^{(i)})}}{\frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}|\mathbf{z}^{(i)})}{p(\hat{\mathbf{z}}^{(i)})}+\sum_j\frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(j)}|\mathbf{z}^{(i)})}{p(\hat{\mathbf{z}}^{(j)})}}\right] \\
&= -\mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K}\log\left[1+\frac{p(\hat{\mathbf{z}}^{(i)})}{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}|\mathbf{z}^{(i)})}\sum_j\frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}|\mathbf{z}^{(j)})}{p(\hat{\mathbf{z}}^{(i)})}\right]\right] \\
&\approx -\mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K}\log\left[1+\frac{p(\hat{\mathbf{z}}^{(i)})}{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}|\mathbf{z}^{(i)})}K\,\mathbb{E}_{\hat{\mathbf{z}}^{(j)}}\frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(j)}|\mathbf{z}^{(i)})}{p(\hat{\mathbf{z}}^{(j)})}\right]\right] \\
&= -\mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K}\log\left[1+\frac{p(\hat{\mathbf{z}}^{(i)})}{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}|\mathbf{z}^{(i)})}K\right]\right] \\
&\leq -\mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K}\log\left[\frac{p(\hat{\mathbf{z}}^{(i)})}{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}|\mathbf{z}^{(i)})}K\right]\right] \\
&= \frac{1}{K}\sum_{i=1}^{K}\left[\mathbb{E}_{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)},\mathbf{z}^{(i)})}\left[\frac{p_{\theta,\hat{\theta}}(\hat{\mathbf{z}}^{(i)}|\mathbf{z}^{(i)})}{p(\mathbf{z}^{(i)})}\right]-\log K\right] \\
&= I_{\theta,\hat{\theta}}(\mathbf{Z};\hat{\mathbf{Z}})-\log K
\end{aligned}
$$

Dengan demikian, didapat $I_{\theta,\hat{\theta}}(\mathbf{Z};\hat{\mathbf{Z}}) \geq \mathcal{L}_{\text{TVEInfoNCE}} + \log K$

APPENDIX B

B.1 Arsitektur Model : CIFAR-10

Tabel B.1 menunjukkan arsitektur yang digunakan untuk membangun *predictive-transformation*, AVT, TVEBA, dan TVEInfoNCE pada *dataset* CIFAR-10. Arsitektur *encoder* mengikuti *network-in-network*.

Tabel B.1: Arsitektur yang digunakan pada eksperimen CIFAR-10.

Encoder	**Decoder,** $f, g,$ **dan** h

`Basic Block(3, 192, 5)`
`Basic Block(192, 160, 1)`
`Basic Block(160, 96, 1)`
`Max Pool(3, 2, 1)`
	`Linear(512, 2048)`
	`ReLU`
	`Linear(2048, 512)`

`Basic Block(96, 192, 5)`
`Basic Block(192, 192)`
`Basic Block(192, 8)`
`Average Pool(3, 2, 1)`	

$\boldsymbol{\mu}_{\phi} \rightarrow$ `Linear(512, 512)` $\log \boldsymbol{\sigma}_{\phi} \rightarrow$ `Linear(512, 512)`	

`Basic Block(in, out, kernel)` merupakan suatu blok yang terdiri dari `Conv2D(in, out, kernel, stride=1, padding=(kernel - 1) // 2)` $\rightarrow$ `Batch Norm 2D` $\rightarrow$ `ReLU`. Parameter `in` menyatakan ukuran *channel* masukan, `out` menyatakan ukuran *channel* keluaran, dan `kernel` menyatakan ukuran kernel yang memiliki lebar dan tinggi yang sama.

B.2 Arsitektur Model : STL-10

Tabel B.2 menunjukkan arsitektur yang digunakan untuk membangun *predictive-transformation*, AVT, TVEBA, dan TVEInfoNCE pada *dataset* STL-10. Pada eksperimen ini, arsitektur *encoder* mengadopsi arsitektur *back bone* Alexnet. `Alex Block(in, out, kernel, stride, padding)` merupakan suatu blok yang terdiri dari `Conv2D(in, out, kernel, stride, padding)` $\rightarrow$ `ReLU`.

B.3 Arsitektur *Classifier*

Tabel B.3 menunjukkan arsitektur yang digunakan untuk membangun *classifier* pada kedua eksperimen. Arsitektur *classifier* hanya terdiri dari tiga MLP.

Tabel B.2: Arsitektur yang digunakan pada eksperimen STL-10.

Encoder	**Decoder,** $f, g,$ **dan** h

`Alex Block(3, 64, 11, 1, 2)`
`Max Pool(3, 2, 0)`
	`Linear(512, 2048)`
	`BatchNorm`
	`ReLU`

`Alex Block(64, 192, 5, 1, 2)`
`Max Pool(3, 2, 0)`
	`Linear(2048, 1024)`
	`BatchNorm`
	`ReLU`

`Alex Block(192, 384, 3, 1, 1)`	`Linear(1024, 512)`
`Alex Block(96, 192, 5)`	
`Alex Block(192, 192)` `Max Pool(3, 2, 0)`	
$\boldsymbol{\mu}_{\phi} \rightarrow$ `Linear(1024, 512)` $\log \boldsymbol{\sigma}_{\phi} \rightarrow$ `Linear(1024, 512)`	

Tabel B.3: Arsitektur yang digunakan untuk membangun *Classifier*

Classifier

`Linear(512, 2048)`
`BatchNorm`
`ReLU`

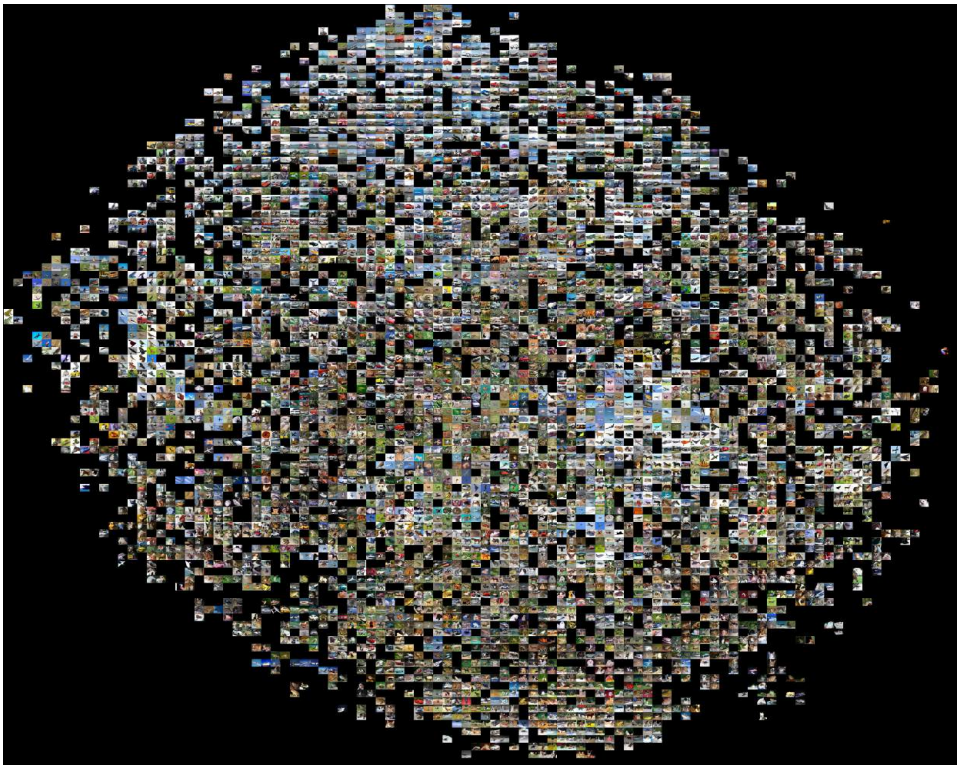
`Linear(2048, 2048)`
`BatchNorm`
`ReLU`

`Linear(2048, 10)`

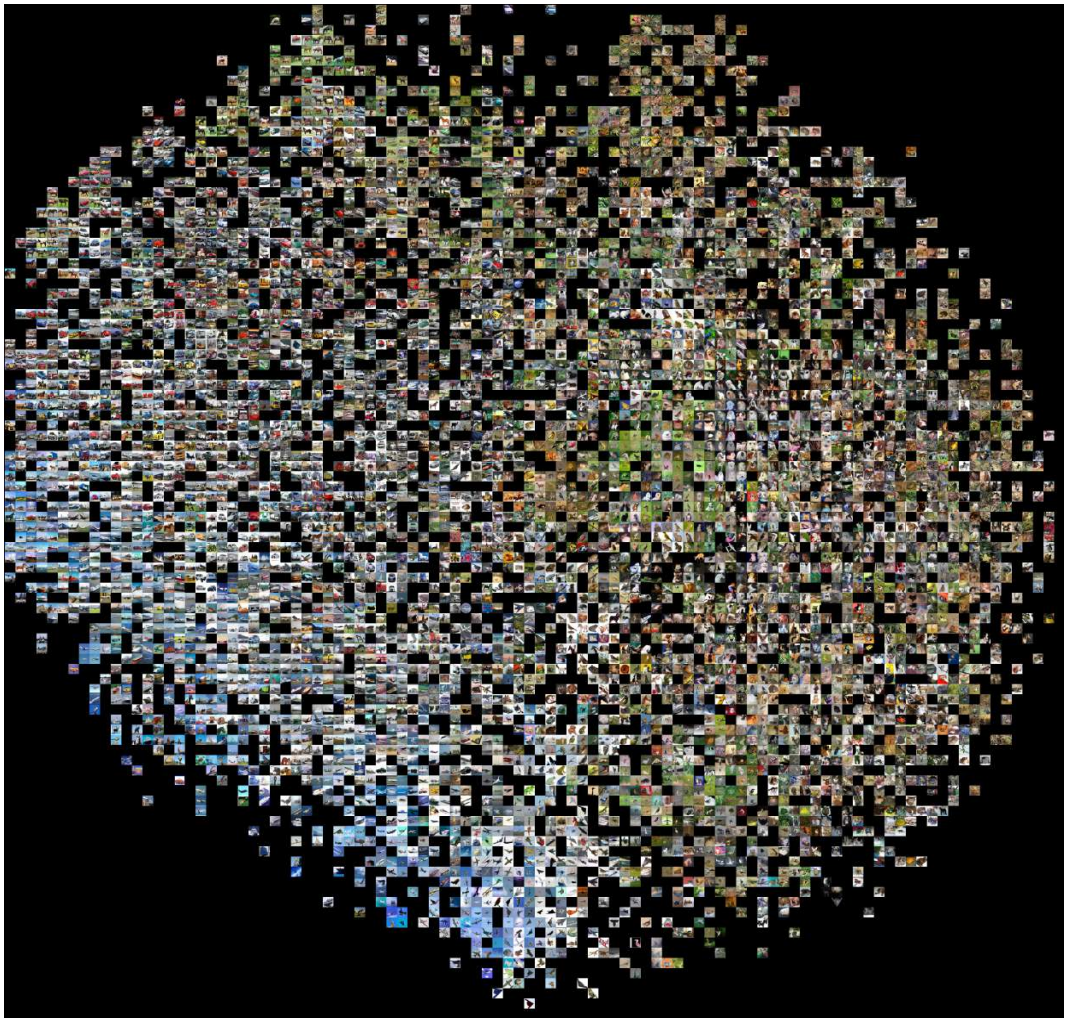
APPENDIX C

C.1 Visualisasi t-SNE CIFAR-10

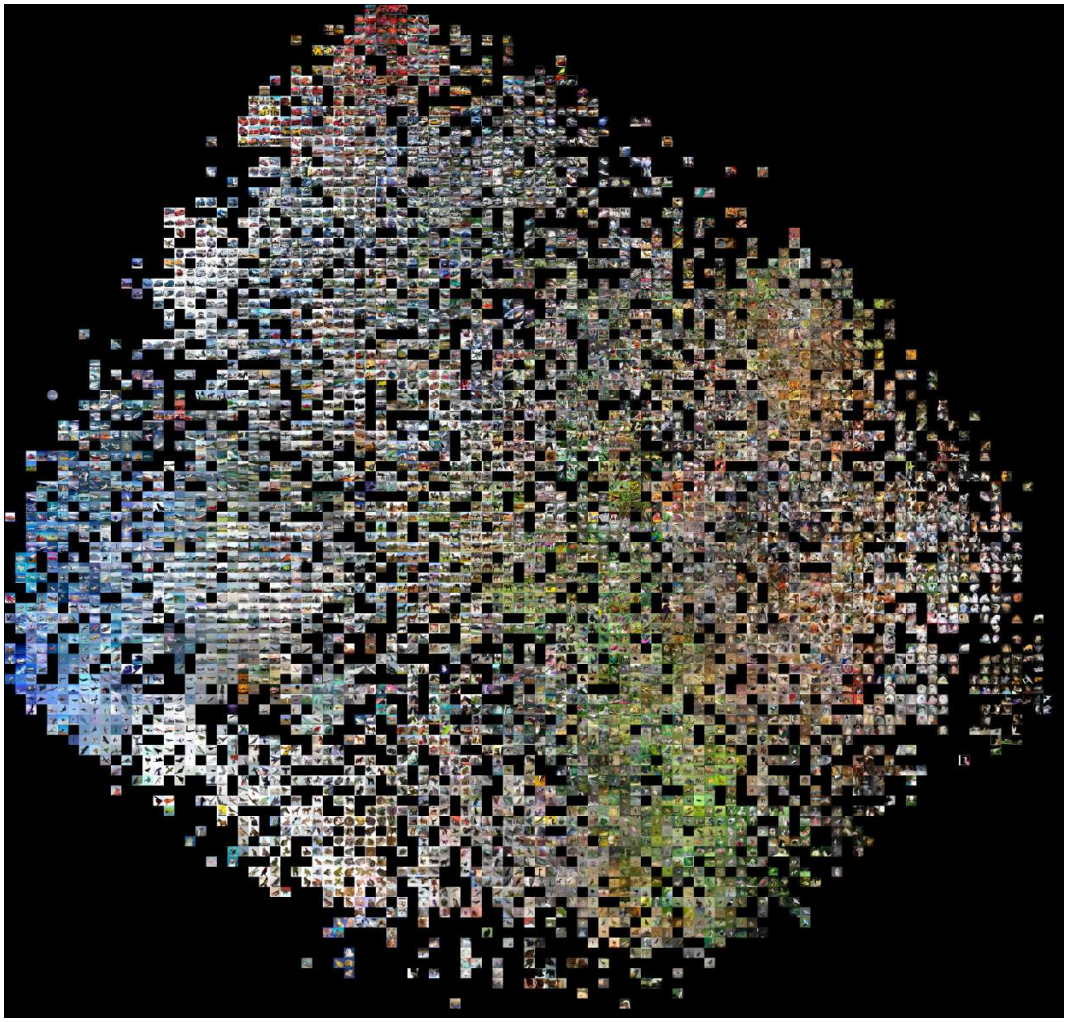
Bagian ini menunjukkan visualisasi dari *test set* CIFAR-10 melalui t-SNE (Flexa et al., 2021). Luaran dari t-SNE berupa vektor representasi yang terdiri atas dua komponen ($\mathbb{R}^2$). Visualisasi dilakukan dengan cara memetakan setiap citra berdasarkan koordinat t-SNE pada $\mathbb{R}^2$.



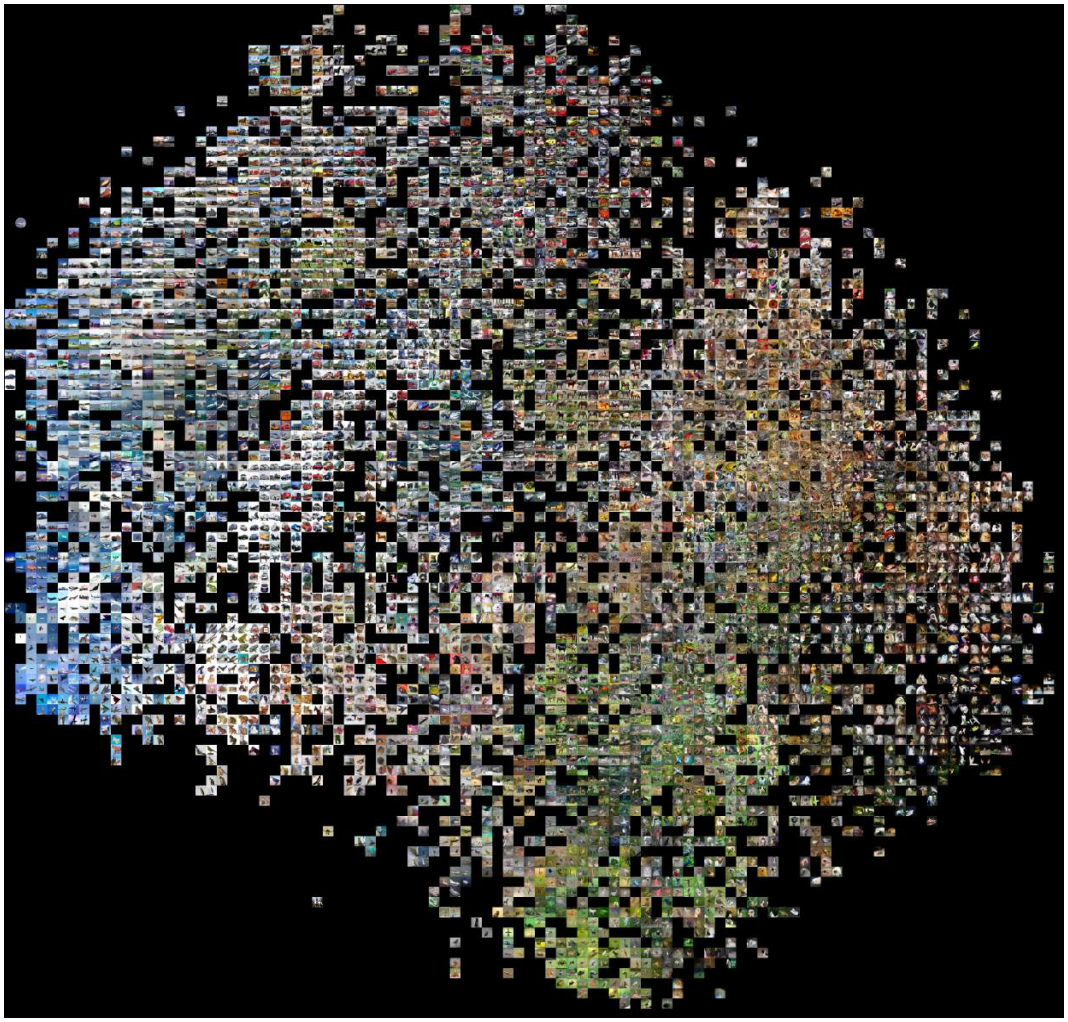
Gambar C.1: Visualisasi t-SNE AVT pada *dataset* CIFAR-10.



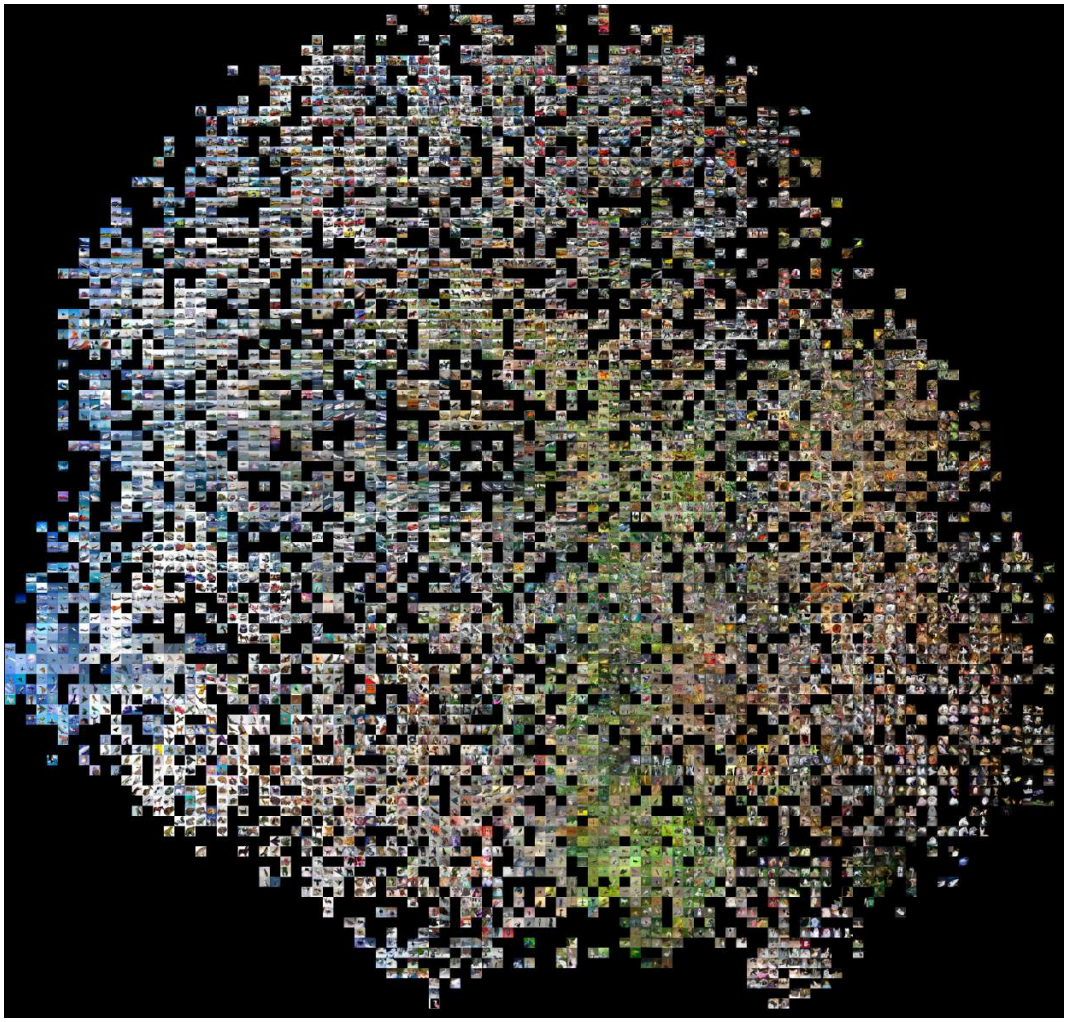
Gambar C.2: Visualisasi t-SNE *predictive-transformation* pada *dataset* CIFAR-10.



Gambar C.3: Visualisasi t-SNE TVEBA pada *dataset* CIFAR-10.



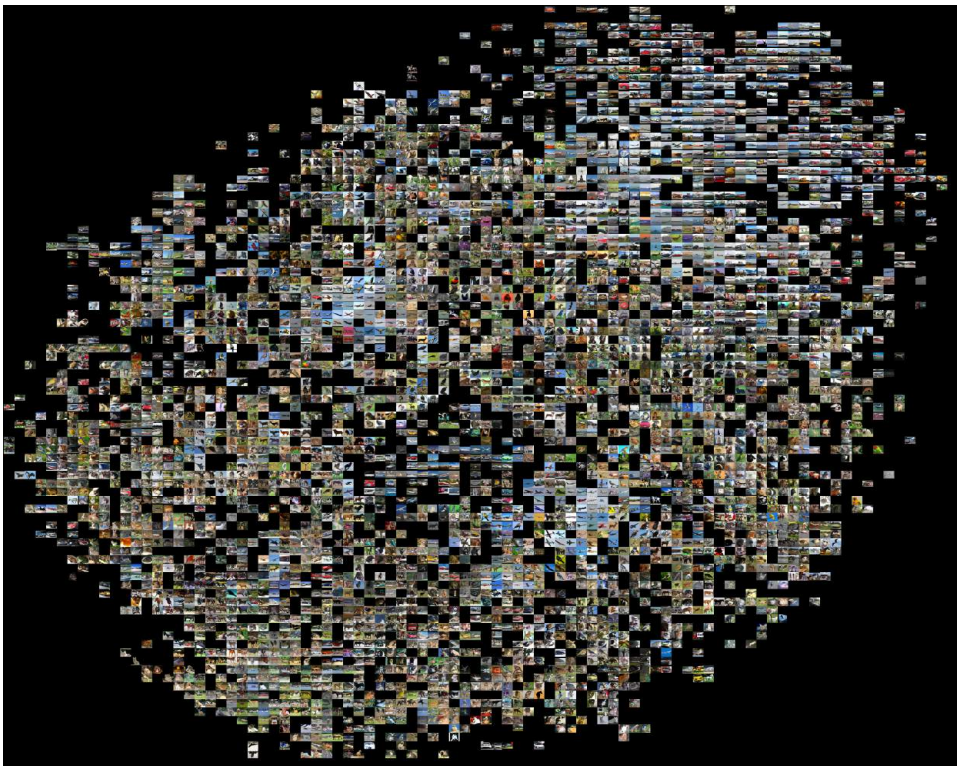
Gambar C.4: Visualisasi t-SNE TVEInfoNCE *separated* pada *dataset* CIFAR-10.



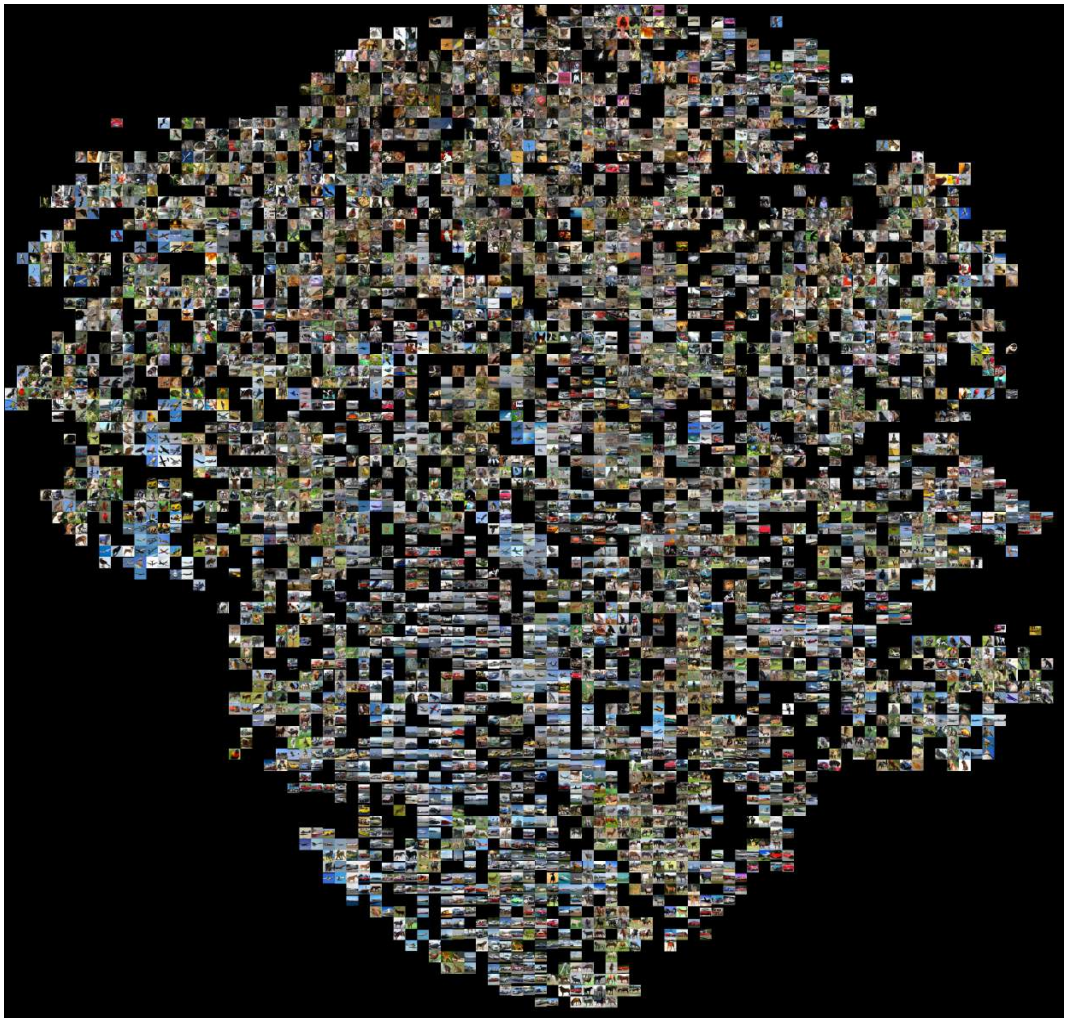
Gambar C.5: Visualisasi t-SNE TVEInfoNCE *concatenated* pada *dataset* CIFAR-10.

C.2 Visualisasi t-SNE STL-10

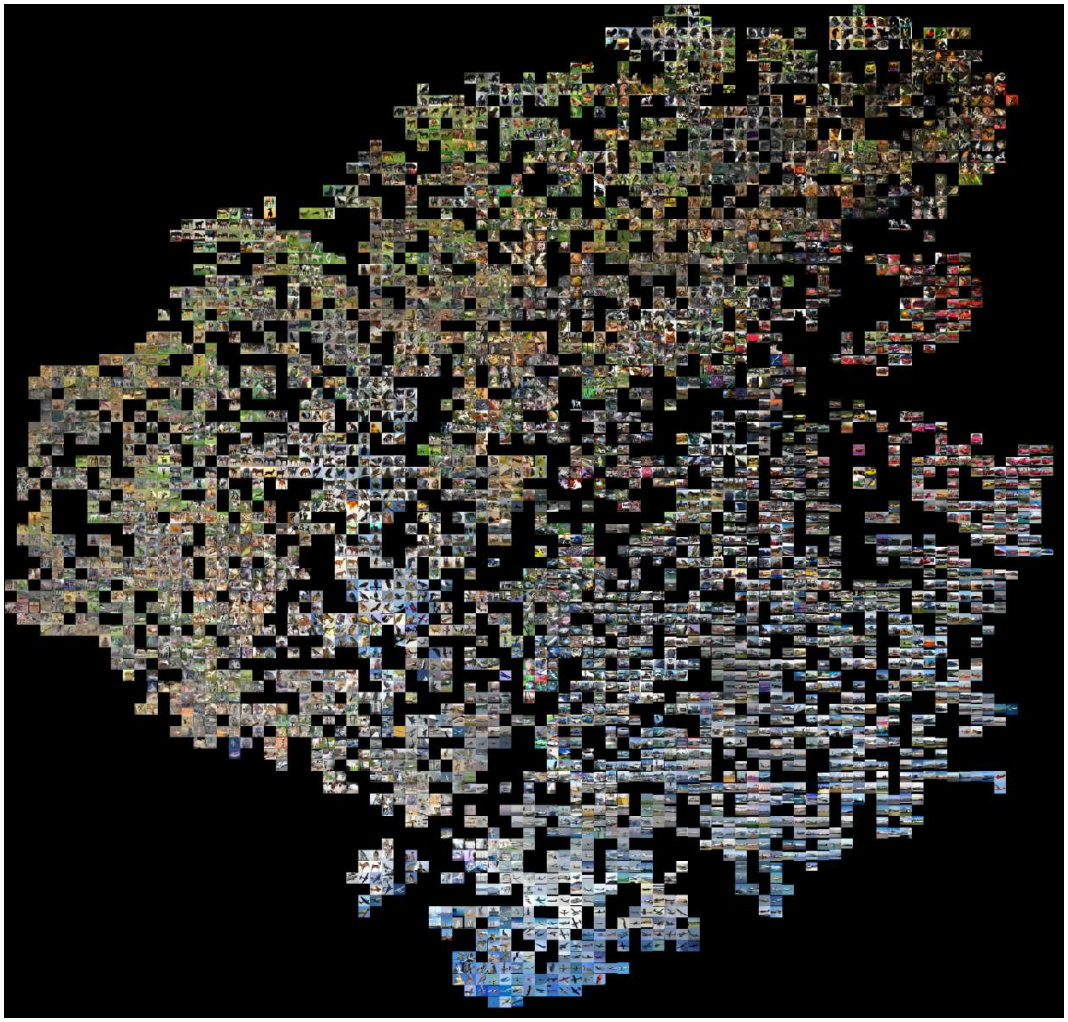
Bagian ini menunjukkan visualisasi dari *test set* STL-10 melalui t-SNE (Flexa et al., 2021). Langkah-langkah untuk melakukan visualisasi sama seperti bagian sebelumnya.



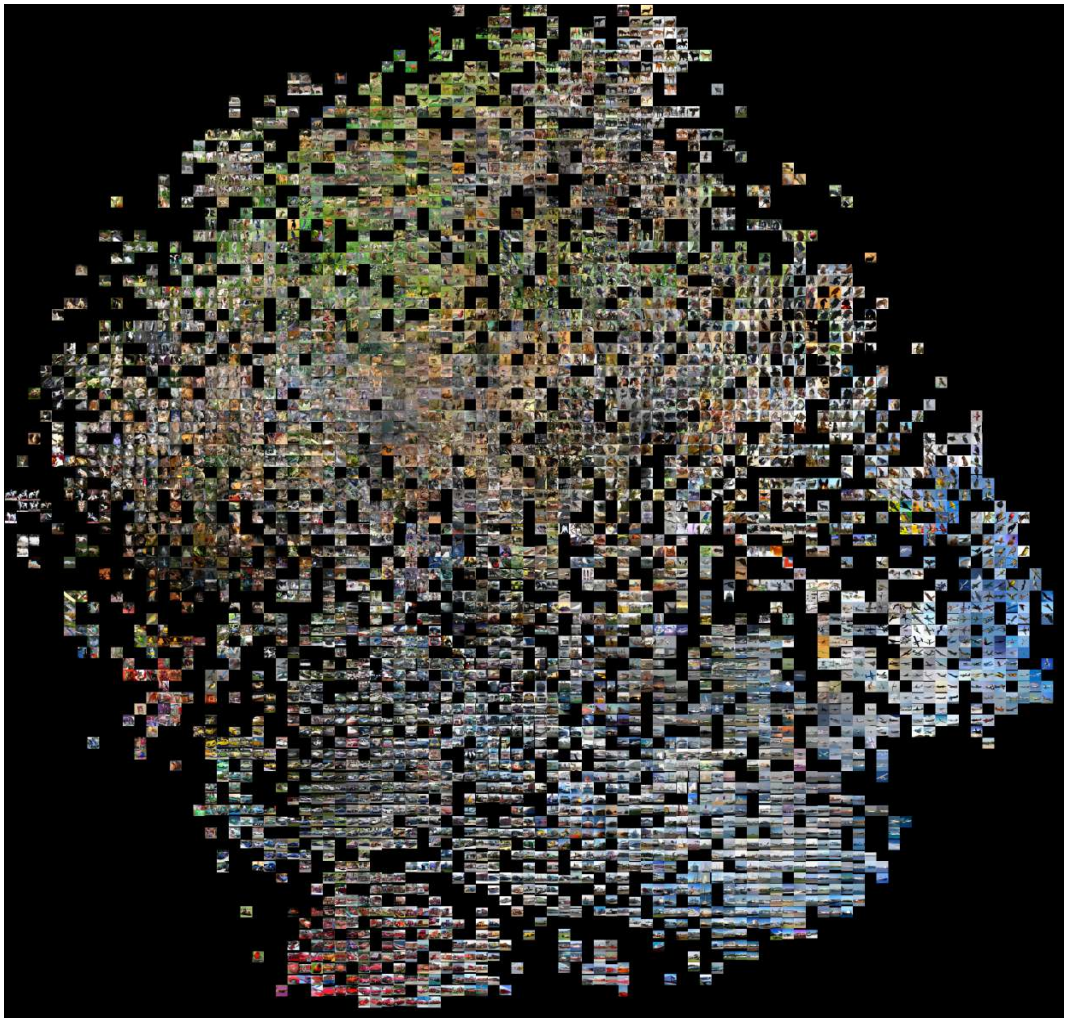
Gambar C.6: Visualisasi t-SNE AVT pada *dataset* STL-10.



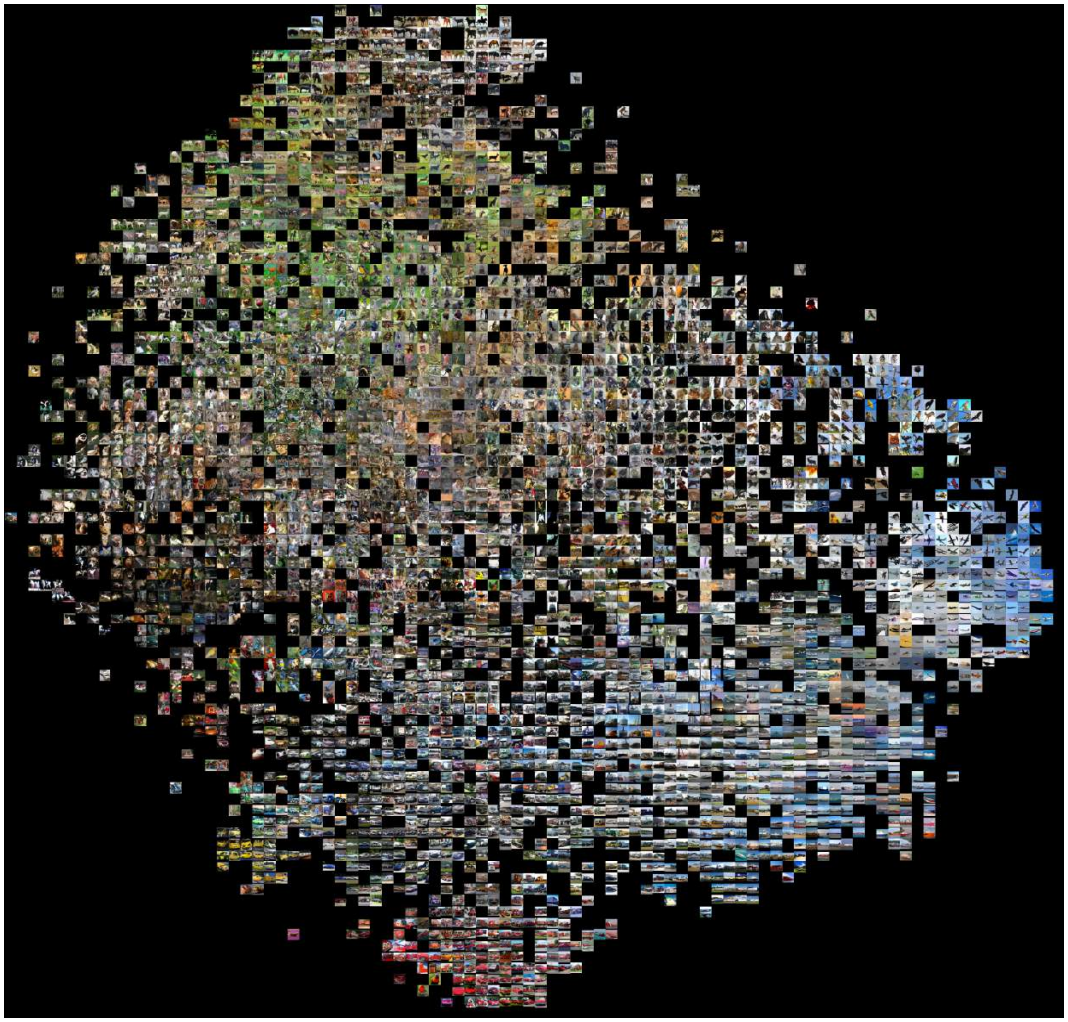
Gambar C.7: Visualisasi t-SNE *predictive-transformation* pada *dataset* STL-10.



Gambar C.8: Visualisasi t-SNE TVEBA pada *dataset* STL-10.



Gambar C.9: Visualisasi t-SNE TVEInfoNCE *separated* pada *dataset* STL-10.

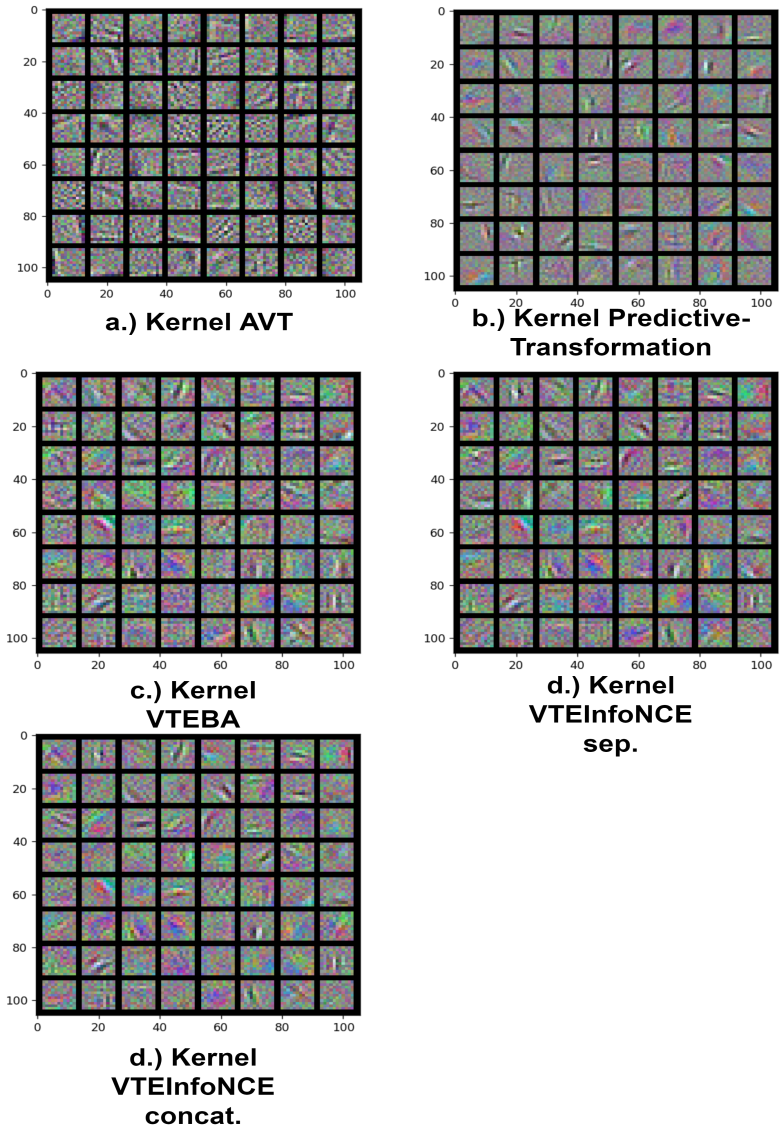


Gambar C.10: Visualisasi t-SNE TVEInfoNCE *concatenated* pada *dataset* STL-10.

APPENDIX D

D.1 Visualisasi Kernel: STL-10

Lampiran ini menunjukkan visualisasi kernel JSK pada lapisan pertama dari *encoder* masing-masing model representasi. Visualisasi hanya dilakukan pada Eksperimen STL-10 karena ukuran kernelnya yang cukup besar, yaitu 11×11 (kernel pada percobaan CIFAR-10 hanya berukuran 5×5).



Gambar D.1: Visualisasi kernel setiap model representasi pada *dataset* STL-10.

Audit Trail

DigiSigner Document ID: b6bef75a-1a66-41e3-a679-2eeab7a3ec4b

Signer	Signature
Email: chan@cs.ui.ac.id IP Address: 2404:8000:1005:d1e:38a2:58e4:8bd7:7c51	
Email: wisnuj@cs.ui.ac.id IP Address: 182.1.121.51	
Email: saptawijaya@cs.ui.ac.id IP Address: 101.128.87.184	
Email: adila@cs.ui.ac.id IP Address: 114.124.211.22	
Email: adila@cs.ui.ac.id IP Address: 114.124.211.22	
Email: fariz@ui.ac.id IP Address: 139.228.61.180	

Event	User	Time	IP Address
Upload document	marshal.arijona01@ui.ac.id	1/10/22 2:14:39 AM EST	114.122.43.88
Send for signing	marshal.arijona01@ui.ac.id	1/10/22 2:18:07 AM EST	114.122.43.88
Open document	chan@cs.ui.ac.id	1/10/22 2:21:34 AM EST	2404:8000:1005:d1e:5dbf:b28f:9ca5:a236
Open document	marshal.arijona01@ui.ac.id	1/10/22 2:22:47 AM EST	114.122.43.88
Open document	chan@cs.ui.ac.id	1/11/22 4:22:23 AM EST	2404:8000:1005:d1e:38a2:58e4:8bd7:7c51
Sign document	chan@cs.ui.ac.id	1/11/22 4:27:24 AM EST	2404:8000:1005:d1e:38a2:58e4:8bd7:7c51
Close document	chan@cs.ui.ac.id	1/11/22 4:27:24 AM EST	2404:8000:1005:d1e:38a2:58e4:8bd7:7c51
Open document	wisnuj@cs.ui.ac.id	1/11/22 7:09:02 AM EST	180.252.165.214
Open document	wisnuj@cs.ui.ac.id	1/11/22 7:24:41 AM EST	182.1.121.51
Open document	wisnuj@cs.ui.ac.id	1/11/22 7:25:52 AM EST	182.1.121.51
Sign document	wisnuj@cs.ui.ac.id	1/11/22 7:27:38 AM EST	182.1.121.51
Close document	wisnuj@cs.ui.ac.id	1/11/22 7:27:38 AM EST	182.1.121.51
Open document	saptawijaya@cs.ui.ac.id	1/11/22 8:19:10 PM EST	101.128.87.184
Sign document	saptawijaya@cs.ui.ac.id	1/11/22 8:20:09 PM EST	101.128.87.184
Close document	saptawijaya@cs.ui.ac.id	1/11/22 8:20:09 PM EST	101.128.87.184

Open document	adila@cs.ui.ac.id	1/13/22 2:24:18 AM EST	65.154.226.101
Open document	adila@cs.ui.ac.id	1/13/22 2:24:32 AM EST	65.154.226.101
Download document	marshal.arijona01@ui.ac.id	1/13/22 2:50:20 AM EST	114.122.39.108
Open document	marshal.arijona01@ui.ac.id	1/13/22 2:52:06 AM EST	114.122.39.108
Resend for signing	marshal.arijona01@ui.ac.id	1/13/22 2:52:20 AM EST	114.122.39.108
Resend for signing	marshal.arijona01@ui.ac.id	1/13/22 2:52:26 AM EST	114.122.39.108
Open document	adila@cs.ui.ac.id	1/13/22 3:26:11 AM EST	114.124.211.22
Sign document	adila@cs.ui.ac.id	1/13/22 3:26:58 AM EST	114.124.211.22
Close document	adila@cs.ui.ac.id	1/13/22 3:26:58 AM EST	114.124.211.22
Open document	fariz@ui.ac.id	1/13/22 12:13:48 PM EST	139.228.61.180
Sign document	fariz@ui.ac.id	1/13/22 12:14:40 PM EST	139.228.61.180
Close document	fariz@ui.ac.id	1/13/22 12:14:40 PM EST	139.228.61.180
Open document	marshal.arijona01@ui.ac.id	1/15/22 6:44:51 AM EST	114.122.38.182
Close document	marshal.arijona01@ui.ac.id	1/15/22 6:44:52 AM EST	114.122.38.182